Xavier Chanet
Patrick Vert

Mathématiques pour l'informatique

Pour le BTS SIO

- Rappels de cours
- Exercices corrigés
- TD, TP
- Annales corrigées

RESSOURCES NUMÉRIQUES

DUNOD

Toutes les marques citées dans cet ouvrage sont des marques déposées par leurs propriétaires respectifs.

Illustration de couverture :
©iStock.com/ahlobystov

5 rue Laromiguière, 75005 Paris
www.dunod.com
ISBN 978-2-10-072807-7

TABLE DES MATIÈRES

Avant-propos VII

PARTIE I

MATHÉMATIQUES

Chapitre 1 • Arithmétique 3
1.1 Numération et conversion 3
1.2 Divisibilité des entiers 8
1.3 Nombres premiers 8
1.4 Congruences 11

TD – Le codage affine 13

Exercices corrigés 16

Chapitre 2 • Suites numériques 35
2.1 Généralités 35
2.2 Suites particulières 36
2.3 Variations d'une suite 40
2.4 Limite d'une suite 42

TD – Évolution d'une liste de diffusion 44

Exercices corrigés 46

Chapitre 3 • Calcul matriciel 61
3.1 Généralités 61
3.2 Calcul matriciel élémentaire 62
3.3 Inverse d'une matrice carrée 68
3.4 Résolution de systèmes à l'aide de matrices 69

Exercices corrigés 70

Chapitre 4 • Logique 87
4.1 Calcul des propositions 87
4.2 Calcul des prédicats 92
4.3 Calcul booléen 95

TD – Expression booléenne 100

Exercices corrigés 103

Chapitre 5 · Ensembles **121**
5.1 Langage ensembliste 121
5.2 Relations binaires 124
5.3 Applications d'un ensemble dans un ensemble 126
TD – Relation binaire dans un ensemble **129**
Exercices corrigés **132**
Chapitre 6 · Graphes et ordonnancement **147**
6.1 Représentations d'un graphe 147
6.2 Chemins d'un graphe 150
6.3 Niveau des sommets d'un graphe sans circuit 155
6.4 Méthode MPM d'ordonnancement d'un graphe 158
TD – Déplacements dans un jeu vidéo **163**
Exercices corrigés **166**
Chapitre 7 · L'examen de mathématiques **183**
Sujet métropole 2014 183

PARTIE II

ALGORITHMIQUE APPLIQUÉE

Chapitre 8 · Premiers pas **189**
8.1 Qu'est-ce qu'un algorithme ? 189
8.2 Le logiciel Python 190
Chapitre 9 · Concepts fondamentaux **191**
9.1 Données : types et opérations 191
9.2 Stockage des données 196
9.3 Lecture et écriture des données 198
9.4 Instructions conditionnelles 200
9.5 Instructions itératives 201
9.6 Fonctions et procédures 204
9.7 Récursivité 208
Exercices corrigés **210**
Chapitre 10 · Travaux pratiques **223**
10.1 Nombres parfaits 223
10.2 Évolution d'un salaire 226
10.3 Nombres premiers palindromes 229
10.4 Calcul formel 232
10.5 Calcul matriciel 234
10.6 Opérations sur les ensembles 238

10.7 Méthodes de tri 242
10.8 Cryptographie 246

Chapitre 11 • L'examen d'algorithmique **249**
Examen 1 – Remplissage d'une tirelire 250
Examen 2 – La suite de Syracuse 254

PARTIE III

ANNEXE

Exercices supplémentaires **261**

Ressources numériques

Le **code source** des exemples est disponible gratuitement en téléchargement à l'adresse suivante :
www.dunod.com/contenus-complementaires/9782100720750

Avant-propos

Ce livre s'adresse en premier lieu aux étudiants de première et de deuxième année préparant le BTS SIO (Services informatiques aux organisations). Il pourra également intéresser les étudiants en IUT d'informatique, ou ceux en classe préparatoire souhaitant acquérir les bases de l'algorithmique, ainsi que tous ceux qui souhaitent connaître et maîtriser les outils mathématiques nécessaires à une bonne pratique de la programmation informatique.

Les auteurs, tous deux enseignants en BTS SIO, ont rédigé cet ouvrage dans le respect le plus strict des derniers programmes en vigueur. L'objectif pédagogique majeur est de fournir un outil d'accompagnement dans les apprentissages, pouvant être utilisé en classe par le professeur ou de manière plus personnelle par l'étudiant.

Dans la partie *Mathématiques*, on trouvera dans chaque chapitre, le cours, présentant les notions essentielles du programme, des exercices, nombreux et variés, corrigés ou non, allant des applications directes du cours à des problèmes plus complexes pour se perfectionner, et des travaux dirigés corrigés.

Dans la partie *Algorithmique appliquée*, où les instructions et algorithmes sont exécutés en langage Python, on commence par présenter expérimentalement avec les activités dites « de découverte », les fondamentaux de l'algorithmique. L'étudiant peut ensuite vérifier qu'il maîtrise les concepts clés en résolvant les nombreux exercices, corrigés ou non. Une série de travaux pratiques, tous corrigés, montrent comment résoudre des problèmes par l'utilisation judicieuse de solutions algorithmiques. On trouve enfin, deux exemples de sujets officiels d'examen d'algorithmique appliquée, corrigés, donnés lors de la session 2014.

Ressources numériques

Le **code source** des exemples est disponible gratuitement en téléchargement à l'adresse suivante :
www.dunod.com/contenus-complementaires/9782100720750

Partie 1

Mathématiques

ARITHMÉTIQUE 1

PLAN

1.1 Numération et conversion
1.2 Divisibilité des entiers
1.3 Nombres premiers
1.4 Congruences

OBJECTIFS

- Présenter les grandes notions arithmétiques utiles à l'informatique.
- Maîtriser les principes de numération indispensables aux langages de bas niveau.
- Maîtriser les outils d'arithmétique modulaire utiles à l'algorithmique.

1.1 NUMÉRATION ET CONVERSION

1.1.1 Rappels sur la division euclidienne

Les **entiers naturels** sont 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, etc.

Effectuer la **division euclidienne** d'un entier naturel A par un entier naturel B non nul c'est déterminer les uniques entiers Q (appelé **quotient**) et R (appelé **reste**) tels que : $A = BQ + R$ et $0 \leq R < B$.

Exemples

Le quotient et le reste de la division euclidienne de $A = 53$ par $B = 6$ sont respectivement $Q = 8$ et $R = 5$. En effet, $53 = 6 \times 8 + 5$ et $0 \leq 5 < 6$.
Dans la division euclidienne de 1 893 par 11, le quotient vaut 172 et le reste vaut 1.

On peut obtenir ces résultats en posant la division ou avec une calculatrice.

1.1.2 Numération des entiers

L'être humain compte naturellement **en base 10** (avec les dix chiffres) : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, … , 97, 98, 99, 100, 101, 102, etc.

On peut compter **en base 2** (on n'utilise que les chiffres 0 et 1) : 0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, etc.

Pour compter **en base 16**, on utilise les dix chiffres et on en rajoute six autres (que l'on note *A, B, C, D, E* et *F*). Cela donne : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, *A, B, C, D, E, F*, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1*A*, 1*B*, 1*C*, 1*D*, 1*E*, 1*F*, 20, 21, 22, etc.

Exemples

4 en base 10, c'est 100 en base 2.
A en base 16, c'est 10 en base 10.
9 en base 10, c'est 1001 en base 2.
1A en base 16, c'est 26 en base 10.

Le microprocesseur d'un ordinateur ne travaille qu'avec deux chiffres : 0 (pas de courant) et 1 (courant). Les calculs s'y font donc naturellement en base 2.

Le nombre 23, écrit en base 10, se note 10111 en base 2 et 17 en base 16, mais on ne peut pas écrire 23 = 10111 = 17.

C'est pourquoi nous adoptons la **notation** $(n)_p$ pour indiquer que le nombre n est écrit en base p : $(23)_{10} = (10111)_2 = (17)_{16}$.

Les écritures en bases 2, 10 et 16 s'appellent aussi respectivement les écritures **binaire**, **décimale** et **hexadécimale**.

Lorsqu'un nombre est écrit en base 10, on peut simplifier la notation $(x)_{10}$. Par exemple $(201)_{10}$ peut s'écrire simplement 201.

L'écriture d'un nombre entier en base 2, 10 ou 16 est unique.

Pour convertir un entier d'une base à l'autre, il est important de comprendre la relation algébrique que l'on a entre une base p et l'écriture du nombre dans cette base p. C'est ce qu'énonce la propriété suivante.

Propriété 1.1

Quels que soient les nombres entiers $a_0, a_1, a_2, \ldots, a_n$ compris entre 0 et $p - 1$, où p désigne 2, 10 ou 16, on a :
$(a_n...a_2a_1a_0)_p = a_n \times p^n + ... + a_2 \times p^2 + a_1 \times p + a_0$

Dans le cas où $p = 16$, on remplace dans l'écriture du membre de gauche, tout a_i égal à 10, 11, 12, 13, 14 ou 15 par A, B, C, D, E ou F respectivement.

Exemple

Conversion vers la base 10 :
$(11011)_2 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1 = 16 + 8 + 2 + 1 = 27$
$(5C8)_{16} = 5 \times 16^2 + 12 \times 16 + 8 = 1\,280 + 192 + 8 = 1\,480$

Il existe plusieurs méthodes pour convertir un entier vers la base p = 2 ou 16. Nous allons déterminer l'écriture en base 2 du nombre 75 et l'écriture en base 16 du nombre 2014 (méthodes 1 et 2), puis effectuer des conversions directes entre les bases 2 et 16 (méthode 3).

Méthode 1 : on effectue la division euclidienne du nombre par la plus grande puissance de *p* qui lui est inférieure ou égale, puis on recommence avec le reste et ainsi de suite jusqu'à ce qu'il soit nul.

$75 = 1 \times 2^6 + 11$, $11 = 1 \times 2^3 + 3$ et $3 = 1 \times 2 + 1$

$$
\begin{aligned}
\text{Donc } 75 &= 1 \times 2^6 + 1 \times 2^3 + 1 \times 2 + 1 \\
&= 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1 \\
&= (1001011)_2
\end{aligned}
$$

De même $2\,014 = 7 \times 16^2 + 222$ et $222 = 13 \times 16 + 14$ d'où $222 = 7 \times 16^2 + 13 \times 16 + 14 = (7DE)_{16}$.

Méthode 2 : on effectue la division euclidienne du nombre par *p* puis on recommence avec le quotient et ainsi de suite jusqu'à obtenir 0. À la fin, on inverse l'ordre des restes obtenus.

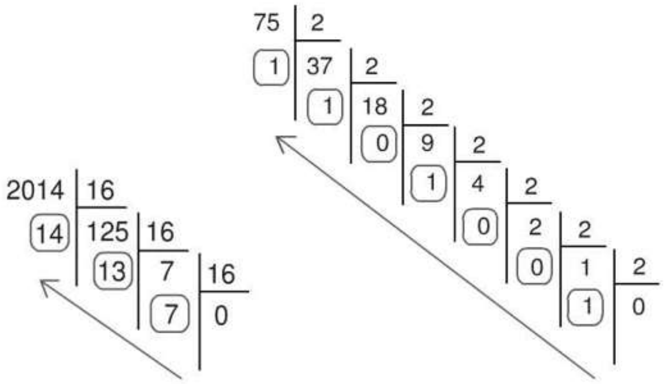


Figure 1.1

Ainsi, on retrouve $2\,014 = (7DE)_{16}$ et $75 = (1001011)_2$.

Méthode 3 : on peut passer directement de la base 2 à la base 16, et inversement, en utilisant le tableau de conversion suivant :

Base 16	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Base 2	0000	0001	0010	0011	0100	0101	0110	0111
Base 16	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
Base 2	1000	1001	1010	1011	1100	1101	1110	1111

Conversion directe de $(2B)_{16}$ en base 2 :

$$(2B)_{16} = (00101011)_2 = (101011)_2$$

On ajoute, à partir du tableau, les écritures de 2 et de B en base 2, puis on supprime les zéros inutiles.

Conversion directe de $(110111)_2$ en base 16 :

$$(110111)_2 = (00110111)_2 = (37)_{16}$$

On ajoute à gauche des zéros inutiles pour « faire des paquets » de quatre chiffres que l'on remplace ensuite par leurs équivalents en base 16.

Puisque l'écriture en base 2 d'un nombre peut être très « longue » et du fait de la simplicité de conversion entre les bases 2 et 16, on utilise beaucoup le **système hexadécimal** en informatique.

Les calculatrices permettent de convertir un nombre d'une base à l'autre.

1.1.3 Numération des réels

Dans le dernier paragraphe, nous avons défini les écritures binaire, décimale et hexadécimale d'un nombre entier. Plus largement, tout nombre réel peut être écrit en base 2, 10 ou 16.

Pour comprendre, prenons l'exemple de deux réels et de la base 10 :

$(53{,}627)_{10} = 5 \times 10 + 3 + 6/10 + 2/10^2 + 7/10^3$

$(456{,}905)_{10} = 4 \times 10^2 + 5 \times 10 + 6 + 9/10 + 0/10^2 + 5/10^3$

Plus généralement on a la propriété suivante.

Propriété 1.2

Quels que soient les nombres entiers $a_0, \ldots, a_n$ et $b_1, \ldots, b_m$ compris entre 0 et $p-1$ (où p désigne 2, 10 ou 16), on a :

$$\left(a_n \ldots a_0, b_1 \ldots b_m\right)_p = a_n \times p^n + \ldots + a_2 \times p^2 + a_1 \times p + a_0 + \frac{b_1}{p} + \frac{b_2}{p^2} + \ldots + \frac{b_m}{p^m}$$

Dans le cas où p = 16, on remplace dans l'écriture du membre de gauche, tout a_i ou b_i égal à 10, 11, 12, 13, 14 ou 15 par A, B, C, D, E ou F respectivement.

Exemple

7,25 en base 2 : $7{,}25 = 7 + 0{,}25 = 1 \times 2^2 + 1 \times 2 + 1 + 1/2^2 = (111{,}01)_2$
$(101{,}11)_2$ en base 10 : $(101{,}11)_2 = 1 \times 2^2 + 0 \times 2 + 1 + 1/2 + 1/2^2 = 5{,}75$
$(B{,}3C)_{16}$ en base 10 : $(B{,}3C)_{16} = 11 + 3/16 + 12/16^2 = 11{,}234375$

1.1.4 Opérations élémentaires

En base 2 ou 16, on pose les opérations de la même manière qu'en base 10.

- L'**addition** : exemple $(101)_2 + (1110)_2 = (10011)_2$ (en base 10, cela donne $5 + 14 = 19$).

$$\begin{array}{cccccc}
 & & 1 & 1 & & \\
 & & & 1 & 0 & 1 \\
+ & & 1 & 1 & 1 & 0 \\
\hline
 & 1 & 0 & 0 & 1 & 1
\end{array}$$

$(1)_2 + (0)_2 = (1)_2$
$(0)_2 + (1)_2 = (1)_2$
$(1)_2 + (1)_2 = (10)_2$: on pose 0, on retient 1.
$(1)_2 + (1)_2 = (10)_2$: on pose 0, on retient 1.

- La **soustraction** : exemple $(10110)_2 - (111)_2 = (1111)_2$ (en base 10 cela donne $22 - 7 = 15$).

$$\begin{array}{cccccc}
 & & 1 & 1 & 1 & 1 \\
 & 1 & 0 & 1 & 1 & 0 \\
- & & & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & \\
\hline
 & 0 & 1 & 1 & 1 & 1
\end{array}$$

Puisque 1 > 0, on pose une retenue de 1, puis $(10)_2 - (1)_2 = (1)_2$.
Nouvelle retenue de 1 puis $(11)_2 - ((1)_2 + (1)_2) = (1)_2$, etc.

- La **multiplication** : exemple $(F3)_{16} \times (2A)_{16} = (27DE)_{16}$ (en base 10 cela donne $42 \times 243 = 10\,206$).

$$\begin{array}{ccccc}
 & & & F & 3 \\
 & \times & & 2 & A \\
\hline
 & & 9 & 7 & E \\
+ & 1 & E & 6 & . \\
\hline
 & 2 & 7 & D & E
\end{array}$$

$(A)_{16} \times (3)_{16} = 10 \times 3 = 30 = (1E)_{16}$, on pose E, on retient 1.
$(A)_{16} \times (F)_{16} = 10 \times 15 = 150 = (96)_{16}$, avec la retenue, on obtient, $(96)_{16} + (1)_{16} = (97)_{16}$, d'où 97E.
$(2)_{16} \times (3)_{16} = (6)_{16}$, on pose 6 .
$(2)_{16} \times (F)_{16} = 2 \times 15 = 30 = (1E)_{16}$, on obtient 1E6 avec le décalage « . ».
On effectue ensuite l'addition $(97E)_{16} + (1E60)_{16}$: $(E)_{16} + (0)_{16} = (E)_{16}$, on pose E ; $(7)_{16} + (6)_{16} = 7 + 6 = 13 = (D)_{16}$, on pose D ; $(9)_{16} + (E)_{16} = 9 + 14 = 23 = (17)_{16}$, on pose 7 et on retient 1 ; enfin, $(1)_{16} + (1)_{16} = (2)_{16}$, on obtient alors le résultat 27DE.

- La **division** : exemple $(110)_2 \div (11)_2 = (10)_2$ (en base 10 cela donne $6 \div 2 = 3$).

$$\begin{array}{rrr|rr}
1 & 1 & 0 & 1 & 1 \\
-1 & 1 & & \multicolumn{2}{l}{\overline{}} \\
\cline{1-2}
 & 0 & 0 & 1 & 0 \\
\cline{2-3}
 & 0 & 0 & &
\end{array}$$

On sélectionne les deux premiers chiffres de 110. Dans $(11)_2$ on a une fois $(11)_2$ et il reste 0. On pose alors 1 au quotient. On abaisse ensuite le 0 de 110. On obtient 00. Dans $(00)_2$ on a zéro fois $(11)_2$ et il reste 0. On pose 0 au quotient, ce qui termine la division.

1.1.5 Précision et arrondi

Considérons le nombre $N = 432{,}615$:

- L'arrondi à **10** près de N est 430 (car ce nombre est plus proche de N que 440).
- L'arrondi à **1** près de N est 433 (car ce nombre est plus proche de N que 432).
- L'arrondi à **0,1** près de N est 432,6 (car ce nombre est plus proche de N que 432,7).
- L'arrondi à **0,01** près de N est 432,62 (ce nombre est aussi plus proche de N que 432,61 et dans un tel cas on prend le plus grand).

Cette **notion d'arrondi** se généralise aux nombres réels écrits dans n'importe quelle base p : l'arrondi d'un nombre réel x à une certaine précision est le nombre le plus proche de x tel que tous les chiffres allant au-delà de cette précision soient nuls. Par convention, lorsqu'il existe deux nombres possibles, l'arrondi est alors le plus grand.

Considérons les nombres $N = (101{,}10011)_2$ et $M = (B82A{,}7AB)_{16}$:

- L'arrondi à $(10)_2$ près de N est $(110)_2$ car ce nombre est plus proche de N que $(100)_2$.
- L'arrondi à $(0{,}1)_2$ près de N est $(101{,}1)_2$ car ce nombre est plus proche de N que $(110{,}0)_2$.
- L'arrondi à $(100)_{16}$ près de M est $(B800)_{16}$ car ce nombre est plus proche de M que $(B900)_{16}$.
- L'arrondi à $(0{,}1)_{16}$ près de M est $(B82A{,}8)_{16}$ car ce nombre est plus proche de M que $(B82A{,}7)_{16}$.

1.2 Divisibilité des entiers

Soit a et b des entiers naturels. a est **divisible** par b s'il existe un entier naturel k tel que : $a = bk$.

On dit alors que b divise a, que b est un **diviseur** de a et que a est un **multiple** de b.

Exemple

$65 = 13 \times 5$ donc on peut dire que 65 est divisible par 13 (et par 5 aussi), que 13 et 5 sont des diviseurs de 65 et que 65 est un multiple de 13 (et de 5 aussi).

Propriété 1.3

– Si a est divisible par b alors tout multiple de a est divisible par b.
– Si a est divisible par b et si b est divisible par c, alors a est divisible par c.
– Si a et b sont divisibles par c, alors $a+b$ et $a-b$ sont divisibles par c.

Les démonstrations de ces propriétés seront proposées dans l'exercice corrigé **1.20**.

1.3 Nombres premiers

1.3.1 Reconnaissance des nombres premiers

Un **entier naturel** est premier s'il a exactement deux diviseurs : 1 et lui-même.

- 0 n'est pas premier car il possède une infinité de diviseurs.
- 1 n'est pas premier car il n'a qu'un seul diviseur : 1.
- 2 est **premier** car il a exactement deux diviseurs : 1 et 2.
- 3 est **premier** car il a exactement deux diviseurs : 1 et 3.

Théorème 1.1

Soit n un entier naturel supérieur ou égal à 2.
Si n n'est pas premier, alors n possède au moins un diviseur premier inférieur ou égal à $\sqrt{n}$.

Ce théorème permet d'avoir un critère de reconnaissance des nombres premiers : pour savoir si un nombre n est premier, on calcule $\sqrt{n}$. Si n n'est divisible par aucun des nombres premiers inférieurs ou égaux à $\sqrt{n}$, alors n est premier.

Mémorisez les nombres premiers inférieurs à 20, ils vous seront utiles par la suite : 2, 3, 5, 7, 11, 13, 17 et 19.

Exemple

Cherchons si 347 et 713 sont des nombres premiers :

$\sqrt{347} \approx 18{,}6$. Les nombres premiers inférieurs ou égaux à 18,6 sont 2, 3, 5, 7, 11, 13 et 17. 347 n'est divisible par aucun d'eux donc **347 est premier**.

$\sqrt{713} \approx 26{,}7$. Les nombres premiers inférieurs ou égaux à 26,7 sont 2, 3, 5, 7, 11, 13, 17, 19 et 23. 713 est divisible par 23 donc **713 n'est pas premier**.

1.3.2 Décomposition d'un nombre en produit de facteurs premiers

Un nombre qui n'est pas premier est divisible par un nombre premier, et peut même s'écrire uniquement avec des nombres premiers, sous forme de produit.

Théorème 1.2

Tout entier naturel supérieur ou égal à 2 se décompose de façon unique en un produit de facteurs premiers.

Exemple

84 n'est pas premier, il se décompose en un produit de facteurs premiers, cette décomposition est unique, à l'ordre des facteurs près :
$84 = 2 \times 42 = 2 \times 6 \times 7 = 2 \times 2 \times 3 \times 7 = 2^2 \times 3 \times 7$
975 n'est pas premier, il se décompose en un produit de facteurs premiers :
$975 = 5 \times 195 = 5 \times 5 \times 39 = 5 \times 5 \times 3 \times 13 = 3 \times 5^2 \times 13$

En pratique, on divise le nombre par son **plus petit diviseur premier**, et on recommence jusqu'à ce que le quotient soit 1.

84	2	975	3
42	2	325	5
21	3	65	5
7	7	13	13
1		1	

La décomposition d'un nombre en produit de facteurs premiers permet d'obtenir tous ses diviseurs. Par exemple, $84 = 2^2 \times 3 \times 7$, un diviseur de 84 s'écrit donc : $2^i \times 3^j \times 7^k$ avec $0 \leq i \leq 2$, $0 \leq j \leq 1$, $0 \leq k \leq 1$.

En utilisant l'algorithme suivant, on obtient les 12 diviseurs de 84 qui sont 1, 2, 3, 4, 6, 7, 12, 14, 21, 28, 42 et 84.

```
Variables : i, j, k (entiers)
Début
    Pour i de 0 à 2 Faire
        Pour j de 0 à 1 Faire
            Pour k de 0 à 1 Faire
                Afficher 2^i × 3^j × 7^k
            FinPour
        FinPour
    FinPour
Fin
```

1.3.3 PGCD de deux entiers naturels non nuls

Étant donné deux entiers naturels non nuls a et b, il existe un diviseur commun à a et à b qui est plus grand que tous les autres. Ce diviseur est appelé **Plus Grand Commun Diviseur** et se note PGCD($a;b$).

Deux entiers naturels non nuls sont **premiers entre eux** si et seulement si leur PGCD est égal à 1.

Exemples

Les diviseurs de 45 sont 1, 3, 5, 9, 15 et 45. Ceux de 105 sont 1, 3, 5, 7, 15, 21, 35 et 105. Les diviseurs communs à 45 et 105 sont 1, 3, 5, 15.
Le plus grand d'entre eux est 15 donc 15 est le PGCD de 45 et 105.
On écrit PGCD(45;105) = 15.
Les diviseurs de 15 sont 1, 3, 5 et 15. Ceux de 44 sont 1, 2, 4, 11, 22 et 44.
Le seul diviseur commun à 15 et à 44 est 1. Il est leur PGCD.
Donc PGCD(15;44) = 1.
15 et 44 sont premiers entre eux.

Propriété 1.4

Soit a et b deux entiers naturels supérieurs ou égaux à 2 dont on connaît les décompositions en produits de facteurs premiers :
– s'ils n'ont pas de facteur commun, alors leur PGCD est 1 ;
– sinon, leur PGCD est le produit des facteurs communs aux deux décompositions, chaque facteur étant affecté du plus petit exposant avec lequel il figure dans les deux décompositions.

Exemples

Soit a = 4 950 = $2 \times 3^2 \times 5^2 \times 11$ et b = 4 875 = $3 \times 5^3 \times 13$.
Les facteurs communs aux deux décompositions sont 3 et 5.
3 figure avec les exposants 2 et 1, on garde le plus petit, c'est-à-dire 1.
5 figure avec les exposants 2 et 3, on garde le plus petit, c'est-à-dire 2.
Donc PGCD (4 950;4 875) = $3 \times 5^2 = 75$.
Soit $a = 2^4 \times 3 \times 5^2 \times 7 \times 19$ et $b = 2^3 \times 3^2 \times 7^2 \times 17$. Les facteurs communs aux deux décompositions sont 2, 3 et 7. Les plus petits exposants sont 3 pour le facteur 2, 1 pour le facteur 3 et 1 pour le facteur 7.
Donc PGCD $(a;b) = 2^3 \times 3 \times 7$.

Propriété 1.5

Soit a et b deux entiers naturels non nuls tels que $a > b$.
Soit r le reste de la division euclidienne de a par b.
Alors PGCD(a;b) = PGCD(b;r)

Cette propriété (1.5) permet d'avoir une autre méthode pour chercher un PGCD. On applique plusieurs fois la propriété jusqu'à obtenir un reste nul. Le PGCD est alors le dernier diviseur essayé.

Son avantage est qu'elle est algorithmique, donc programmable. Elle est connue sous le nom **d'algorithme d'Euclide**.

Exemple

PGCD(420;182) = PGCD(182;56) car 56 est le reste de la division euclidienne de 420 par 182.
PGCD(182;56) = PGCD(56;14) car 14 est le reste de la division euclidienne de 182 par 56.
Le reste de la division euclidienne de 56 par 14 est 0, donc 14 est le PGCD de 420 et 182.

1.4 Congruences

Soit n un entier naturel non nul. On dit que deux entiers naturels ***a* et *b* sont congrus modulo *n*** si et seulement si a et b ont le même reste dans la division euclidienne par n. On écrit alors $a \equiv b[n]$ ou $b \equiv a[n]$.

Quel que soit a entier, $a \equiv a[n]$ et $a \equiv r[n]$, r étant le reste de la division euclidienne de a par n.

Exemples

$101 = 7 \times 14 + 3$ et $66 = 7 \times 9 + 3$
donc 101 et 66 sont congrus modulo 7 car ils ont le même reste dans les divisions euclidiennes par 7. On peut donc écrire $101 \equiv 66\ [7]$.
$67 = 12 \times 5 + 7$ et $139 = 12 \times 11 + 7$
donc $67 \equiv 139\ [12]$ car le reste est le même dans les divisions euclidiennes par 12.
$29 = 8 \times 3 + 5$ donc $29 \equiv 5\ [8]$.
90 et 80 ne sont pas congrus modulo 11, car les restes dans les divisions euclidiennes par 11 sont 2 et 3.

Propriété 1.6

Soit a et b deux entiers naturels tels que $a > b$ et soit n un entier naturel non nul.
a est congru à b modulo n si et seulement si $a - b$ est multiple de n.

En effet, si $a \equiv b[n]$ alors a et b ont le même reste r dans la division euclidienne par n.

Donc on a $a = nq + r$ et $b = nq' + r$ puis $a - b = (nq + r) - (nq' + r) = nq - nq'$ $= n(q - q')$ qui est un multiple de n puisque $q-q'$ est un entier.

Propriété 1.7

Soit a, b, c, d des entiers naturels et n un entier naturel non nul.

Si $a \equiv b[n]$ et $c \equiv d[n]$ alors :

– $a + c \equiv b + d[n]$ et $a - c \equiv b - d[n]$

– $pa \equiv pb[n]$ pour tout entier naturel p

– $ac \equiv bd[n]$

– $a^p \equiv b^p[n]$ pour tout entier naturel p

Des démonstrations de certaines de ces propriétés sont proposées dans l'exercice corrigé **1.45**.

Exemples

À partir des congruences $2\,014 \equiv 1[3]$ et $1\,000 \equiv 1[3]$, les propriétés précédentes permettent d'obtenir facilement d'autres congruences :
$2\,014 + 1\,000 \equiv 1 + 1[3]$ c'est-à-dire $3014 \equiv 2[3]$
$20 \times 2\,014 \equiv 20 \times 1[3]$ c'est-à-dire $40\,280 \equiv 20[3]$ donc $40\,280 \equiv 2[3]$
$2\,014 \times 1\,000 \equiv 1 \times 1[3]$ c'est-à-dire $2\,014\,000 \equiv 1[3]$
$2\,014^{70} \equiv 1^{70}[3]$ c'est-à-dire $2\,014^{70} \equiv 1[3]$ (sur cet exemple, on vient de montrer très facilement que le reste de la division euclidienne de $2\,014^{70}$ par 3 est 1, ce qui n'avait rien d'évident...).

TD – Le codage affine

Le chiffrement affine est une méthode simple de codage d'un message. À chaque lettre de l'alphabet, on commence par associer son rang dans l'alphabet, diminué de 1, comme l'indique le tableau 1.1. On obtient un entier x entre 0 et 25.

Tableau 1.1

Lettre	A	B	C	D	E	F	G	H	I	J	K	L	M
Nombre	0	1	2	3	4	5	6	7	8	9	10	11	12
Lettre	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Nombre	13	14	15	16	17	18	19	20	21	22	23	24	25

Le codage affine nécessite deux clés a et b, qui sont des entiers naturels compris entre 0 et 25. On calcule alors le reste de $ax + b$ dans la division euclidienne par 26. On obtient un entier y tel que $y \equiv ax + b[26]$. On cherche à quelle lettre correspond cet entier y. Cette lettre code alors la lettre de départ.

Partie A

Dans cette partie, on choisit les clés $a = 3$ et $b = 11$. La fonction de codage est donc $y \equiv 3x + 11[26]$.

1. Montrer que G est codé par D. Comment est codé S ?
2. Remplir le tableau 1.2.

Tableau 1.2

Lettre	A	B	C	D	E	F	G	H	I	J	K	L	M
x							6						
y							3						
Codage							D						
Lettre	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
x													
y													
Codage													

3. Quel mot est codé par VBUTSB ?
4. On va maintenant chercher la fonction de décodage, c'est-à-dire l'expression de x en fonction d'y. Chercher l'inverse de 3 modulo 26, c'est-à-dire le nombre entier k tel que $0 \leq k \leq 25$ et $3k \equiv 1[26]$. En déduire la fonction de décodage.

Partie B

Dans cette partie, on choisit $a = 7$ et $b = 12$.

1. Comment va-t-on coder le mot AFFINE ?
2. Déterminer la fonction de décodage.
3. Décoder le message CBMNJ QISO.

Partie C

Dans cette partie, on ne connaît pas les clés de codage a et b. On sait que E est codé par I et que V est codé par T.

1. Écrire les deux congruences vérifiées par a et b.
2. Montrer que $17a \equiv 11[26]$. Déterminer a puis b, puis la fonction de codage.
3. Déterminer la fonction de décodage.

Solution du TD

Partie A

1. Pour la lettre G, on a $x = 6$. $3x + 11 = 29$. Comme $29 \equiv 3[26]$, alors $y = 3$ ce qui correspond à la lettre D. G est donc codé par D.
   Pour S, $x = 18$, $3x + 11 = 65 \equiv 13[26]$ donc $y = 13$. S est donc codé par N.
2.

Tableau 1.3

Lettre	A	B	C	D	E	F	G	H	I	J	K	L	M
x	0	1	2	3	4	5	6	7	8	9	10	11	12
y	11	14	17	20	23	0	3	6	9	12	15	18	21
Codage	L	O	R	U	X	A	D	G	J	M	P	S	V
Lettre	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
x	13	14	15	16	17	18	19	20	21	22	23	24	25
y	24	1	4	7	10	13	16	19	22	25	2	5	8
Codage	Y	B	E	H	K	N	Q	T	W	Z	C	F	I

3. Pour décoder le mot **VBUTSB**, il suffit de repérer les correspondances du tableau. Le mot codé par est VBUTSB est MODULO.

4. L'inverse de 3 est 9 modulo 26, car $3 \times 9 = 27 \equiv 1[26]$.
   $y \equiv 3x + 11[26] \Rightarrow 9y \equiv 27x + 99[26] \Rightarrow 9y \equiv x + 21[26]$
   $\Rightarrow 9y - 21 \equiv x[26] \Rightarrow x \equiv 9y + 5[26]$.

La fonction de décodage est donc $x \equiv 9y + 5[26]$. Par exemple, pour décoder la lettre D, on a $y = 3$, $9y + 5 = 32 \equiv 6[26]$, donc $x = 6$, ce qui correspond à la lettre G.

Partie B

1. La fonction de codage est $y \equiv 7x + 12[26]$. AFFINE se code MVVQZO.

Tableau 1.4

Lettre	A	F	I	N	E
x	0	5	8	13	4
y	12	21	16	25	14
Codage	M	V	Q	Z	O

2. L'inverse de 7 modulo 26 est 15 car $7 \times 15 = 105 \equiv 1[26]$
   $y \equiv 5x + 12[26] \Rightarrow 15y \equiv x + 180[26] \Rightarrow 15y \equiv x + 24[26] \Rightarrow 15y - 24 \equiv x[26]$
   $\Rightarrow x \equiv 15y + 2[26]$.
3. La fonction de décodage est $x \equiv 15y + 2[26]$.

Tableau 1.5

Codage	C	B	M	N	J	Q	I	S	O
y	2	1	12	13	9	16	8	18	14
x	6	17	0	15	7	8	18	12	4
Lettre	G	R	A	P	H	I	S	M	E

Partie C

1. E est codé par I, donc $y = 8$ lorsque $x = 4$ de sorte que $8 \equiv 4a + b[26]$.
   V est codé par T donc $y = 19$ quand $x = 21$ et par conséquent $19 \equiv 21a + b[26]$.
2. Par soustraction membre à membre des deux congruences, on obtient $19 - 8 \equiv 21a - 4a[26]$ donc $11 \equiv 17a[26]$ ce qui peut aussi s'écrire $17a \equiv 11[26]$.
   L'inverse de 17 modulo 26 est 23 car $17 \times 23 = 391 \equiv 1[26]$ donc $23 \times 17a \equiv 23 \times 11[26] \Rightarrow a \equiv 253[26] \Rightarrow a \equiv 19[26]$.
   On remplace maintenant a par 19 dans une des congruences :
   $8 \equiv 4 \times 19 + b[26] \Rightarrow 8 - 76 \equiv b[26] \Rightarrow b \equiv -68[26] \Rightarrow b \equiv 10[26]$
3. La fonction de décodage était $y \equiv 19x + 10[26]$.

Exercices corrigés

1.1 Écrire en base 10 les nombres suivants, donnés en base 2 ou 16 : $(101010)_2$, $(1011101)_2$, $(11011010)_2$, $(135)_{16}$, $(F2)_{16}$ et $(4C)_{16}$.

1.2 Écrire en base 2 les nombres suivants : 14, 71, 238, $(B4)_{16}$, $(30A)_{16}$ et $(6D)_{16}$.

1.3 Écrire en base 16 les nombres suivants : 401, 8247, 10 000, $(10001)_2$, $(110110)_2$ et $(10010111)_2$.

1.4 Écrire en base 10 les nombres suivants : $(1{,}11)_2$, $(10{,}01)_2$, $(0{,}101)_2$, $(C{,}4)_{16}$, $(20{,}5)_{16}$ et $(8{,}1)_{16}$.

1.5 Écrire en base 2 puis en base 16 les nombres suivants : 4,75; 25,25 et 16,5.

1.6 a) On donne $a = (10110)_2$ et $b = (1101)_2$. Calculer $a + b$, $a - b$ et ab.

b) Même exercice avec $a = (10101101)_2$ et $b = (1100)_2$.

1.7 a) On donne $a = (D1)_{16}$ et $b = (19)_{16}$. Calculer $a + b$, $a - b$ et ab.

b) Même exercice avec $a = (1B4)_{16}$ et $b = (37)_{16}$.

1.8 a) On donne $a = (1011111)_2$ et $b = (100110)_2$. Calculer $a + b$, $a - b$, ab et $a{:}b$.

b) Même exercice avec $a = (1001110)_2$ et $b = (11000)_2$.

1.9 a) On donne $a = (110{,}01)_2$ et $b = (100{,}1)_2$. Calculer $a + b$, $a - b$, ab.

b) Même exercice avec $a = (1000{,}1)_2$ et $b = (1{,}11)_2$.

1.10 a) Quel est l'arrondi de $(1101011)_2$ à $(100)_2$ près ?

b) Quel est l'arrondi de $(10{,}011)_2$ à $(0{,}1)_2$ près ?

c) Quel est l'arrondi de $(A{,}BB)_{16}$ à $(0{,}1)_{16}$ près ?

1.11 Dans chaque cas, donner le quotient q et le reste r de la division euclidienne de a par b, et écrire la division euclidienne en ligne :

a) $a = 65$ et $b = 23$ b) $a = 308$ et $b = 45$ c) $a = 1\,789$ et $b = 41$

1.12 Dans chaque cas, dire si les égalités proposées correspondent à des divisions euclidiennes. Préciser alors les valeurs du quotient et du reste.

a) $37 = 8 \times 4 + 5$ b) $100 = 14 \times 7 + 2$ c) $215 = 13 \times 14 + 33$

1.13 Dans chaque cas, dire si les égalités proposées correspondent à des divisions euclidiennes. Préciser alors les valeurs du quotient et du reste.

a) $900 = 16 \times 55 + 20$ b) $662 = 31 \times 20 + 42$ c) $981 = 26 \times 37 + 19$

1.14 On écrit les unes à la suite des autres, les 26 lettres de l'alphabet. Arrivé au Z, on recommence avec un deuxième alphabet, et ainsi de suite…

a) Quelle est la 10 000^e lettre écrite ? Combien d'alphabets complets ont été écrits ?

b) Mêmes questions avec la 50 000^e lettre.

1.15 Alice possède un nombre n de CD. Si elle les empile par 10, il lui reste 7 CD. Si elle les empile par 7, elle fait 13 piles de plus que si elle les empile par 10, et il lui reste 3 CD. Combien Alice possède-t-elle de CD ?

1.16 Un texte saisi avec un logiciel comporte 5 070 lignes. L'éditeur étudie quelques possibilités de mise en pages du texte :

a) Si l'éditeur décide de mettre 64 lignes par page, combien de lignes comporte la dernière page sachant que toutes les autres sont complètes ?

b) Si l'éditeur décide de mettre 81 pages, combien de lignes comporte chaque page sachant que la dernière en comporte alors 48 ?

1.17 Faire la liste des diviseurs de 12, de 20 et de 30.

1.18 Chercher tous les couples $(a;b)$ tels que $a < b$ et a divise b, dans la liste de nombres suivante : 7, 13, 18, 28, 65, 84, 91. Présenter les résultats sous forme d'un graphique en reliant les nombres concernés par des flèches orientées comme ce modèle : $a < b$ et a divise b se code $a \to b$.

1.19 a) Montrer que la somme de trois entiers naturels consécutifs est un multiple de 3.

b) La somme de quatre entiers naturels consécutifs est-elle un multiple de 4 ?

c) La somme de cinq entiers naturels consécutifs est-elle un multiple de 5 ?

1.20 Démontrer les propriétés suivantes (voir propriété 1.3) :

a) Si a est divisible par b alors tout multiple de a est divisible par b.

b) Si a est divisible par b et si b est divisible par c, alors a est divisible par c.

c) Si a et b sont divisibles par c, alors $a + b$ et $a - b$ sont divisibles par c.

1.21 Les nombres suivants sont-ils premiers ? Si oui, dire quel est le dernier diviseur testé, si non, donner un diviseur.

107 – 161 – 179 – 241 – 311 – 323 – 437 – 563 – 677 – 779 – 971

1.22 Un début de liste prometteur... Pour tout n entier naturel, on pose : $A_n = n^2 + n + 17$.

a) Vérifier que, pour tout n de 0 à 15, A_n est un nombre premier.

b) Montrer que A_{16} n'est pas premier.

1.23 On se propose de chercher s'il existe des nombres premiers de la forme $n^2 - 4$, pour n entier naturel supérieur ou égal à 3, et si possible, d'en faire la liste. On note A_n le nombre $n^2 - 4$.

a) Tester si A_n est premier pour n valant 3, 4, 5, 6 puis 7.

b) Montrer que, à part pour $n = 3$, A_n n'est jamais premier.

1.24 On se propose de chercher s'il existe des nombres premiers de la forme $n^3 + 8$, pour n entier naturel, et d'en faire la liste. On note A_n le nombre $n^3 + 8$.

a) Montrer que si n est pair, alors A_n n'est pas premier car il est divisible par 8.

b) Développer $(n + 2)(n^2 - 2n + 4)$. En déduire que A_n n'est jamais premier.

1.25 Sur un ordinateur 16 bits, un entier N est représenté de la manière suivante : le premier bit à gauche donne le signe (0 correspond à + et 1 à –), et lorsque l'entier est positif, les 15 suivants sont les chiffres de l'écriture binaire de N. Par exemple : 0000000000011101 représente $+(11101)_2$, c'est-à-dire +29.

1) a) Que devrait logiquement représenter le nombre **1**000000000011101 ?
b) À quoi devrait être égal **0**000000000011101 + **1**000000000011101 ? Est-ce le cas ?

À cause du problème précédent on opère différemment pour coder les entiers négatifs : le premier bit représente bien – s'il est égal à 1. Cependant, la somme d'un nombre N et de son opposé $-N$ devra toujours être nulle. Pour cela on adopte la méthode dite des **compléments.**
Par exemple, pour –29, on part de la représentation de 29, c'est-à-dire **0**000000000011101 et on détermine les 16 bits **X**XXXXXXXXXXXXXXX tels que : **0**000000000011101 + **X**XXXXXXXXXXXXXXX = **0**000000000000000
Il s'agit alors de changer les 0 par des 1 et les 1 par des 0 dans la représentation de 29, puis d'ajouter 1 : cela donne **1**111111111100010+1 donc **1**111111111100011. La représentation de 29+(–29) donne alors **1**0000000000000000, donc **0**000000000000000 (par dépassement des capacités de l'ordinateur).

2) Déterminer la représentation de l'entier –123 dans un ordinateur 16 bits, puis celle de l'entier –43.

1.26 On crée un jeu simple sur un ordinateur : une bille est installée au « départ » de la grille suivante (figure 1.2 à gauche), on doit la faire parvenir à « l'arrivée » en utilisant uniquement des déplacements vers la droite et vers le bas.

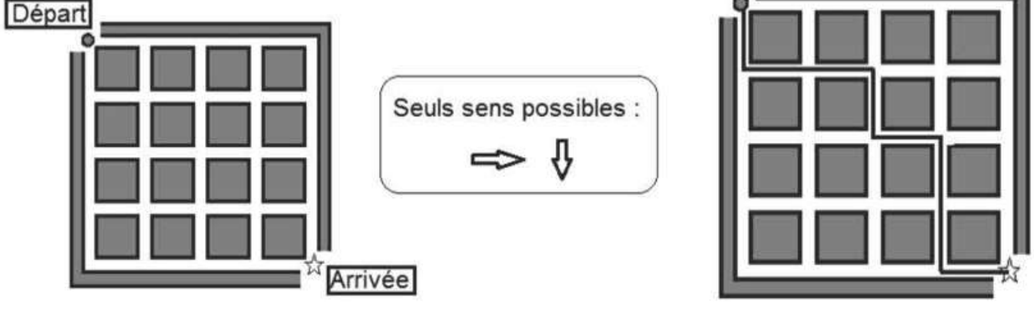


Figure 1.2

Pour programmer ce jeu, on associe à chaque parcours un nombre entier de la manière suivante :

- On note 0 chaque déplacement vers le bas et 1 chaque déplacement vers la droite.
- On obtient alors un nombre de 8 chiffres écrit en base 2.
- On écrit ensuite ce nombre en base 10.

Par exemple, pour le parcours de la figure 1.2 (à droite), on obtient le nombre entier 105. En effet, on se déplace d'abord vers le bas (0), puis deux fois vers la droite (011), puis une fois vers le bas (0110), puis une fois vers la droite (01101), puis deux fois vers le bas (0110100), et enfin une fois vers la droite (01101001). Le nombre obtenu, écrit en base 2, est $(01101001)_2$. Il est égal à 105, d'où le résultat.

1) On considère le parcours de la figure 1.3.

a) Donner le nombre en base 2 associé à ce parcours.

b) Donner l'écriture décimale (en base 10) de ce nombre.

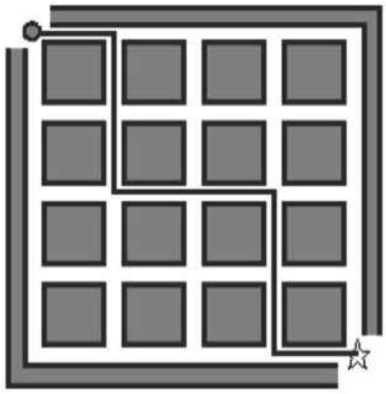


Figure 1.3

2) a) Écrire en base 2 le nombre entier 85.

b) Représenter à main levée le parcours associé au nombre 85.

3) On considère un parcours sur cette grille, et on note N le nombre entier associé, écrit en base 10.

a) Est-il possible d'avoir $N = 31$? (Justifier)

b) Quelle est la plus petite valeur possible de N ?

c) Quelle est la plus grande valeur possible de N ?

1.27 Écrire la décomposition en produit de facteurs premiers des nombres suivants : a) 80, b) 200.

1.28 Écrire la décomposition en produit de facteurs premiers des nombres suivants : a) 180, b) 630.

1.29 Écrire la décomposition en produit de facteurs premiers des nombres suivants : a) 1 150, b) 1 360.

1.30 Écrire la liste des diviseurs des nombres suivants en utilisant leurs décompositions en produit de facteurs premiers : a) 48, b) 135.

1.31 Écrire la liste des diviseurs des nombres suivants en utilisant leurs décompositions en produit de facteurs premiers : a) 1 617, b) 5 915.

1.32 a) Écrire la décomposition en produit de facteurs premiers de 350.

b) Quel est le plus petit entier naturel par lequel il faut multiplier 350 pour obtenir le carré d'un entier naturel ?

1.33 a) Écrire la décomposition en produit de facteurs premiers de 22 869.

b) Quel est le plus petit entier naturel par lequel il faut multiplier 22 869 pour obtenir le carré d'un entier naturel ?

1.34 Décomposer les nombres a et b en produits de facteurs premiers et déterminer leur PGCD dans les cas suivants :

a) $a = 96$ et $b = 728$ b) $a = 1\ 071$ et $b = 2\ 200$

1.35 Décomposer les nombres a et b en produits de facteurs premiers et déterminer leur PGCD dans les cas suivants :

a) $a = 315$ et $b = 1\ 176$ b) $a = 4\ 840$ et $b = 4\ 356$

1.36 En utilisant la règle PGCD$(a;b)$ = PGCD$(b;r)$ où r est le reste de la division euclidienne de a par b, déterminer le PGCD des nombres a et b dans les cas suivants :

a) $a = 130$ et $b = 85$ b) $a = 4\ 114$ et $b = 1\ 530$

1.37 En utilisant l'algorithme d'Euclide, chercher PGCD$(a;b)$ dans les cas suivants :

a) $a = 882$ et $b = 540$ b) $a = 1\ 725$ et $b = 1\ 309$

1.38 On dispose de 280 roses rouges et 490 roses blanches, avec lesquelles on veut faire le plus grand nombre possible de bouquets identiques. Combien peut-on faire de tels bouquets et quelle est la composition de chacun d'eux ?

1.39 Une feuille A4 a pour dimensions 21 cm et 29,7 cm. Alice cherche à savoir comment elle peut quadriller sa feuille à l'aide de carrés de mêmes dimensions, qui soient les plus gros possibles. Quelle sera la taille des carrés ? Combien en fera-t-elle ?

1.40 a) Vérifier que $90 \equiv 6[7]$ et que $66 \equiv 3[7]$.

b) En utilisant les propriétés des congruences, compléter les résultats suivants en mettant l'entier naturel le plus petit possible :

$90 + 66 \equiv ......[7]$ $4 \times 90 \equiv ......[7]$ $90 \times 66 \equiv ......[7]$

$90^2 \equiv ......[7]$ $66^3 \equiv ......[7]$

1.41 a) Faire les divisions euclidiennes de 200 et de 900 par 13 et traduire les résultats en congruences.

b) En utilisant les propriétés des congruences, compléter les résultats suivants en mettant l'entier naturel le plus petit possible :

$200 + 900 \equiv ......[13]$ $\quad$ $200 \times 900 \equiv ......[13]$

$200^2 \equiv ......[13]$ $\quad$ $900^3 \equiv ......[13]$

c) Quel est le reste de la division euclidienne de $200^4 + 900^6$ par 13 ?

1.42 a) Expliquer pourquoi tout entier naturel est congru à 0, à 1, à 2, à 3, à 4 ou à 5 modulo 6.

b) Remplir le tableau 1.6 avec des entiers naturels les plus petits possibles.

Tableau 1.6

$n \equiv$ ......[6]	0	1	2	3	4	5
$n + 1 \equiv$ ......[6]						
$n + 2 \equiv$ ......[6]						
$n(n + 1)(n + 2) \equiv$ ......[6]						

c) Que peut-on conclure au sujet du produit de trois entiers naturels consécutifs ?

1.43 a) Remplir le tableau 1.7, avec des entiers naturels les plus petits possibles.

Tableau 1.7

$n \equiv$ ......[5]	0	1	2	3	4
$n^2 \equiv$ ......[5]					
$n^3 \equiv$ ......[5]					
$n^5 \equiv$ ......[5]					
$4n \equiv$ ......[5]					
$n^5 + 4n \equiv$ ......[5]					

b) Interpréter la dernière ligne du tableau à l'aide d'une phrase concernant le reste d'une division euclidienne de $n^5 + 4n$.

1.44 a) Remplir le tableau 1.8, avec des entiers naturels les plus petits possibles.

Tableau 1.8

$n \equiv$ ......[7]	0	1	2	3	4	5	6
$n^3 \equiv$ ......[7]							

b) Montrer que 1 789 $\equiv$ 4[7]. Compléter la congruence suivante : 1 $789^3 \equiv$ ......[7].

c) Montrer que 1 $789^{1789} \equiv (1\,789^3)^{596} \times 1\,789$.

d) En déduire que 1 $789^{1789} \equiv 4[7]$. Exprimer ce résultat avec une phrase concernant un reste de division euclidienne.

1.45 Soit a, b, c d, n, p des entiers naturels tels que $a \equiv b[n]$ et $c \equiv d[n]$. Démontrer les propriétés suivantes :

a) $a + c \equiv b + d[n]$ b) $pa \equiv pb[n]$ c) $ac \equiv bd[n]$

1.46 Soit E = {1;2;3;4;5;6}.

a) Vérifier que pour chaque élément a de E, il existe un élément b de E tel que $ab \equiv 1[7]$. On dit que b est l'inverse de a modulo 7.

b) Résoudre l'équation $3x \equiv 4[7]$. Donner l'ensemble des solutions comprises entre 100 et 140.

1.47 Soit E l'ensemble des entiers naturels non nuls inférieurs ou égaux à 10.

a) Déterminer l'inverse modulo 11 de tous les éléments de E.

b) Résoudre l'équation $4x + 7 \equiv 8[11]$.

1.48 Pour des questions de saisie, les billets en euros sont tous numérotés. Le numéro est constitué d'une lettre suivie de 11 chiffres.

On commence par remplacer la lettre par son rang dans l'alphabet (A = 1, B = 2, C = 3, D = 4, … , Z = 26) puis on additionne ce nombre avec les 11 autres chiffres du billet. Le reste de la division par **9** de ce résultat doit être **8** (sinon, le billet est faux…).

Par exemple, vérifions un billet numéroté z10708476264. On remplace la lettre z par 26 et on additionne 26 + 1 + 0 + 7 + 0 + 8 + 4 + 7 + 6 + 2 + 6 + 4 = 71. Le reste de la division de 71 par 9 est bien **8**.

a) Des billets numérotés v02387040334 et m12065974438 sont-ils des vrais ?

b) Sur un billet authentique figure le numéro s0216644810[redacted] mais le dernier chiffre est illisible. Quel est ce chiffre ?

c) Sur un autre billet authentique, la lettre est illisible [redacted]16122340242. Quelles sont les lettres possibles ?

d) Ce moyen de contrôle n'est pas très poussé. En effet, pour un billet, un opérateur peut faire une faute de saisie sur un chiffre sans que celle-ci ne soit détectée ! Donner un exemple d'erreur de saisie non détectée sur un billet numéroté x30564853796.

1.49 Le code-barres rencontré au supermarché est le code EAN-13 formé de 13 chiffres. Le dernier chiffre est une clé de contrôle calculée à partir des 12 autres.

Pour calculer cette clé de contrôle :

- On multiplie par 3 chaque chiffre de rang pair, on ajoute tous les résultats et on ajoute tous les chiffres de rang impair.

- On calcule le reste dans la division par 10 de la somme obtenue.
- Si le reste est 0, la clé est 0 sinon on retranche le reste de 10 pour avoir la clé.

Par exemple, prenons le code-barres 471-9-5120-0288-*x* (où *x* est la clé de contrôle que l'on cherche).

Tableau 1.9

Chiffre	4	7	1	9	5	1	2	0	0	2	8	8
Coefficient multiplicateur	1	3	1	3	1	3	1	3	1	3	1	3
Résultat	4	21	1	27	5	3	2	0	0	6	8	24

La somme vaut 4 + 21 + 1 + 27 + 5 + 3 + 2 + 0 + 0 + 6 + 8 + 24 = 101, le reste dans la division par 10 est 1 donc la clé est 10 – 1 = 9.

a) Quelle est la clé de contrôle des codes-barres suivants, 4 971850 13816 x et 6 547531 18367 *x* ?

b) Peut-on retrouver le chiffre manquant dans les codes-barres suivants, 9 782216 1■247 4 et 3 2821■2 62913 4 ?

Solutions

1.1 $(101010)_2 = 1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2 + 0 = 42$

$$(1011101)_2 = 1\times 2^6 + 0\times 2^5 + 1\times 2^4 + 1\times 2^3 + 1\times 2^2 + 0\times 2 + 1 = 93$$

$$(11011010)_2 = 1\times 2^7 + 1\times 2^6 + 0\times 2^5 + 1\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2 + 0 = 218$$

$$(135)_{16} = 1\times 16^2 + 3\times 16 + 5 = 309$$

$$(F2)_{16} = 15\times 16 + 2 = 242$$

$$(4C)_{16} = 4\times 16 + 12 = 76$$

1.2 $14 = 8 + 4 + 2 = 1\times 2^3 + 1\times 2^2 + 1\times 2 + 0 = (1110)_2$

$$\begin{aligned} 71 &= 64 + 4 + 2 + 1 \\ &= 1\times 2^6 + 0\times 2^5 + 0\times 2^4 + 0\times 2^3 + 1\times 2^2 + 1\times 2 + 1 \\ &= (1000111)_2 \end{aligned}$$

$$\begin{aligned} 238 &= 128 + 64 + 32 + 8 + 4 + 2 \\ &= 1\times 2^7 + 1\times 2^6 + 1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 1\times 2^2 + 1\times 2 + 0 \\ &= (11101110)_2 \end{aligned}$$

Pour les conversions de base 16 vers la base 2, il suffit de convertir chaque chiffre en base 2 et de concaténer (au besoin, se servir du tableau de conversion donné dans la partie cours)

$(B4)_{16} = (10110100)_2 \qquad (30A)_{16} = (1100001010)_2 \qquad (6D)_{16} = (1101101)_2$

1.3 $401 = 1 \times 16^2 + 9 \times 16 + 1 = (191)_{16}$

$8247 = 2 \times 16^3 + 0 \times 16^2 + 3 \times 16 + 7 = (2037)_{16}$

$10000 = 2 \times 16^3 + 7 \times 16^2 + 1 \times 16 + 0 = (2710)_{16}$

Pour les conversions de base 2 vers la base 16, on regroupe les chiffres par paquets de 4 en partant de la droite et on utilise la table de conversion.

$(10001)_2 = (00010001)_2 = (11)_{16}$

$(110110)_2 = (00110110)_2 = (36)_{16}$

$(10010111)_2 = (97)_{16}$

1.4

$(1{,}11)_2 = 1 + \dfrac{1}{2} + \dfrac{1}{2^2} = 1{,}75 \qquad (10{,}01)_2 = 1 \times 2 + 0 \times 1 + \dfrac{0}{2} + \dfrac{1}{2^2} = 2{,}25$

$(0{,}101)_2 = 0 + \dfrac{1}{2} + \dfrac{0}{2^2} + \dfrac{1}{2^3} = 0{,}625$

$(C{,}4)_{16} = 12 \times 16 + \dfrac{4}{16} = 192{,}25 \qquad (20{,}5)_{16} = 2 \times 16 + 0 + \dfrac{5}{16} = 32{,}3125$

$(8{,}1)_{16} = 8 \times 16 + \dfrac{1}{16} = 128{,}0625$

1.5 En base 2 :

$4{,}75 = 4 + 0{,}75 = 1 \times 2^2 + \dfrac{1}{2} + \dfrac{1}{2^2} = (100{,}11)_2$

$25{,}25 = 1 \times 2^4 + 1 \times 2^3 + 1 + \dfrac{1}{2^2} = (11001{,}01)_2$

$16{,}5 = 1 \times 2^4 + \dfrac{1}{2} = (10000{,}1)_2$

En base 16 :

$4{,}75 = 4 + \dfrac{3}{4} = 4 + \dfrac{12}{16} = (4{,}C)_{16}$

$$25,25 = 1\times 16 + 9 + \frac{1}{4} = 1\times 16 + 9 + \frac{4}{16} = (19,4)_{16}$$

$$16,5 = 1\times 16 + \frac{8}{16} = (10,8)_{16}$$

1.6 a) $a + b = (100011)_2$ $a–b = (1001)_2$ $ab = (100011110)_2$

b) $a + b = (10111001)_2$ $a–b = (10100001)_2$ $ab = (100000011100)_2$

1.7 a) $a + b = (EA)_{16}$ $a–b = (B8)_{16}$ $ab = (1469)_{16}$

b) $a + b = (1EB)_{16}$ $a–b = (17D)_{16}$ $ab = (5DAC)_{16}$

1.8 a) $a + b = (10000101)_2$ $a–b = (111001)_2$ $ab = (111000011010)_2$
$a : b = (10,1)_2$

b) $a + b = (1100110)_2$ $a–b = (110110)_2$ $ab = (11101010000)_2$
$a:b = (11,01)_2$

1.9 a) $a + b = (1010,11)_2$ $a–b = (1,11)_2$ $ab = (11100,001)_2$

b) $a+b = (1010,01)_2$ $a–b = (110,11)_2$ $ab = (1110,111)_2$

1.10 a) $(1101000)_2$ b) $(10,1)_2$ c) $(A,C)_{16}$

1.11 a) $q = 2$ et $r = 19$ $65 = 23\times 2 + 19$

b) $q = 6$ et $r = 38$ $308 = 45\times 6 + 38$

c) $q = 43$ et $r = 26$ $1789 = 41\times 43 + 26$

1.12 a) C'est la division euclidienne de 37 par 8. Le quotient est 4 et le reste est 5.

b) Cette égalité correspond à deux divisions euclidiennes : à celle de **100 par 14**, avec 7 pour quotient et 2 pour reste, et à celle de **100 par 7**, avec 14 pour quotient et 2 pour reste.

c) Cette égalité ne correspond à aucune division euclidienne car le reste potentiel serait 33, et il est supérieur aux diviseurs possibles qui sont 13 et 14.

1.13 a) C'est la division euclidienne de 900 par 55. Le quotient est 16 et le reste est 20.

b) Cette égalité ne correspond à aucune division euclidienne car le reste potentiel serait 42, et il est supérieur aux diviseurs possibles qui sont 31 et 20.

c) Cette égalité correspond à deux divisions euclidiennes : à celle de 981 par 26, avec 37 pour quotient et 19 pour reste, et à celle de 981 par 26, avec 37 pour quotient et 19 pour reste.

1.14 a) 10 000 = 26 × 384 + 16. On a donc écrit 384 alphabets complets et la 10 000^{e} lettre est la lettre de rang 16, c'est-à-dire un P.

b) 50 000 = 26 × 1 923 + 2. On a donc écrit 1 923 alphabets complets et la 50 000^e^ lettre est la lettre de rang 2, c'est-à-dire un B.

1.15 Si on appelle q le nombre de piles de 10, alors $n = 10q + 7$. Si elle fait des piles de 7, elle aura $q + 13$ piles donc $n = 7(q + 13) + 3 = 7q + 94$.
Donc $10q + 7 = 7q + 94$ d'où $3q = 87$ et $q = 29$.

Alice possède donc n = 10 × 29 + 7 = 297 CD.

1.16 Notons b le nombre de lignes par page, q le nombre de pages et r le nombre de lignes de la dernière page. On a l'égalité $5\,070 = bq + r$.

a) Dans ce cas, $b = 64$, et la division euclidienne de 5 070 par 64 donne $q = 79$ et $r = 14$. La dernière page aura donc 14 lignes.

b) Dans ce cas, $q = 81$ et $r = 48$ donc $b = (5\,070 - 48):81 = 62$. Chaque page aura donc 62 lignes.

1.17 Les diviseurs de 12 sont 1, 2, 3, 4, 6 et 12, ceux de 20 sont 1, 2, 4, 5, 10 et 20 et ceux de 30 sont 1, 2, 3, 5, 6, 10, 15 et 30.

1.18

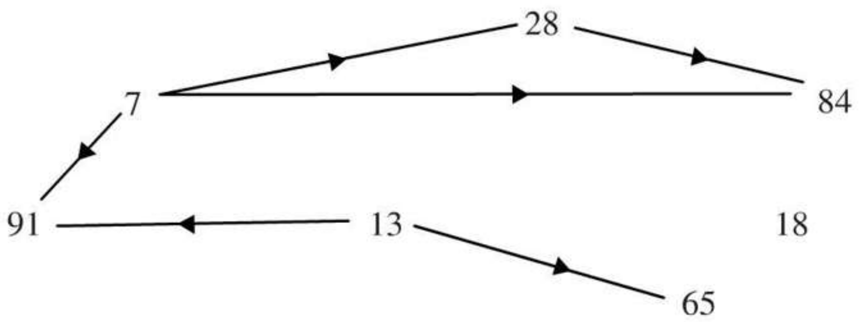


Figure 1.4

1.19 a) Trois entiers naturels consécutifs peuvent s'écrire n, $n + 1$ et $n + 2$. Leur somme vaut $3n + 6 = 3\,(n + 2)$ et c'est un multiple de 3 puisque $n + 2$ est un entier naturel.

b) C'est faux, il suffit de prendre un contre exemple : $1 + 2 + 3 + 4 = 10$ et 10 n'est pas multiple de 4.

c) Oui. En effet, cinq entiers naturels consécutifs sont n, $n + 1$, $n + 2$, $n + 3$ et $n + 4$ dont la somme est $5n + 10 = 5(n + 2)$ qui est un multiple de 5 car $n + 2$ est un entier naturel.

1.20 On rappelle d'abord la définition : « a est divisible par b s'il existe un entier naturel k tel que $a = kb$ ».

a) a est divisible par b donc il existe un entier naturel k tel que $a = bk$. Un multiple de a s'écrit na avec n entier naturel. Donc $na = nbk = b(nk)$ ce qui prouve que na est divisible par b puisque nk est un entier.

b) a est divisible par b donc il existe un entier naturel k tel que $a = kb$. b est divisible par c donc il existe un entier naturel k' tel que $b = ck'$. Alors $a = kb = ck'k$ ce qui prouve que a est divisible par c puisque kk' est un entier.

c) a est divisible par c donc il existe un entier naturel k tel que $a = ck$. b est divisible par c donc il existe un entier naturel k' tel que $b = ck'$. Alors $a + b = ck + ck'$ $= c(k + k')$ ce qui prouve que $a + b$ est divisible par c puisque $k + k'$ est un entier.

La démonstration pour a–b est calquée sur celle de $a + b$...

1.21 $\sqrt{107} \approx 10{,}3$. 107 n'est divisible ni par 2, ni par 3, ni par 5, ni par 7 (qui sont les nombres premiers inférieurs ou égaux à $\sqrt{107}$) donc 107 est premier.

161 n'est pas premier car il est divisible par 7.

179 est premier, il faut tester jusqu'à 13.

241 est premier, il faut tester jusqu'à 13.

311 est premier, on teste jusqu'à 17.

323 n'est pas premier, il est divisible par 17.

437 n'est pas premier, il est divisible par 19.

563 est premier, on teste jusqu'à 23.

677 est premier, on teste jusqu'à 23.

779 n'est pas premier, il est divisible par 19.

971 est premier, on teste jusqu'à 31.

1.22 a) En faisant varier n de 0 à 15, on obtient les nombres 17, 19, 23, 29, 37, 47, 59, 73, 89, 107, 127, 149, 173, 199, 227 et 257 qui sont tous premiers (les vérifications sont faciles).

b) $A_{16} = 16^2 + 16 + 17 = 16(16+1) + 17 = 16 \times 17 + 17 = 17(16 + 1) = 17^2$ donc A_{16} n'est pas premier.

1.23 a) A_n est premier quand $n = 3$. Il n'est pas premier quand n vaut 4, 5, 6 ou 7.

b) $A_n = (n + 2)(n - 2)$. A_n apparaît comme un produit de deux facteurs donc il n'est jamais premier sauf si un des deux facteurs vaut 1. C'est le cas lorsque $n - 2 = 1$ c'est-à-dire pour $n = 3$.

Finalement, A_n est premier seulement lorsque $n = 3$.

1.24 a) Si n est pair, on peut poser $n = 2m$, avec m entier naturel.

Alors, $A_n = (2m)^3 + 8 = 8m^3 + 8 = 8(m^3 + 1)$ qui est divisible par 8.

b) $(n + 2)(n^2 - 2n + 4) = n^3 - 2n^2 + 4n + 2n^2 - 4n + 8 = n^3 + 8$

On en déduit que $A_n = (n + 2)(n^2 - 2n + 4)$. C'est un produit de deux facteurs, aucun des deux ne pouvant être égal à 1, donc A_n n'est jamais premier.

1.25 1) a) Logiquement, cela devrait être le nombre $-(11101)_2$, c'est-à-dire -29.

b) La somme devrait être égale à 0. Or :
$(0000000000011101)_2 + (1000000000011101)_2 = (1000000000111010)_2 = +32\,826$

2) $123 = (1111011)_2$ donc la représentation de 123 sur 16 bits est 0000000001111011. Celle de -123 est donc :
1111111110000100+1 = 1111111110000101

$43 = (101011)_2$ donc sa représentation est 0000000000101011. Celle de -43 est donc 1111111111010100+1 = 1111111111010101.

1.26 1) a) Le nombre associé à ce parcours est $(10011001)_2$.

b) Son écriture décimale est $2^7 + 2^4 + 2^3 + 1 = 153$.

2) a) $85 = 2^6 + 2^4 + 2^2 + 1 = (01010101)_2$

b) Le parcours est BDBDBDBD (B = bas, D = droite).

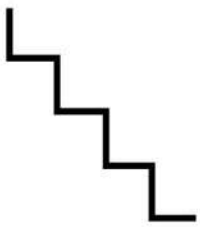


Figure 1.5

3) a) $31 = (11111)_2 = (00011111)_2$. Le nombre 31 correspond à un trajet comportant cinq déplacements vers le bas. Or tout trajet se décompose en quatre déplacements vers le bas et autant vers la droite. Il est donc impossible d'avoir N = 31.

b) La plus petite valeur de N est $(00001111)_2 = 15$.

c) La plus grande valeur de N est $(11110000)_2 = 240$.

1.27 a) $80 = 2^4 \times 5$ b) $200 = 2^3 \times 5^2$

1.28 a) $180 = 2^2 \times 3^2 \times 5$ b) $630 = 2 \times 3^2 \times 5 \times 7$

1.29 a) $1\,150 = 2 \times 5^2 \times 23$ b) $1\,360 = 2^4 \times 5 \times 17$

1.30 a) $48 = 2^4 \times 3$. Les diviseurs de 48 sont de la forme $2^i \times 3^j$ avec i qui vaut 0, 1, 2, 3 ou 4 et j qui vaut 0 ou 1. Ce sont donc :

$2^0 \times 3^0 = 1$ $2^1 \times 3^0 = 2$ $2^2 \times 3^0 = 4$ $2^3 \times 3^0 = 8$ $2^4 \times 3^0 = 16$

$2^0 \times 3^1 = 3$ $2^1 \times 3^1 = 6$ $2^2 \times 3^1 = 12$ $2^3 \times 3^1 = 24$ $2^4 \times 3^1 = 48$

Classés par ordre croissant, les diviseurs de 48 sont : 1, 2, 3, 4, 6, 8, 12, 16, 24 et 48.

b) $135 = 3^3 \times 5$. Les diviseurs de 135 sont de la forme $3^i \times 5^j$ avec i qui vaut 0, 1, 2 ou 3 et j qui vaut 0 ou 1. Classés par ordre croissant, les diviseurs de 135 sont : 1, 3, 5, 9, 15, 27, 45 et 135.

1.31 a) $1\,617 = 3 \times 7^2 \times 11$. Les diviseurs de 1617 sont 1, 3, 7, 11, 21, 33, 49, 77, 147, 231, 539 et 1617.

b) $5\,915 = 5 \times 7 \times 13^2$. Les diviseurs de 5915 sont 1, 5, 7, 13, 35, 65, 91, 169, 455, 845, 1183 et 5915.

1.32 a) $350 = 2 \times 5^2 \times 7$

b) La décomposition en produit de facteurs premiers d'un carré ne doit comporter que des exposants pairs. Cela deviendra le cas si on multiplie 350 par 2×7. La décomposition sera alors $2^2 \times 5^2 \times 7^2 = (2 \times 5 \times 7)^2 = 70^2$.

1.33 a) $22\,869 = 3^3 \times 7 \times 11^2$

b) Il faut multiplier par 3×7 pour obtenir $3^4 \times 7^2 \times 11^2 = (3^2 \times 7 \times 11)^2$ qui est le carré de 693.

1.34 a) $a = 2^5 \times 3$ $\quad b = 2^3 \times 7 \times 13$ $\quad \text{PGCD}(a;b) = 2^3 = 8$

b) $a = 3^2 \times 7 \times 17$ $\quad b = 2^3 \times 5^2 \times 11$

$\text{PGCD}(a;b) = 1$ ce qui prouve que 1 071 et 2 200 sont premiers entre eux.

1.35 a) $a = 315 = 3^2 \times 5 \times 7$ $\quad b = 1\,176 = 2^3 \times 3 \times 7^2$

$\text{PGCD}(a;b) = 3 \times 7 = 21$

b) $a = 4\,840 = 2^3 \times 5 \times 11^2$ $\quad b = 4\,356 = 2^2 \times 3^2 \times 11^2$

$\text{PGCD}(a;b) = 2^2 \times 11^2 = 484$

1.36 a) Cette méthode est l'algorithme d'Euclide. On va effectuer des divisions euclidiennes jusqu'à obtenir un reste nul. Le PGCD sera alors le dernier diviseur testé.

$130 = 85 \times 1 + 45 \quad 85 = 45 \times 1 + 35 \quad 45 = 35 \times 1 + 10$

$35 = 10 \times 3 + 5 \quad 10 = 5 \times 2 + 0$

Le reste est 0, le dernier diviseur testé est 5 donc PGCD(130;85) = 5.

b) $4\,114 = 1\,530 \times 2 + 1\,054$

$1530 = 1054 \times 1 + 476 \quad 1054 = 476 \times 2 + 102 \quad 476 = 102 \times 4 + 68$

$102 = 68 \times 1 + 34 \quad 68 = 34 \times 2 + 0$

Le reste est 0, le dernier diviseur testé est 34 donc PGCD(4 114;1 530) = 34.

1.37 a) PGCD(882;540) = 18 $\quad$ b) PGCD(1 725;1 309) = 1

1.38 Le nombre de bouquets dit être le PGCD de 280 et de 490.

$280 = 2^3 \times 5 \times 7$ et $490 = 2 \times 5 \times 7^2$ donc PGCD(280;490) $= 2 \times 5 \times 7 = 70$.

On peut donc faire au plus 70 bouquets. Chacun d'eux aura alors 4 roses rouges et 7 roses blanches.

1.39 Le côté de chaque carré doit être un diviseur des dimensions en millimètres de la feuille et pour que les carrés soient les plus gros possibles, il faut que ce soit le PGCD de 210 et 297 (dimensions en mm).

$210 = 2 \times 3 \times 5 \times 7$ et $297 = 3^3 \times 11$ et donc PGCD(210;297) = 3.

Les carrés auront 3 mm de côté. Sur la largeur, on fera 210:3 = 70 carrés, sur la longueur, on en fera 297:3 = 99, ce qui fera $70 \times 99 = 6\,930$ carrés.

1.40 a) 6 est le reste de la division euclidienne de 90 par 7, et 3 est le reste de la division euclidienne de 66 par 7.

b) $90+66 \equiv 6+3[7] \equiv 2[7]$ $\quad 4\times 90 \equiv 4\times 6[7] \equiv 24[7] \equiv 3[7]$

$90\times 66 \equiv 6\times 3[7] \equiv 18[7] \equiv 4[7]$ $\quad 90^2 \equiv 6^2[7] \equiv 36[7] \equiv 1[7]$

$66^3 \equiv 3^3[7] \equiv 27[7] \equiv 6[7]$

1.41 a) $200 = 13\times 15+5$ donc $200 \equiv 5[13]$

$900 = 13\times 69+3$ donc $900 \equiv 3[13]$

b) $200+900 \equiv 5+3[13] \equiv 8[13]$ $\quad 200\times 900 \equiv 5\times 3[13] \equiv 2[13]$

$200^2 \equiv 5^2[13] \equiv 12[13]$ $\quad 900^3 \equiv 3^3[13] \equiv 1[13]$

c) $200^4 = \left(200^2\right)^2$ donc $200^4 \equiv 12^2[13] \equiv 144[13] \equiv 1[13]$

$900^6 = \left(900^3\right)^2$ donc $900^6 \equiv 1^2[13] \equiv 1[13]$

$200^4 + 900^6 \equiv 1+1[13] \equiv 2[13]$, le reste de la division euclidienne de $200^4 + 900^6$ par 13 est 2.

1.42 a) Dans la division euclidienne par 6, le reste est 0, 1, 2, 3, 4 ou 5. Donc tout entier naturel est congru à un de ces 6 nombres.

b)

Tableau 1.10

$n \equiv$ ......[6]	0	1	2	3	4	5
$n + 1 \equiv$ ......[6]	1	2	3	4	5	0
$n + 2 \equiv$ ......[6]	2	3	4	5	0	1
$n(n + 1)(n + 2) \equiv$ ......[6]	0	0	0	0	0	0

c) Le produit de trois entiers naturels consécutifs est toujours un multiple de 6.

1.43 a)

Tableau 1.11

$n \equiv$ ......[5]	0	1	2	3	4
$n^2 \equiv$ ......[5]	0	1	4	4	1
$n^3 \equiv$ ......[5]	0	1	3	2	4
$n^5 \equiv$ ......[5]	0	1	2	3	4
$4n \equiv$ ......[5]	0	4	3	2	1
$n^5 + 4n \equiv$ ......[5]	0	0	0	0	0

b) Pour tout entier naturel n, le reste de la division euclidienne de $n^5 + 4n$ par 5 est 0. Autrement dit, $n^5 + 4n$ est toujours un multiple de 5.

1.44 a)

Tableau 1.12

$n \equiv$ ......[7]	0	1	2	3	4	5	6
$n^3 \equiv$ ......[7]	0	1	1	6	1	6	6

b) 4 est le reste de la division euclidienne de 1 789 par 7 donc $1\,789 \equiv 4[7]$. Donc $1\,789^3 \equiv 4^3[7] \equiv 1[7]$.

c) $1\,789 = 3 \times 596 + 1$ donc $1\,789^{1789} \equiv 1\,789^{3\times 596+1} = (1\,789^3)^{596} \times 1\,789$.

d) $1\,789^{1789} \equiv (1\,789^3)^{596} \times 1\,789[7] \equiv 1^{596} \times 1\,789[7] \equiv 1\,789[7] \equiv 4[7]$.

Le reste de la division euclidienne de $1\,789^{1789}$ par 7 est 4.

1.45 $a \equiv b[n]$ donc il existe un entier naturel k tel que $a–b = kn$.

$c \equiv d[n]$ donc il existe un entier naturel k' tel que $c–d = k'n$.

a) $a - b + c - d = kn + k'n \Leftrightarrow (a+c) - (b+d) = (k+k')n$. Puisque $(k + k')n$ est un multiple de n, cela prouve que $a + c \equiv b + d[n]$.

b) $a - b = kn \Rightarrow p(a - b) = pkn \Rightarrow pa - pb = (pk)n$. Puisque $(pk)n$ est un multiple de n, on a prouvé que $pa \equiv pb[n]$.

c) $a - b = kn \Leftrightarrow a = b + kn$ et $c - d = k'n \Leftrightarrow c = d + k'n$.

Alors $ac = (b+kn)(d+k'n) = bd + bk'n + dkn + kk'n^2 = bd + (bk' + dk + kk'n)n$.

Donc $ac - bd = (bk' + dk + kk'n)n$, qui est un multiple de n, donc on a prouvé que $ac \equiv bd[n]$.

1.46 a) 1 est son propre inverse.

$2 \times 4 = 8 \equiv 1[7]$ donc 2 et 4 sont inverses l'un de l'autre.

$3 \times 5 = 15 \equiv 1[7]$ donc 3 et 5 sont inverses l'un de l'autre.

$6 \times 6 = 36 \equiv 1[7]$ donc 6 est son propre inverse.

b) $3x \equiv 4[7] \Leftrightarrow 5 \times 3x \equiv 5 \times 4[7] \Leftrightarrow x \equiv 6[7]$

Les solutions de l'équation sont tous les entiers congrus à 6 modulo 7.

Les solutions comprises entre 100 et 140 sont : 104, 111, 118, 125, 132, 139.

1.47 a)

Tableau 1.13

Nombre	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Inverse modulo 11	1	6	4	3	9	2	8	7	5	10

b) $4x + 7 \equiv 8[11] \Leftrightarrow 3(4x + 7) \equiv 3 \times 8[11] \Leftrightarrow 12x + 21 \equiv 24[11] \Leftrightarrow x \equiv 3[11]$

Les solutions de l'équation sont tous les entiers congrus à 3 modulo 11

1.48 a) $v = 22$, on additionne $22 + 0 + 2 + 3 + 8 + 7 + 0 + 4 + 0 + 3 + 3 + 4 = 56$. Dans la division par 9, le reste est 2 donc ce billet est un faux.

$m = 13$, on additionne $13 + 1 + 2 + 0 + 6 + 5 + 9 + 7 + 4 + 4 + 3 + 8 = 62$. Dans la division par 9, le reste est 8 donc le billet est un vrai.

b) $s = 19$. Si on note x le dernier chiffre, la somme est $51 + x$. Le reste dans la division par 9 est 8 lorsque $x = 2$ (c'est la seule possibilité avec $0 \leq x \leq 9$).

c) Notons α la valeur lettre manquante, $0 \leq \alpha \leq 26$. La somme est $\alpha + 27$, elle doit être congrue à 8 modulos 9. Donc comme $27 \equiv 0[9]$, alors on doit avoir $\alpha \equiv 8[9]$ ce qui donne comme possibilités $\alpha = 8$, $\alpha = 17$, $\alpha = 26$ c'est-à-dire les lettres i, q et z.

d) Taper un 9 à la place d'un 0 ne changerait pas le reste de la division euclidienne, c'est donc une erreur non détectable.

1.49 a) Pour le premier code :

Tableau 1.14

Chiffre	**4**	**9**	**7**	**1**	**8**	**5**	**0**	**1**	**3**	**8**	**1**	**6**
Coefficient multiplicateur	1	3	1	3	1	3	1	3	1	3	1	3
Résultat	4	27	7	3	8	15	0	3	3	24	1	18

La somme totale est 113. Le reste dans la division par 10 est 3 donc la clé est $10 - 3 = 7$.

Pour le deuxième code :

Tableau 1.15

Chiffre	6	5	4	7	5	3	1	1	8	3	6	7
Coefficient multiplicateur	1	3	1	3	1	3	1	3	1	3	1	3
Résultat	6	15	4	21	5	9	1	3	8	9	6	21

La somme totale est 108. Le reste dans la division par 10 est 8 donc la clé est : $10 - 8 = 2$

b) Pour le premier code : soit x le chiffre inconnu, la somme est :
$9 + 21 + 8 + 6 + 2 + 3 + 6 + 3x + 6 + 4 + 21 = 89 + x$
La clé est 4 donc le reste de la division par 10 doit être 6. C'est possible uniquement avec $x = 7$.

Pour le deuxième code : soit x le chiffre inconnu, la somme est :
$3 + 6 + 8 + 6 + 1 + 3x + 2 + 18 + 2 + 27 + 1 + 9 = 83 + 3x$
La clé est 4 donc le reste dans la division par 10 doit être 6. C'est possible uniquement si $x = 1$.

SUITES NUMÉRIQUES 2

PLAN

2.1 Généralités
2.2 Suites particulières
2.3 Variations d'une suite
2.4 Limite d'une suite

OBJECTIFS

- ➤ Étude de phénomènes discrets
- ➤ Comparaison de modèles mathématiques et étude de la tendance à long terme
- ➤ Étude de situations relevant de suites arithmétiques ou géométriques

2.1 GÉNÉRALITÉS

Une liste de nombres réels du type $(u_0\,;u_1\,;u_2\,;u_3\,;\ldots)$ ou $(u_1\,;u_2\,;u_3\,;\ldots)$ s'appelle une suite numérique et peut se noter plus simplement (u_n). Les éléments de cette liste sont appelés les termes de la suite. Lorsque u_n est définie par une formule du type $u_n = f(n)$, on dit que $f(n)$ est le terme général de la suite.

> ***Exemple***
>
> La liste (1 ; 3 ; 5 ; 7 ; 9 ; 11 ; 13 ; ...) des entiers naturels impairs peut être générée en posant, pour tout entier $n \geq 0$, $u_n = 2n + 1$:
> $u_0 = 2 \times 0 + 1 = 1$; $u_1 = 2 \times 1 + 1 = 3$; $u_2 = 2 \times 2 + 1 = 5$; etc.
> Il s'agit donc de la suite (u_n) de terme général $2n + 1$.

Lorsqu'une suite a été définie à l'aide de son terme général, on parle de **suite définie explicitement**.

> ***Exemple***
>
> La suite (v_n) définie pour tout entier $n \geq 1$ par $v_n = 2/n$ admet pour terme général $2/n$.
> Ses termes sont $v_1 = 2/1 = 2$; $v_2 = 2/2 = 1$; $v_3 = 2/3$, etc.

On peut aussi générer une suite sans connaître son terme général, à l'aide de son premier terme et d'une relation entre des termes consécutifs comme dans l'exemple suivant. On parle alors de **suite définie par récurrence**.

> ***Exemple***
> Soit (u_n) la suite définie par $u_0 = 2$ et $u_{n+1} = 2u_n + 1$ (pour tout $n \geq 0$). Les termes de cette suite sont $u_0 = 2$; $u_1 = 2u_0 + 1 = 2 \times 2 + 1 = 5$; $u_2 = 2u_1 + 1 = 2 \times 5 + 1 = 11$; $u_3 = 2u_2 + 1 = 2 \times 11 + 1 = 23$; etc.
> On a utilisé un tableur pour obtenir les premiers termes de cette suite. Dans la cellule B2, on a saisi 2, puis « = 2*B2+1 » dans B3 avant de l'étirer vers le bas.
> On a représenté ci-dessous en sélectionnant la plage A2:B12, les onze premiers termes (de u_0 à u_{10}).

	A	B
1	n	Un
2	0	2
3	1	5
4	2	11
5	3	23
6	4	47
7	5	95
8	6	191
9	7	383
10	8	767
11	9	1535
12	10	3071

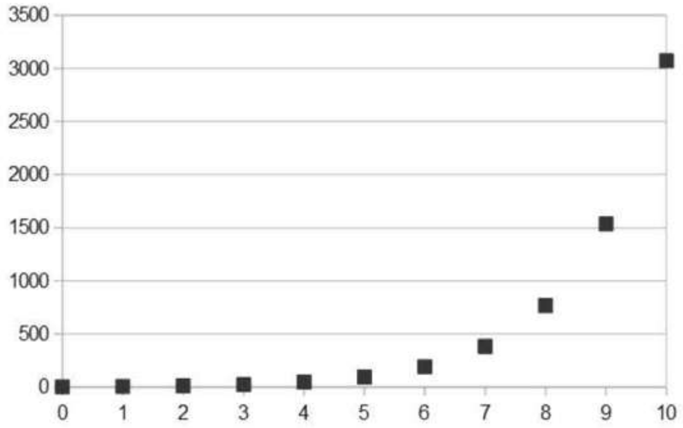


Figure 2.1

Le menu « suites » des calculatrices graphiques permet d'obtenir, sous la forme d'une table de valeurs, les premiers termes d'une suite (définie par récurrence ou explicitement).

> ***Exemple***
> Pour une suite récurrente comme celle de l'exemple précédent, on peut aussi utiliser le menu « calcul » en saisissant la séquence d'instructions suivante :
> 2, **EXE**, ANS×2+1, **EXE**, **EXE**, **EXE**, etc.

2.2 Suites particulières

2.2.1 Suites arithmétiques

On dit qu'une suite (u_n) est **arithmétique** lorsqu'on passe d'un terme de cette suite au suivant en ajoutant toujours le même nombre, c'est-à-dire s'il existe un réel r tel que pour tout entier n, $u_{n+1} = u_n + r$. Le nombre r est alors appelé la **raison** de la suite (u_n).

Exemple

Soit (u_n) la suite définie par $u_0 = 3$ et $u_{n+1} = u_n + 2$. On passe d'un terme de la suite (u_n) au suivant en ajoutant 2 : $u_1 = u_0 + 2 = 3 + 2 = 5$; $u_2 = u_1 + 2 = 5 + 2 = 7$; $u_3 = u_1 + 2 = 7 + 2 = 9$; etc. La suite (u_n) est arithmétique de raison 2 et de premier terme 3.

Propriété 2.1

Si (u_n) est une suite arithmétique de raison r, alors :
– Pour tout entier $n \geq 0$, $u_n = u_0 + nr$.
– Pour tout entier $n \geq 1$, $u_n = u_1 + (n-1)r$.
– Pour tout entier $n \geq 2$, $u_n = u_2 + (n-2)r$.
– Etc.

Exemples

– Soit (u_n) la suite numérique définie par $u_0 = 5$ et $u_{n+1} = u_n + n$.
On a $u_1 = u_0 + 0 = 5 + 0 = 5$ et $u_2 = u_1 + 1 = 5 + 1 = 6$. Cette suite n'est pas arithmétique car on ne passe pas d'un terme au suivant en ajoutant toujours le même nombre (on ajoute 0 pour passer de u_0 à u_1 et on ajoute 1 pour passer u_1 de à u_2).
– Soit (v_n) la suite numérique définie par $v_0 = 7$ et $v_{n+1} = v_n - 0{,}1$. Cette suite est arithmétique de raison $r = -0{,}1$ puisqu'on ajoute toujours $-0{,}1$ pour passer d'un terme au suivant. On a donc, pour tout entier $n \geq 0$, $v_n = v_0 + n \times (-0{,}1) = 7 - 0{,}1\,n$.

Propriété 2.2

Une suite de terme général $an + b$ (avec $a, b \in \mathbb{R}$) est arithmétique de raison a.
En effet si $u_n = an + b$, alors $u_{n+1} = a(n + 1) + b = an + a + b = an + b + a = u_n + a$.

Exemples

Des suites de termes généraux $7n + 2$, $-3{,}2n + 5$ et $n - 10$ sont arithmétiques de raisons 7, –3,2 et 1 respectivement.

Nous allons voir dans la propriété suivante que si (u_n) est une suite arithmétique, on peut calculer toute somme de termes consécutifs de cette suite (par exemple $S = u_{10} + u_{11} + \ldots + u_{100}$) en utilisant uniquement le nombre de termes, le premier et le dernier terme de cette somme (S peut donc se calculer à partir de 91, u_{10} et u_{100}).

Propriété 2.3

Si S est une somme de termes consécutifs d'une suite arithmétique, alors on a :
S = *nombre de termes* × (*premier terme* + *dernier terme*) / 2

Cas particuliers importants – Si (u_n) est une suite arithmétique alors :

- Pour tout entier $n \geq 0$, $u_0 + u_1 + ... + u_n = (n+1)\dfrac{u_0 + u_n}{2}$
- Pour tout entier $n \geq 1$, $u_1 + ... + u_n = n\dfrac{u_1 + u_n}{2}$ (*)
- Pour tout entier $n \geq 1$, $1 + 2 + ... + n = \dfrac{n(n+1)}{2}$ (**)

On obtient l'égalité (**) en utilisant (*) avec la suite arithmétique (u_n) de terme général n.

Exemple

$1 + 2 + ... + 99 + 100 = 100 \times (100+1) / 2 = 5\ 050$

Exercice d'application 2.1

Soit (u_n) la suite arithmétique de raison $r = 3$ et de premier terme $u_0 = 5$.

a) Exprimer u_n en fonction de n.

b) Calculer $u_0 + u_1 + ... + u_{89}$.

SOLUTION. a) $u_n = u_0 + nr = 5 + 3n$

b) $u_0 + u_1 + ... + u_{89} = 90 \times (u_0 + u_{89})/2 = 90 \times (5 + (5 + 3 \times 89))/2 = 12\ 465$

2.2.2 Suites géométriques

On dit qu'une suite (u_n) est **géométrique** lorsqu'on passe d'un terme de cette suite au suivant en multipliant toujours par le même nombre, c'est-à-dire s'il existe un réel q tel que pour tout entier n, $u_{n+1} = u_n \times q$. Le nombre q est alors appelé la **raison** de la suite (u_n).

Exemples

Soit (v_n) la suite définie par $v_0=3$ et $v_{n+1} = 2v_n$. On passe d'un terme de la suite (v_n) au suivant en multipliant par 2 : $v_1 = 2v_0 = 3 \times 2 = 6$; $v_2 = 2v_1 = 6 \times 2 = 12$; $v_3 = 2v_2 = 12 \times 2 = 24$; etc. La suite (v_n) est géométrique de raison 2 et de premier terme 3.

Propriété 2.4

Si (u_n) est une suite géométrique de raison q alors :
– Pour tout entier $n \geq 0$, $u_n = u_0 \times q^n$.
– Pour tout entier $n \geq 1$, $u_n = u_1 \times q^{n-1}$.
– Pour tout entier $n \geq 2$, $u_n = u_2 \times q^{n-2}$.
– Etc.

Exemples

▪ Soit (u_n) la suite numérique définie par $u_1 = 2$ et $u_{n+1} = nu_n$. On a $u_2 = 1 \times u_1 = 1 \times 2 = 2$ et $u_3 = 2 \times u_2 = 2 \times 2 = 4$. On ne multiplie pas toujours par le même nombre pour passer d'un terme au suivant : on multiplie par 1 pour passer de u_1 à u_2 et par 2 pour passer de u_2 à u_3. Cette suite n'est donc pas géométrique.

▪ Soit (v_n) la suite numérique définie par $v_0 = 2$ et $v_{n+1} = v_n/10$. Cette dernière égalité peut s'écrire $v_{n+1} = 0{,}1\,v_n$. On multiplie donc toujours par 0,1 pour passer d'un terme au suivant, ce qui montre que la suite (v_n) est géométrique de raison 0,1. On a alors, pour tout entier $n \geq 0$, $v_n = v_0 \times 0{,}1^n = 2 \times 0{,}1^n$.

Propriété 2.5

Une suite de terme général $a \times b^n$ (avec $a, b \in \mathbb{R}$) est géométrique de raison b. En effet si $u_n = a \times b^n$, alors $u_{n+1} = a \times b^{n+1} = a \times b^n \times b = u_n \times b$.

Exemple

Des suites de termes généraux 3^n, $6 \times (-2)^n$ et $-4 \times 1{,}8^n$ sont géométriques de raisons 3, –2 et 1,8 respectivement.

Nous allons voir dans la propriété suivante que si (u_n) est une suite géométrique de raison q, on peut calculer toute somme de termes consécutifs de cette suite (comme par exemple $S = u_0 + u_1 + u_2 + \ldots + u_{80}$) en utilisant uniquement la raison de la suite, le nombre de termes et le premier terme de la somme (S peut donc se calculer à partir des nombres q, 81 et u_0).

Propriété 2.6

Si S est une somme de termes consécutifs d'une suite géométrique de raison $q \neq 1$, alors on a $S = \textit{premier terme} \times (1 - q^{\text{nombre de termes}}) / (1 - q)$.

Cas particuliers importants – Si (u_n) est une suite géométrique de raison $q \neq 1$ alors :

- Pour tout entier $n \geq 0$, $u_0 + u_1 + \ldots + u_n = u_0 \times \dfrac{1 - q^{n+1}}{1 - q}$. (*)
- Pour tout entier $n \geq 1$, $u_1 + u_2 + \ldots + u_n = u_1 \times \dfrac{1 - q^n}{1 - q}$.
- Pour tout entier $n \geq 0$, $1 + q + q^2 + \ldots + q^n = \dfrac{1 - q^{n+1}}{1 - q}$. (**)

On obtient (**) en utilisant (*) avec la suite géométrique (u_n) de terme général q^n.
Si $q = 1$, tous les termes de la suite (u_n) sont égaux à u_0, donc :

$$u_0 + u_1 + \ldots + u_n = (n+1)u_0$$

Exemple

$1 + 2 + 2^2 + \ldots + 2^{20} = (1 - 2^{21}) / (1 - 2) = 2\,097\,151$

Exercice d'application 2.2

Soit (u_n) une suite géométrique de raison $q = 0{,}9$ et de premier terme $u_0 = 10$.

a) Exprimer u_n en fonction de n.

b) Déterminer $u_0 + u_1 + \ldots + u_{24}$.

SOLUTION. a) $u_n = u_0 \times q^n = 10 \times 0{,}9^n$

b) $u_0 + u_1 + \ldots + u_{24} = u_0 \times (1 - q^{24+1}) / (1 - q) = 10 \times (1 - 0{,}9^{25}) / (1 - 0{,}9) \approx 92{,}82$ (à 10^{-2} près)

2.3 Variations d'une suite

On dit qu'une suite (u_n) de nombres réels est :

- si ses termes sont rangés par ordre croissant, c'est-à-dire si pour tout n, $u_n \leq u_{n+1}$.
- si ses termes sont rangés par ordre décroissant, c'est-à-dire si pour tout n, $u_n \geq u_{n+1}$.
- **monotone** si elle est croissante ou décroissante.

Exemples

– Soit (u_n) la suite définie pour tout $n \geq 0$ par $u_0 = 36$ et $u_{n+1} = u_n/2 + 4$.
Ses premiers termes sont : $u_0 = 36$; $u_1 = 36/2 + 4 = 22$; $u_2 = 22/2 + 4 = 15$; $u_3 = 15/2 + 4 = 11{,}5$; etc. On les a obtenus avec un tableur : en rentrant 36 dans la cellule B2 puis « =B2/2+4 » dans la cellule B3 avant de l'étirer vers le bas. On peut remarquer que les termes sont rangés par ordre décroissant, la suite (u_n) est donc décroissante.
– Soit (u_n) la suite définie pour tout $n \geq 0$ par $u_n = 3n + 1$. Ses termes sont 1, 4, 7, 10, etc. Ils sont rangés par ordre croissant. La suite (u_n) est donc croissante.
– Les termes de la suite (v_n) définie pour tout $n \geq 0$ par $v_n = (-2)^n$ sont $v_0 = 1$, $v_1 = -2$, $v_2 = 4$, $v_3 = -8$, etc. Ils ne sont rangés ni par ordre croissant, ni par ordre décroissant, la suite n'est donc pas monotone.

	A	B
1	n	Un
2	0	36
3	1	22
4	2	15
5	3	11,5
6	4	9,75
7	5	8,875
8	6	8,4375
9	7	8,21875
10	8	8,109375
11	9	8,0546875
12	10	8,02734375

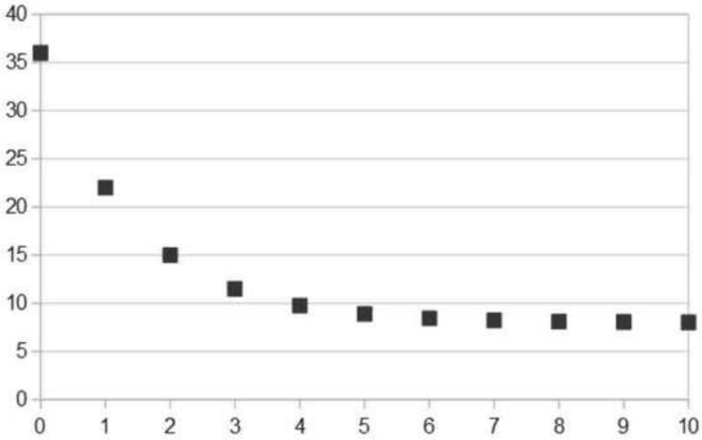


Figure 2.2

Propriété 2.7

Soit (   ) une suite arithmétique de raison   . Si   > 0 alors (   ) est croissante, si   < 0 alors (   ) est décroissante.

Exemples

Une suite de terme général $5n-3$ est croissante.
Une suite de terme général $17-3n$ est décroissante.

Propriété 2.8

Soient   un nombre réel strictement positif et (   ) une suite de terme général   . Si 0 <   < 1 alors (   ) est décroissante. Si   > 1 alors (   ) est croissante.

Exemples

Une suite de terme général 2^n est croissante.
Une suite de terme général $0{,}9^n$ est décroissante.

Exercice d'application 2.3

Soit (u_n) la suite définie pour tout $n \geq 0$ par $u_n = 0{,}7^{3n}$. Quel est le plus petit entier n tel que $u_n < 10^{-4}$?

SOLUTION. $u_n = 0{,}7^{3n} = (0{,}7^3)^n = 0{,}343^n$; $0 < 0{,}343 < 1$, la suite (u_n) est donc décroissante. Avec une calculatrice, on obtient $u_8 \approx 2 \times 10^{-4}$ et $u_9 \approx 7 \times 10^{-5}$, d'où $u_8 > 10^{-4}$ et $u_9 < 10^{-4}$. La valeur cherchée est donc 9.

2.4 Limite d'une suite

Soient (u_n) une suite numérique et L un nombre réel :

- On dit que la limite de la suite (u_n) est L (ou que la suite tend vers L) lorsque, pour tout entier n suffisamment grand, tous les u_n peuvent être rendus aussi proches de L qu'on le souhaite. On peut alors noter : $\lim\limits_{n\to+\infty} u_n = L$.
- On dit que la limite de la suite (u_n) est $+\infty$ (ou que la suite tend vers $+\infty$) lorsque, pour tout entier n suffisamment grand, tous les u_n peuvent être rendus aussi grands qu'on le souhaite. On peut alors noter : $\lim\limits_{n\to+\infty} u_n = +\infty$.
- On dit que la limite de la suite (u_n) est $-\infty$ (ou que la suite tend vers $-\infty$) lorsque, pour tout entier n suffisamment grand, tous les u_n peuvent être rendus aussi petits qu'on le souhaite. On peut alors noter : $\lim\limits_{n\to+\infty} u_n = -\infty$.

Exemple

Soient (u_n), (v_n) et (w_n) les suites numériques définies pour tout $n \geq 1$ par :
$u_n = 100/n$, $v_n = n^2$ et $w_n = 1 - 10n$.
On a représenté à partir d'un tableur les premiers termes de ces suites.

	A	B	C	D
1	n	Un	Vn	Wn
2	1	100	1	-9
3	2	50	4	-19
4	3	33,33333333	9	-29
5	4	25	16	-39
6	5	20	25	-49
7	6	16,66666667	36	-59
8	7	14,28571429	49	-69
9	8	12,5	64	-79
10	9	11,11111111	81	-89
11	10	10	100	-99
12	11	9,090909091	121	-109
13	12	8,333333333	144	-119
14	13	7,692307692	169	-129
15	14	7,142857143	196	-139

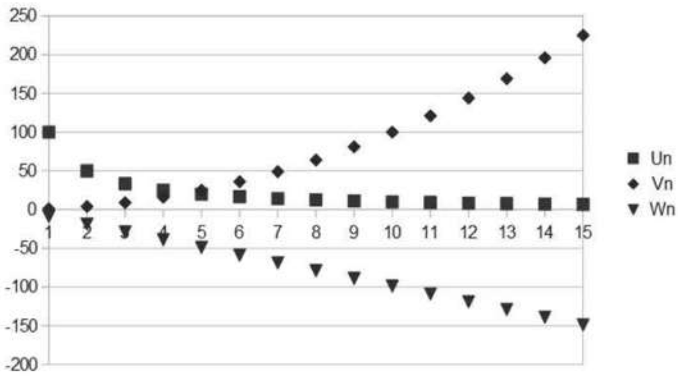


Figure 2.3

On peut conjecturer et on pourrait démontrer que pour n suffisamment grand :
- Tous les u_n peuvent être rendus aussi proches de 0 qu'on le souhaite.
- Tous les v_n peuvent être rendus aussi grands qu'on le souhaite.
- Tous les w_n peuvent être rendus aussi petits qu'on le souhaite.

La limite de la suite (u_n) est donc 0, celle de la suite (v_n) est $+\infty$ et celle de la suite (w_n) est $-\infty$. On peut écrire : $\lim\limits_{n\to+\infty} u_n = 0$; $\lim\limits_{n\to+\infty} v_n = +\infty$; $\lim\limits_{n\to+\infty} w_n = -\infty$.

Considérons la suite (u_n) définie pour tout $n \geq 0$ par $u_n = (-1)^n$. Les termes de cette suite sont 1 ; –1 ; 1 ; –1 ; 1 ; … Lorsque n devient grand, les u_n ne se stabilisent vers aucune valeur particulière. On dit que cette suite n'a pas de limite.

Exemple

Soient (u_n), (v_n) les suites géométriques définies pour tout $n \geq 0$ par $u_n = 10 \times 0{,}8^n$ et $v_n = 0{,}8 \times 1{,}2^n$. On a représenté fig. 2.4 les premiers termes de ces suites.

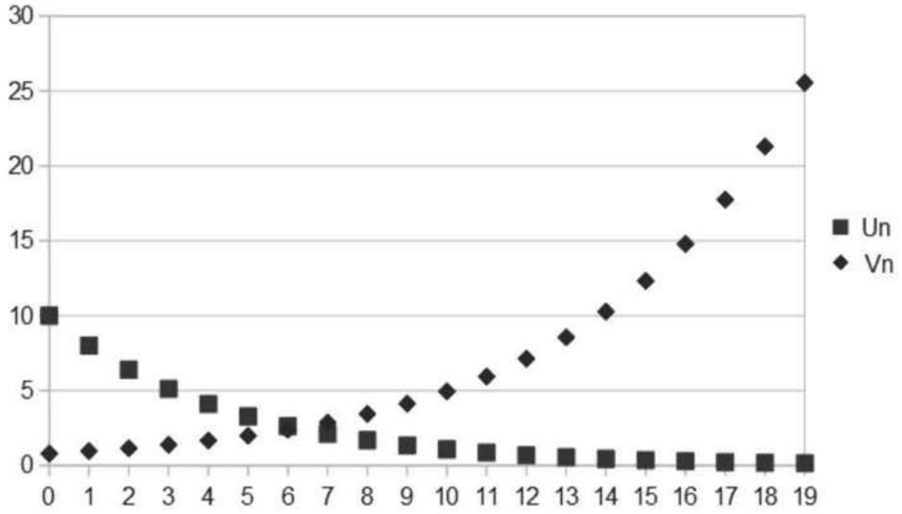


Figure 2.4

On peut conjecturer que (u_n) a pour limite 0, tandis que (v_n) a pour limite $+\infty$.

Propriété 2.9

Soient q et λ des nombres réels avec $q > 0$:

– Si $0 < q < 1$ alors $\lim\limits_{n\to+\infty}\left(\lambda\times q^n\right)=0$

–Si $q > 1$ et $\lambda > 0$ alors $\lim\limits_{n\to+\infty}\left(\lambda\times q^n\right)=+\infty$

– Si $q > 1$ et $\lambda < 0$ alors $\lim\limits_{n\to+\infty}\left(\lambda\times q^n\right)=-\infty$.

Exemples

$\lim\limits_{n\to+\infty} 0,98^n = 0$ car $0 < 0,98 < 1$.

$\lim\limits_{n\to+\infty}(3\times 1,01^n)=+\infty$ car $1,01 > 1$ et $3 > 0$.

$\lim\limits_{n\to+\infty} 1,2^{2n} = \lim\limits_{n\to+\infty}\left(1,2^2\right)^n = \lim\limits_{n\to+\infty} 1,44^n = +\infty$ car $1,44 > 1$.

$\lim\left(-6\times 2,4^n\right)=-\infty$ car $2,4 > 1$ et $-6 < 0$.

Propriété 2.10

Soit (u_n) une suite, soient L et λ des nombres réels. Si $\lim\limits_{n\to+\infty} u_n = L$ alors $\lim\limits_{n\to+\infty}\left(u_n+\lambda\right)=L+\lambda$.

Exemple

$\lim\limits_{n\to+\infty} 0,9^n = 0$ car $0 < 0,9 < 1$, donc $\lim\limits_{n\to+\infty}\left(5+0,9^n\right)=5$

TD – Évolution d'une liste de diffusion

Le responsable d'un site internet a mis en place une liste de diffusion pour fidéliser ses utilisateurs. Cette liste compte actuellement exactement 10 000 inscrits.

Depuis sa création, on a pu constater que chaque semaine 20 % se désinscrivent alors que dans le même temps 4 800 utilisateurs s'inscrivent. On suppose que le nombre d'inscrits va suivre cette évolution.

Le but de ce TP est alors de répondre aux questions suivantes :

- **Question 1** : le nombre d'inscrits va-t-il finir par se stabiliser ?
- **Question 2** : arrivera-t-on à dépasser les 23 000 inscrits ? Si oui, en combien de temps ?

Pour cela, on note u_n le nombre d'inscrits (en milliers) dans n semaine(s). On a donc :

$$u_0 = 10 \text{ et } u_{n+1} = 0{,}8u_n + 4{,}8$$

1. Conjecture

À l'aide d'une calculatrice, émettre une conjecture sur la limite de la suite (u_n).

2. Suite auxiliaire

Pour tout $n \in \mathbb{N}$, on pose $v_n = (24 - u_n) / 14$. On a utilisé un tableur, avec la disposition de la fig. 2.5, pour obtenir les premiers termes de la suite (u_n) et ceux de la suite (v_n).

	A	B	C
1	n	Un	Vn
2	0		
3	1		
4	2		
5	3		
6	4		

Figure 2.5

a) Qu'a-t-on saisi dans la cellule B2, dans la cellule B3 avant de l'étirer vers le bas et dans la cellule C2 avant de l'étirer vers le bas ?

b) Émettre une conjecture sur l'expression de v_n en fonction de n.

3. Démonstrations

a) Montrer que (v_n) est une suite géométrique de raison 0,8 et de premier terme $v_0 = 1$.

b) En déduire l'expression de v_n, puis celle de u_n en fonction de n.

c) Combien devrait-il y avoir d'inscrits dans dix semaines ?

4. Conclusion

a) Répondre à la question 1.

b) Écrire puis implémenter un algorithme permettant de répondre à la question 2.

Solution du TD

1. Conjecture

On saisit « 10 » puis on valide. On saisit ensuite « ANS×0.8+4.8 » puis on valide un certain nombre de fois. On obtient les premiers termes de la suite (u_n). On peut alors conjecturer que $\lim\limits_{n\to+\infty} u_n = 24$.

2. Suite auxiliaire

a) Dans les cellules B2, B3 et C2 on a saisi respectivement « 10 », « =0,8*B2+4,8 » et « =(24–B2)/14 ».

	A	B	C
1	n	Un	Vn
2	0	10	1
3	1	12,8	0,8
4	2	15,04	0,64
5	3	16,832	0,512
6	4	18,2656	0,4096

Figure 2.6

b) On peut conjecturer que $v_n = 0{,}8^n$.

3. Démonstrations

a) $v_{n+1} = (24 - u_{n+1}) / 14 = (24 - (0{,}8u_n + 4{,}8)) / 14 = (24 - 0{,}8u_n - 4{,}8) /14 = (19{,}2 - 0{,}8u_n) / 14 = 0{,}8 \times (24 - u_n)/14 = 0{,}8v_n$

La suite (v_n) est donc une suite géométrique de raison 0,8. Son premier terme est $v_0 = (24 - u_0) / 14 = (24 - 10) / 14 = 1$.

b) $v_n = v_0 \times 0{,}8^n = 1 \times 0{,}8^n = 0{,}8^n$

$v_n = (24 - u_n) / 14$ donc $24 - u_n = 14\ v_n$, d'où $u_n = 24 - 14\ v_n = 24 - 14 \times 0{,}8^n$.

c) $u_{10} = 24 - 14 \times 0{,}8^{10} \approx 22{,}497$. Dans dix semaines, il devrait y avoir environ 22 500 inscrits.

4. Conclusion

a) Puisque $0 < 0{,}8 < 1$, alors $\lim\limits_{n\to+\infty}\left(-14\times 0{,}8^n\right) = 0$, d'où $\lim\limits_{n\to+\infty} u_n = 24 + 0 = 24$.

Réponse à la question 1 : le nombre d'inscrits va finir par se stabiliser vers 24 000.

b) Puisque $\lim_{n\to+\infty} u_n = 24$, tous les u_n, à partir d'un certain rang, dépassent 23. Le nombre d'inscrits finira donc par dépasser 23 000. L'algorithme suivant permet d'obtenir la première semaine où ce nombre sera effectivement dépassé.

```
Variable : n (entier)
Début
  n ← 0
  TantQue 24-14*0.8^n<=23 Faire
    n ← n+1
  FinTantQue
  Afficher n
Fin
```

L'exécution de l'algorithme donne $n = 12$.

Réponse à la question 2 : le palier des 23 000 inscrits sera franchi en 12 semaines.

Exercices corrigés

2.1 Calculer les trois premiers termes de la suite (u_n) définie pour tout entier $n \geq 0$ par $u_n = n^2 - 10n$.

2.2 Soit (v_n) la suite définie par $v_0 = 1{,}2$ et pour tout entier $n \geq 0$ par $v_{n+1} = 10v_n + 0{,}1$. Calculer v_1 et v_2.

2.3 Soit (u_n) la suite définie par $u_0 = 2$ et pour tout entier $n \geq 0$ par $u_{n+1} = 2u_n + 3$. Pour tout $n \in \mathbb{N}$, on pose $S_n = u_0 + u_1 + \ldots + u_n$. On souhaite déterminer, à l'aide d'un tableur, quelques valeurs particulières de u_n et de S_n. On adopte pour cela la disposition de la fig. 2.7.

	A	B	C
1	n	Un	Sn
2	0		
3	1		
4	2		
5	3		
6	4		

Figure 2.7

a) Que doit-on rentrer dans la cellule B2 ?

b) Quelle formule doit-on saisir dans la cellule B3, avant de l'étirer vers le bas ?

c) Que peut-on rentrer dans la cellule C2 ?

d) Quelle formule doit-on saisir dans la cellule C3, avant de l'étirer vers le bas ?

e) Déterminer, à l'aide du tableur, les valeurs de u_{12} et S_{12}.

2.4 On place 5 000 € sur un compte bancaire rémunéré à 1,2 % par an. Cela signifie que chaque année, on perçoit des intérêts d'un montant égal à 1,2 % du capital, qui s'ajoutent alors au capital. Le montant du capital sera donc de $5\,000 \times 1{,}012 = 5\,060$ € au bout d'un an, puis de $5\,060 \times 1{,}012 = 5\,120{,}72$ € au bout de deux ans, etc.

On souhaite déterminer, à l'aide d'un tableur, la durée, en année, à partir de laquelle, le capital dépassera 6 000 €. On utilise pour cela un tableur avec la disposition de la fig. 2.8.

	A	B
1	Année	Capital
2	0	
3	1	
4	2	
5	3	

Figure 2.8

a) Que doit-on rentrer dans la cellule B2 ?

b) Quelle formule doit-on rentrer dans la cellule B3, avant de l'étirer vers le bas ?

c) Quel sera le capital acquis à 5 ans ?

d) Répondre au problème posé.

2.5 Quels sont les six premiers termes de la suite (u_n) définie par $u_0 = 1$, $u_1 = 1$ et $u_{n+2} = u_{n+1} + u_n$?

2.6 Soient (u_n), (v_n) et (w_n) les suites définies pour tout entier naturel n par $u_n = n^3$, $v_n = 5n$ et $w_n = 3^n/2$. Pour chacune de ces suites, dire si elle est arithmétique, géométrique ou ni arithmétique et ni géométrique.

2.7 Dans chacun des cas suivants, calculer u_k.

a) (u_n) est une suite arithmétique de raison 3 et on a $u_0 = -11$ et $k = 12$.

b) (u_n) est une suite arithmétique et on a $u_0 = 12$, $u_4 = 20$ et $k = 20$.

2.8 Dans chacun des cas suivants, calculer u_k.

a) (u_n) est une suite géométrique de raison 5 et on a $u_0 = 0{,}44$ et $k = 7$.

b) (u_n) est une suite géométrique de raison 0,4 et on a $u_3 = 16$ et $k = 8$.

c) (u_n) est une suite géométrique de raison $q > 0$ et on a $u_4 = 4{,}8$, $u_6 = 19{,}2$ et $k = 9$.

2.9 Soit (u_n) une suite arithmétique de premier terme $u_0 = 3$ et de raison $r = 2$. Montrer que la suite (v_n) définie par $v_n = 3^{u_n}$ est géométrique. Préciser son premier terme et sa raison.

2.10 Le nombre 10 peut-il être le premier terme d'une suite arithmétique de raison 3 dont un des termes est 101 ?

2.11 Le nombre 3,75 peut-il être le premier terme d'une suite géométrique de raison 2 dont un des termes est 240 ?

2.12 Soit (w_n) la suite définie par $w_0 = 1$ et $w_{n+1} = w_n/(w_n+1)$.

1) Calculer w_1 et w_2.

2) Soit (v_n) la suite définie pour tout entier naturel n par $v_n = 1/w_n$.

a) Montrer que (v_n) est une suite arithmétique. Préciser sa raison.

b) En déduire l'expression de v_n puis celle de w_n en fonction de n.

2.13 Calculer les sommes suivantes :

- $S = 1 + 2 + 2^2 + 2^3 + \ldots + 2^{30}$
- $T = 0{,}1 + 0{,}2 + 0{,}3 + \ldots + 9{,}9 + 10$

2.14 Calculer les sommes suivantes :

- $U = 2 + 4 + 6 + 8 + \ldots + 998 + 1\,000$
- $V = v_1 + v_2 + \ldots + v_{10}$

où (v_n) est une suite géométrique de premier terme $v_0 = 10$ et de raison $q = 0{,}5$.

2.15 Déterminer l'entier n tel que $1 + 2 + 2^2 + 2^3 + ... + 2^n = 2\,047$.

2.16 Calculer la somme des cent premiers nombres impairs positifs.

2.17 (Complexité d'un algorithme) On considère l'algorithme suivant :

```
Variables : i, j, n, S (entiers)
Début
  S ← 0
  Saisir n
  Pour i de 1 à n Faire
    Pour j de 1 à i Faire
      S ← S+2
    FinPour
  FinPour
  Afficher S
Fin
```


a) Déterminer, en fonction de l'entier $n \geq 1$ saisi par l'utilisateur, le nombre d'additions effectuées pendant l'exécution de l'algorithme.

b) Quelle valeur (en fonction de n) de S sera affichée, après exécution ?

c) En déduire la valeur affichée lorsque l'utilisateur saisit 50.

2.18 (Complexité d'un algorithme) On considère l'algorithme suivant :

```
Variables : i, j, k, n, S (entiers)
Début
  S ← 0
  Saisir n
  Pour i de 1 à n Faire
    Pour j de 1 à i Faire
      Pour k de 1 à i Faire
        S ← S+3
      FinPour
    FinPour
  FinPour
  Afficher S
Fin
```

a) On admet que pour tout entier $n \geq 1$, on a $1^2 + 2^2 + 3^2 + n^2 = n(n + 1)(2n + 1) / 6$. En déduire, en fonction de l'entier $n \geq 1$ saisi par l'utilisateur, le nombre d'additions effectuées pendant l'exécution de l'algorithme.

b) Quelle valeur (en fonction de n) de S sera affichée, après exécution ?

c) En déduire la valeur affichée lorsque l'utilisateur saisit 20.

2.19 Soit (u_n) la suite définie par $u_0 = 20$ et pour tout entier naturel n par $u_{n+1} = 0{,}2u_n$.

1) a) Quelle est la nature de cette suite ?

b) En déduire l'expression de u_n en fonction de n.

2) Pour tout $n \in \mathbb{N}$, on pose $S_n = u_0 + u_1 + \ldots + u_n$.

a) Déterminer une expression simplifiée de S_n en fonction de n.

b) En déduire l'arrondi à 10^{-3} près de S_{12}.

2.20 Soit (u_n) la suite arithmétique de premier terme $u_0 = 12$ et de raison $r = 16$. Pour tout entier naturel n, on pose $S_n = u_0 + u_1 + \ldots + u_n$.

1) a) Donner l'expression de u_n en fonction de n.

b) Déterminer le plus petit entier n tel que $u_n > 500$.

2) a) Donner l'expression de S_n en fonction de n.

b) Déterminer, à l'aide d'un tableur, le plus petit entier n tel que $S_n > 1\,000$.

2.21 Donner le sens de variations de chacune des suites (u_n), (v_n) et (w_n) définies pour tout $n \in \mathbb{N}$ par $u_n = \left(\sqrt{2}\right)^{4n}$, $v_n = 1 - 2n$ et $w_n = 1/4^n$.

2.22 Montrer que la suite (u_n) définie pour tout entier $n \geq 0$ par $u_n = n/2^n$ n'est pas monotone.

2.23 Soit (u_n) la suite définie par $u_n = 1{,}5^n + 9$.

a) Déterminer le plus petit entier naturel n tel que $u_n > 300$.

b) Écrire un algorithme permettant de déterminer le plus petit entier n tel que $u_n > 10^{12}$.

2.24 Soit (u_n) la suite définie par $u_n = 5 \times 2^{-2n+1}$.

a) Déterminer le plus petit entier naturel n tel que $u_n < 10^{-6}$.

b) Écrire un algorithme permettant de déterminer le plus petit entier n tel que $u_n < 10^{-20}$.

2.25 Soit (u_n) la suite définie par $u_0 = 1$ et $u_{n+1} = (u_n + 1)/4$.

a) Calculer u_1 et u_2.

b) Émettre une conjecture sur $\lim\limits_{n\to+\infty} u_n$ (à partir d'une calculatrice).

2.26 Déterminer les limites suivantes :

$$A = \lim_{n\to+\infty} 0{,}3^n \qquad B = \lim_{n\to+\infty} 1{,}07^n \qquad C = \lim_{n\to+\infty} (-5 \times 1{,}7^n)$$

$$D = \lim_{n\to+\infty} \frac{5 \times 2^n}{3^n} \qquad E = \lim_{n\to+\infty} 2^{-2n}$$

2.27 Déterminer les limites suivantes :

$$A = \lim_{n\to+\infty} \left(7 - 0{,}77^n\right) \qquad B = \lim_{n\to+\infty} \left(3 + 0{,}99^n\right)$$

$$C = \lim_{n\to+\infty} (50 - 3 \times 0{,}09^n) \qquad D = \lim_{n\to+\infty} \left(6 + \frac{2 \times 3^n}{5^n}\right)$$

2.28 On achète 430 € un ordinateur portable. On estime, qu'une fois sorti du magasin sa valeur u_n (en euro) après n mois est donnée par la formule :

$$u_n = 40 + 300 \times 0{,}95^n$$

a) Que vaut l'ordinateur à la sortie du magasin ?

b) Que vaut l'ordinateur un an après l'achat ?

c) À long terme, à quel prix peut-on espérer revendre cet ordinateur ?

d) Déterminer le mois à partir duquel l'ordinateur aura une valeur inférieure à 100 €.

2.29 Avec un tableur, on saisit « 1 » dans la cellule A1, puis « =2*A1+1 » dans la cellule A2 avant de l'étirer vers le bas (fig. 2.9).

Pour tout entier $n \geq 1$, on note u_n la valeur contenue dans la cellule An.

1) Donner u_1, u_2, u_3 et u_4.

2) Exprimer u_{n+1} en fonction de u_n.

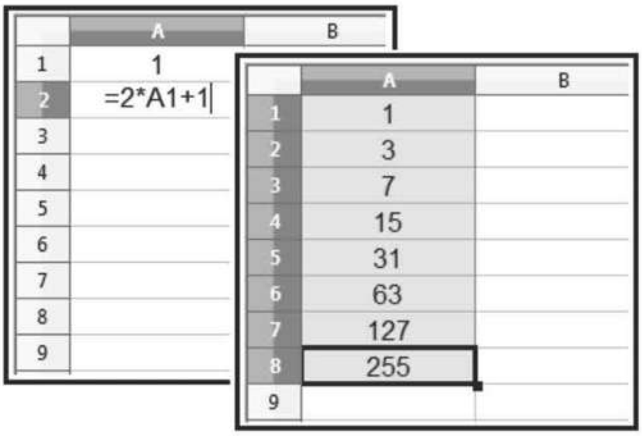


Figure 2.9

3) Calculer u_9.

4) Émettre une conjecture sur l'expression de u_n en fonction de n.

5) Soit (v_n) la suite définie pour tout entier $n \geq 1$ par $v_n = u_n + 1$.

a) Montrer que la suite (v_n) est géométrique.

b) En déduire l'expression de v_n puis de u_n en fonction de n.

c) À partir de quelle ligne n, le nombre contenu dans la cellule An sera-t-il supérieur à 10^{10} ?

2.30 La population mondiale était de 7,2 milliards en 2014. Elle augmente de 1,2 % par an. On note u_n la population mondiale (en milliards) de l'année $2014 + n$.

a) Donner une estimation de la population mondiale en 2016.

b) Quelle est la nature de la suite (u_n) ?

c) Exprimer u_n en fonction de n.

d) Quelle devrait être la population mondiale en 2030 ?

e) À partir de quelle année, devrions-nous dépasser les 10 milliards d'habitants sur Terre ?

2.31 Le 1er janvier 2014, on place une somme de $C_0 = 10\ 000$ € sur un compte bancaire au taux annuel de 3 % avec intérêts composés, ce qui signifie que des intérêts d'une valeur de 3 % du capital s'ajoutent au capital chaque 31 décembre. On note C_n le capital obtenu le 1er janvier de l'année $2014 + n$.

a) Calculer C_1 et C_2.

b) Quel sera le capital obtenu le 1er janvier 2017 ?

c) Montrer que la suite (C_n) est géométrique. Préciser son premier terme et sa raison.

d) En déduire l'expression de C_n en fonction de n.

e) À partir de quelle année le capital dépassera-t-il 15 000 € ?

f) Déterminer la somme des intérêts perçus entre le 31 décembre 2014 et le 31 décembre 2024 inclus (arrondir au centime d'euro près).

2.32 Le responsable d'un magasin de matériel informatique a le choix entre deux grossistes qui lui proposent leurs contrats d'achat pour s'approvisionner en tablettes d'un certain type :

- **Grossiste A** : le $n^{\text{ième}}$ mois, chaque tablette achetée sera facturée :

$$150 + 10 \times 2^{-n+2} \text{ €}$$

- **Grossiste B** : le $n^{\text{ième}}$ mois, chaque tablette achetée sera facturée 151 €.

Chacun de ces contrats est valable deux ans.

Pour tout entier $n \geq 1$, posons $u_n = 150 + 10 \times 2^{-n+2}$.

a) Quel est le contrat le plus avantageux le premier mois ? Le dernier mois ?

b) À partir de quel mois le grossiste A deviendra-t-il plus avantageux ?

c) Calculer $\lim\limits_{n \to +\infty} u_n$. En déduire le montant vers lequel se stabilisera le prix du matériel avec le grossiste A.

d) Le responsable du magasin prévoit une commande de 10 tablettes par mois pendant deux ans. Quel est alors le grossiste le plus avantageux ?

Solutions

2.1 $u_0 = 0^2 - 10 \times 0 = 0$; $u_1 = 1^2 - 10 \times 1 = -9$; $u_2 = 2^2 - 10 \times 2 = -16$

2.2 $v_1 = 10v_0 + 0{,}1 = 10 \times 1{,}2 + 0{,}1 = 12{,}1$;
$v_2 = 10v_1 + 0{,}1 = 10 \times 12{,}1 + 0{,}1 = 121{,}1$

2.3 a) 2 b) =2*B2+3 c) 2 d) = C2+B3

e) $u_{12} = 20\,477$ (cellule B14, fig. 2.10) et $S_{12} = 40\,916$ (cellule C14, fig. 2.10).

	A	B	C
1	n	Un	Sn
2	0	2	2
3	1	7	9
4	2	17	26
5	3	37	63
6	4	77	140
7	5	157	297
8	6	317	614
9	7	637	1251
10	8	1277	2528
11	9	2557	5085
12	10	5117	10202
13	11	10237	20439
14	12	20477	40916
15	13	40957	81873

Figure 2.10

2.4 a) 5 000 b) =1,012*B2

c) Au bout de 5 ans, on obtient un capital d'environ 5 307,29 € (cellule B7, fig. 2.11).

	A	B
1	Année	Capital
2	0	5000
3	1	5060
4	2	5120,72
5	3	5182,16864
6	4	5244,354664
7	5	5307,28692
8	6	5370,974363
9	7	5435,426055
10	8	5500,651168
11	9	5566,658982
12	10	5633,458889
13	11	5701,060396
14	12	5769,473121
15	13	5838,706798
16	14	5908,77128
17	15	5979,676535
18	16	6051,432654
19	17	6124,049846

Figure 2.11

d) Le capital dépassera 6 000 € au bout de 16 ans (environ 6 051,43 €, cellule B18, fig. 2.11).

2.5 Les six premiers termes de la suite (u_n) sont : $u_0 = 1$; $u_1 = 1$; $u_2 = u_1 + u_0 = 1 + 1 = 2$; $u_3 = u_2 + u_1 = 2 + 1 = 3$; $u_4 = u_3 + u_2 = 3 + 2 = 5$; $u_5 = u_4 + u_3 = 5 + 3 = 8$

2.6 $u_0 = 0$; $u_1 = 1$; $u_2 = 8$. On ne passe d'un terme de la suite (u_n) au suivant en ajoutant le même nombre, ni même en multipliant par un même nombre, cette suite n'est donc ni arithmétique, ni géométrique.

- (v_n) est une suite arithmétique de raison 5 car $v_{n+1} = v_n + 5$.
- (w_n) est une suite géométrique de raison 3 car $w_{n+1} = 3w_n$.

2.7 a) $u_n = u_0 + n \times 3 = -11 + 3n$; $u_{12} = -11 + 3 \times 12 = -11 + 36 = 25$

b) Soit r la raison de la suite (u_n) ; $u_n = u_0 + nr = 12 + nr$

$u_4 = 12 + r \times 4 = 20$ donc $4r = 20 - 12$, d'où $r = 2$.

$u_n = 12 + 2n$ donc $u_{20} = 12 + 2 \times 20 = 52$.

2.8 a) $u_n = u_0 \times 5^n = 0{,}44 \times 5^n$; $u_7 = 0{,}44 \times 5^7 = 34\,375$

b) $u_8 = u_3 \times 0{,}4^5 = 16 \times 0{,}4^5 = 0{,}16384$

c) Soit q la raison de la suite (u_n). $u_6 = u_4 \times q^2$, donc $19{,}2 = 4{,}8 \times q^2$ d'où $q^2 = 19{,}2/4{,}8 = 4$. Puisque $q > 0$, on a $q = \sqrt{4} = 2$.

$u_n = u_0 \times q^n = u_0 \times 2^n$; $u_4 = u_0 \times 2^4 = 4{,}8$ donc $u_0 = 4{,}8/2^4 = 0{,}3$.

$u_n = 0{,}3 \times 2^n$ donc $u_9 = 0{,}3 \times 2^9 = 153{,}6$.

2.9 $u_n = u_0 + nr = 3 + 2n$

$v_n = 3^{3+2n} = 3^3 \times (3^2)^n = 27 \times 9^n$

$v_0 = 27 \times 9^0 = 27$

$v_{n+1} = 27 \times 9^{n+1} = 27 \times 9^n \times 9 = v_n \times 9$

(v_n) est la suite géométrique de premier terme $v_0 = 27$ et de raison $q = 9$.

2.10 Si (u_n) est une suite arithmétique de premier terme $u_0 = 10$ et de raison $r = 3$, alors $u_n = u_0 + nr = 10 + 3n$. Si de plus, un des termes de la suite est 101, alors il existe un entier n tel que $u_n = 101$, donc tel que $3n + 10 = 101$. On aurait alors $3n = 91$, d'où $n = 91/3$, ce qui est impossible, car 91/3 n'est pas un entier.

Le nombre 10 ne peut pas être le premier terme d'une suite arithmétique de raison 3 dont un des termes est 101.

2.11 Soit (u_n) la suite géométrique de premier terme $u_0 = 3{,}75$ et de raison $q = 2$. On a alors $u_n = u_0 \times q^n = 3{,}75 \times 2^n$.

$$u_n = 240 \Leftrightarrow 3{,}75 \times 2^n = 240 \Leftrightarrow 2^n = 240/3{,}75 \Leftrightarrow 2^n = 64 \Leftrightarrow n = 6$$

Cette suite (u_n) convient. Le nombre 3,75 peut effectivement être le premier terme d'une suite géométrique de raison 2 dont un des termes est 240.

2.12 1) $w_1 = w_0/(w_0 + 1) = 1/(1 + 1) = 1/2$; $w_2 = w_1/(w_1 + 1) = (1/2)/(1/2 + 1) = 1/3$

2) a) $v_{n+1} = 1/w_{n+1} = (w_n + 1)/w_n = w_n/w_n + 1/w_n = 1 + v_n$. La suite (v_n) est donc arithmétique de raison 1.

b) Pour tout $n \geq 0$, $v_n = v_0 + n \times 1 = 1/w_0 + n = 1 + n$, donc $w_n = 1/v_n = 1/(n + 1)$.

2.13 $S = 1 + 2 + 2^2 + 2^3 + \ldots + 2^{30} = (1 - 2^{31})/(1 - 2) = 2\,147\,483\,647$ (somme de la forme $1 + q + q^2 + ... + q^n$).

$T = 0{,}1 + 0{,}2 + 0{,}3 + \ldots, 9{,}9 + 10 = 100 \times (0{,}1 + 10)/2 = 505$ (somme de 100 termes consécutifs d'une suite arithmétique de raison 0,1).

2.14 $U = 2 + 4 + 6 + \ldots + 998 + 1\,000 = 500 \times (2 + 1\,000)/2 = 250\,500$ (somme de 500 termes consécutifs d'une suite arithmétique de raison 2).

$V = v_1 + v_2 + \ldots + v_{10} = v_1 \times (1 - 0{,}5^{10})/(1 - 0{,}5) = 5 \times (1 - 0{,}5^{10})/(1 - 0{,}5) \approx 9{,}99$ (somme de 10 termes consécutifs d'une suite géométrique de raison 0,5).

2.15 $1 + 2 + 2^2 + 2^3 + \ldots + 2^n = (1 - 2^{n+1})/(1 - 2) = 2^{n+1} - 1$

$2^{n+1} - 1 = 2\,047 \Leftrightarrow 2^{n+1} = 2\,048 \Leftrightarrow n + 1 = 11 \Leftrightarrow n = 10$. L'entier cherché est 10.

2.16 La somme des 100 premiers nombres impairs positifs est $u_0 + u_1 + u_2 + \ldots + u_{99}$ (où $u_n = 2n + 1$), donc $u_0 + u_1 + u_2 + \ldots + u_{99} = 100 \times (u_0 + u_{99})/2 = 100 \times (1 + 199)/2 = 10\,000$.

2.17 a) Pour chaque valeur de i (de 1 à n), on compte i additions. On a donc au total $1 + 2 + 3 + \ldots + n = n(n + 1)/2$ additions.

b) Pour obtenir la valeur « finale » de S, on ajoute $n(n + 1)/2$ fois 2 à 0, donc $S = n(n + 1)/2 \times 2 = n(n + 1)$.

c) Lorsque l'utilisateur saisit 50, on obtient l'affichage « 2 550 », qui correspond au produit 50×51.

2.18 a) Pour chaque valeur de i (de 1 à n), on compte i^2 additions. On a donc au total $1^2 + 2^2 + 3^2 + \ldots + n^2 = n(n + 1)(2n + 1)/6$ additions.

b) Pour obtenir la valeur « finale » de S, on ajoute $n(n + 1)(2n + 1)/6$ fois 3 à 0. Donc $S = n(n + 1)(2n + 1)/6 \times 3 = n(n + 1)(2n + 1)/2$.

c) Lorsque l'utilisateur saisit 20, on obtient l'affichage « 8 610 », qui correspond à $(20 \times 21 \times 41)/2$.

2.19 1) a) La suite (u_n) est géométrique (de raison 0,2).

b) $u_n = u_0 \times 0{,}2^n = 20 \times 0{,}2^n$

2) a) $S_n = u_0 + u_1 + \ldots + u_n = u_0 \times (1 - 0{,}2^{n+1})/(1 - 0{,}2)$
$= 20 \times (1 - 0{,}2^{n+1})/(1 - 0{,}2) = 25 \times (1 - 0{,}2^{n+1})$

b) $S_{12} = 25 \times (1 - 0{,}2^{13}) \approx 25$ (à 10^{-3} près)

2.20 1) a) $u_n = u_0 + nr = 12 + 16n$

b) $u_n > 500 \Leftrightarrow 12 + 16n > 500 \Leftrightarrow 16n > 488 \Leftrightarrow n > 30{,}5$. Le plus petite entier n tel que $u_n > 500$ est 31.

2) a) $S_n = (n + 1) \times (u_0 + u_n)/2 = (n + 1) \times (12 + 12 + 16n)/2$
$= (n + 1) \times (24 + 16n)/2 = (n + 1)(12 + 8n)$

b) On a saisi « =(A2+1)*(12+8*A2) » dans la cellule B2 avant de l'étirer vers le bas. Le plus petit entier n tel que $S_n > 1\,000$ est 10.

	A	B
1	n	Sn
2	1	40
3	2	84
4	3	144
5	4	220
6	5	312
7	6	420
8	7	544
9	8	684
10	9	840
11	10	1012
12	11	1200

Figure 2.12

2.21 $u_n = \left(\left(\sqrt{2}\right)^4\right)^n = 4^n$ et $4 > 1$, donc (u_n) est croissante.

(v_n) est décroissante (c'est une suite arithmétique de raison –2).

$w_n = (1/4)^n$ et $0 < 1/4 < 1$ donc (w_n) est décroissante.

2.22 $u_0 = (0/2^0) = 0/1 = 0$; $u_1 = (1/2^1) = 1/2 = 0{,}5$; $u_2 = (2/2^2) = 2/4 = 1/2 = 0{,}5$; $u_3 = (3/2^3) = 3/8 = 0{,}375$. Les quatre premiers termes de la suite (u_n) ne sont rangés ni par ordre croissant, ni par ordre décroissant. Cette suite n'est pas monotone.

2.23 a) $1{,}5^n + 9 > 300 \Leftrightarrow 1{,}5^n > 291$. Puisque $1{,}5 > 1$, une suite de terme général $1{,}5^n$ est croissante. De plus $1{,}5^{13} < 291$ et $1{,}5^{14} > 291$ ($1{,}5^{13} \approx 194{,}6$ et $1{,}5^{14} \approx 291{,}9$). La plus petite valeur de l'entier n telle que $u_n > 300$ est donc 14.

b)

```
Variable : n (entier)
Début
  n ← 0
  TantQue 1,5^n+9<=10^12 Faire
  | n ← n+1
  FinTantQue
  Afficher n
Fin
```

2.24 a) $5 \times 2^{-2n+1} < 10^{-6} \Leftrightarrow 5 \times 2 \times (2^{-2})^n < 10^{-6} \approx 0{,}25^n < 10^{-6}/10 \Leftrightarrow 0{,}25^n < 10^{-7}$

Une suite de terme général $0{,}25^n$ est décroissante (car $0 < 0{,}25 < 1$).

$0{,}25^{11} > 10^{-7}$ et $0{,}25^{12} < 10^{-7}$, la plus petite valeur de l'entier n telle $u_n < 10^{-6}$ est donc 12.

b)

```
Variable : n (entier)
Début
  n ← 0
  TantQue 5*2^(-2n+1)>=10^(-20) Faire
  | n ← n+1
  FinTantQue
  Afficher n
Fin
```

2.25 a) $u_1 = (u_0 + 1) / 4 = 2 / 4 = 0{,}5$; $u_2 = (u_1 + 1) / 4 = 1{,}5 / 4 = 0{,}375$.

b) On saisit « 1 » puis on valide. On saisit alors « (ANS+1)/4 » puis on valide un certain nombre de fois. On obtient les premiers termes de la suite (u_n). On peut remarquer (valider au moins 18 fois) que l'on arrive à l'approximation 0,333333333. On peut conjecturer que $\lim\limits_{n \to +\infty} u_n = 1/3$.

2.26 $A = 0$ (car $0 < 0{,}3 < 1$) ; $B = +\infty$ (car $1{,}07 > 1$) ; $C = -\infty$ (car $1{,}7 > 1$ et $-5 < 0$)

$$D = \lim_{n\to+\infty} \frac{5\times 2^n}{3^n} = \lim_{n\to+\infty}\left(5\times\left(\frac{2}{3}\right)^n\right) = 0 \text{ (car } 0 < 2/3 < 1)$$

$$E = \lim_{n\to+\infty} 2^{-2n} = \lim_{n\to+\infty}\left(2^{-2}\right)^n = \lim_{n\to+\infty} 0,25^n = 0 \text{ (car } 0 < 0,25 < 1)$$

2.27 $\lim_{n\to+\infty} 0,77^n = 0$ car $0 < 0,77 < 1$ donc $A = \lim_{n\to+\infty}\left(7 - 0,77^n\right) = 7$

$\lim_{n\to+\infty} 0,99^n = 0$ car $0 < 0,99 < 1$ donc $B = \lim_{n\to+\infty}\left(3 + 0,99^n\right) = 3$

$\lim_{n\to+\infty}\left(-3\times 0,09^n\right) = 0$ car $0 < 0,09 < 1$ donc $C = \lim_{n\to+\infty}(50 - 3\times 0,09^n) = 50$

$$D = \lim_{n\to+\infty}\left(6 + \frac{2\times 3^n}{5^n}\right) = \lim_{n\to+\infty}\left(6 + 2\times\frac{3^n}{5^n}\right) = \lim_{n\to+\infty}\left(6 + 2\times\left(\frac{3}{5}\right)^n\right)$$

et $\lim_{n\to+\infty}\left(2\times\left(\frac{3}{5}\right)^n\right) = 0$ (car $0 < 3/5 < 1$), donc $D = 6$.

2.28 a) À la sortie du magasin, l'ordinateur vaut $u_0 = 40 + 300 \times 0,95^0$ $= 40 + 300 = 340$ €.

b) Un an après l'achat, l'ordinateur vaut $u_{12} = 40 + 300 \times 0,95^{12} \approx 202,11$ €.

c) $\lim_{n\to+\infty}\left(300\times 0,95^n\right) = 0$ (car $1 < 0,95 < 1$) donc $\lim_{n\to+\infty} u_n = 40$.

À long terme, on peut espérer revendre cet ordinateur au prix de 40 €.

d) $40 + 300 \times 0,95^n < 100 \Leftrightarrow 300 \times 0,95^n < 60 \Leftrightarrow 0,95^n < 60/300 \Leftrightarrow 0,95^n < 0,2$

Une suite de terme général $0,95^n$ est décroissante (car $0 < 0,95 < 1$).
De plus $0,95^{31} > 0,2$ et $0,95^{32} < 0,2$. L'ordinateur aura une valeur inférieure à 100 € à partir du 32ème mois.

2.29 1) $u_1 = 1$; $u_2 = 3$; $u_3 = 7$; $u_4 = 15$.

2) $u_{n+1} = 2u_n + 1$

3) $u_9 = 2u_8 + 1 = 2 \times 255 + 1 = 511$

4) Conjecture : pour tout entier $n \geq 1$, $u_n = 2^n - 1$.

5) a) $v_{n+1} = u_{n+1} + 1 = 2u_n + 1 + 1 = 2u_n + 2 = 2(u_n + 1) = 2v_n$

La suite (v_n) est donc géométrique de raison 2.

b) Pour tout entier $n \geq 1$, $v_n = v_1 \times 2^{n-1}$, or $v_1 = u_1 + 1 = 1 + 1 = 2$ donc $v_n = 2^n$ et $u_n = v_n - 1 = 2^n - 1$

c) $u_n > 10^{10} \Leftrightarrow 2^n - 1 > 10^{10} \Leftrightarrow 2^n > 10^{10} + 1$

Puisque $2 > 1$, une suite de terme général 2^n est croissante. De plus $2^{33} < 10^{10} + 1$ et $2^{34} > 10^{10} + 1$. Le nombre contenu dans la cellule An devient donc supérieur à 10^{10} à partir de la 34ème ligne.

2.30 a) $2\,016 = 2\,014 + 2$; en 2016 il devrait y avoir environ $7{,}2 \times 1{,}012^2 \approx 7{,}37$ milliards d'habitants sur Terre.

b) La suite (u_n) est géométrique de raison 1,012 puisque $u_{n+1} = u_n \times 1{,}012$.

c) On en déduit que $u_n = u_0 \times 1{,}012^n = 7{,}2 \times 1{,}012^n$.

d) En 2030 la population mondiale devrait s'élever à $u_{16} = 7{,}7 \times 1{,}012^{16} \approx 8{,}7$ milliards.

e) $7{,}2 \times 1{,}012^n > 10 \Leftrightarrow 1{,}012^n > 10/7{,}2$. Une suite de terme général $1{,}012^n$ est croissante (car $1{,}012 > 1$). De plus $1{,}012^{27} < 10/7{,}2$ et $1{,}012^{28} > 10/7{,}2$. Nous devrions donc dépasser les 10 milliards d'habitants à partir de l'année $2014 + 28 = 2\,042$.

2.31 a) $C_1 = C_0 + 0{,}03C_0 = 10\,000 + 0{,}03 \times 10\,000 = 10\,300$ €

$C_2 = C_1 + 0{,}03C_1 = 10\,300 + 0{,}03 \times 10\,300 = 10\,609$ €

b) Le capital obtenu le 1er janvier 2017 est :
$C_3 = C_2 + 0{,}03C_2 = 10\,609 + 0{,}03 \times 10\,609 = 10\,927{,}27$ €

c) $C_{n+1} = C_n + 0{,}03C_n = (1 + 0{,}03)C_n = 1{,}03C_n$. La suite (C_n) est donc géométrique de raison 1,03. Son premier terme est $C_0 = 10\,000$.

d) Pour tout entier naturel n, $C_n = C_0 \times 1{,}03^n = 10\,000 \times 1{,}03^n$.

e) $C_n > 15\,000 \Leftrightarrow 10\,000 \times 1{,}03^n > 15\,000 \Leftrightarrow 1{,}03^n > 1{,}5$. Puisque $1{,}03 > 1$, une suite de terme général $1{,}03^n$ est croissante. De plus $1{,}03^{13} < 1{,}5$ et $1{,}03^{14} > 1{,}5$. Le premier capital qui dépassera 15 000 € est donc C_{14}. Il sera obtenu le 1er janvier 2028.

f) La somme des intérêts perçus entre le 31 décembre 2014 et le 31 décembre 2024 inclus est la différence entre le capital acquis le 1er janvier 2025 et le capital de départ, c'est-à-dire :

$$C_{11} - C_0 = 10\,000 \times 1{,}03^{11} - 10\,000 \approx 3\,842{,}34 \text{ €}$$

2.32 a) $u_1 = 150 + 10 \times 2^{-1+2} = 150 + 10 \times 2 = 170$. Le contrat le plus avantageux le premier mois est celui du grossiste B car $u_1 > 151$.

$u_{24} = 150 + 10 \times 2^{-24+2} = 150 + 10 \times 2^{-22} \approx 150$ (à 10^{-2} près). Le contrat le plus avantageux le 24ème mois est celui du grossiste A car $u_{24} < 151$.

b) Le grossiste A devient plus avantageux que le grossiste B lorsque $u_n < 151$.

$$u_n < 151 \Leftrightarrow 150 + 10 \times 2^{-n+2} < 151 \Leftrightarrow 10 \times 2^{-n+2} < 1 \Leftrightarrow 2^{-n+2} < 0{,}1$$
$$\Leftrightarrow 4/2^n < 0{,}1 \Leftrightarrow 2^n > 4/0{,}1 = 40$$

Puisque $2 > 1$, la suite de terme général 2^n est croissante.
De plus $2^5 = 32 < 40$ et $2^6 = 64 > 40$. Le grossiste A devient donc plus avantageux que le grossiste B à partir du 6ème mois.

c) $10 \times 2^{-n+2} = 10 \times 2^2 \times (2^{-1})^n = 40 \times 0{,}5^n$. Puisque $0 < 0{,}5 < 1$ alors $\lim\limits_{n\to\infty}\left(40\times 0{,}5^n\right)=0$, d'où $\lim\limits_{n\to\infty} u_n = 150+0=150$. Le prix du matériel finira par se stabiliser vers 150 €.

d) Coût du grossiste A :

$$
\begin{aligned}
&10u_1 + 10u_2 + \ldots + 10u_{24}\\
&=10\left(\left(150+40\times 0{,}5^1\right)+\left(150+40\times 0{,}5^2\right)+\ldots+\left(150+40\times 0{,}5^{24}\right)\right)\\
&=\left(1500+400\times 0{,}5^1\right)+\left(1500+400\times 0{,}5^2\right)+\ldots+\left(1500+400\times 0{,}5^{24}\right)\\
&=24\times 1500+400\times\left(0{,}5^1+0{,}5^2+\ldots+0{,}5^{24}\right)\\
&=3600+400\times 0{,}5\times\left(1-0{,}5^{24}\right)/\left(1-0{,}5\right)\\
&\approx 36\,400\text{ € (au centime près)}
\end{aligned}
$$

Coût du grossiste B : $24 \times 10 \times 151 = 36\,240$ €

Pour une commande de 10 tablettes par mois pendant deux ans, le grossiste B est plus avantageux.

CALCUL MATRICIEL 3

PLAN

3.1 Généralités
3.2 Calcul matriciel élémentaire
3.3 Inverse d'une matrice carrée
3.4 Résolution d'un système à l'aide de matrices

OBJECTIFS

- Comprendre la notion de matrice.
- Maîtriser le calcul matriciel élémentaire.
- Maîtriser la notion d'inverse.
- Savoir résoudre un système de *n* équations à *n* inconnues.

3.1 GÉNÉRALITÉS

Une **matrice** A, à n lignes et p colonnes est un tableau de la forme :

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,p} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,p} \end{pmatrix}$$

où $a_{i,j}$ désigne le nombre de la $i^{\text{ème}}$ ligne et $j^{\text{ème}}$ colonne. On peut noter plus simplement $A = \left(a_{i,j}\right)$.

- Si $n = p$, on parle de matrice carrée d'ordre n.
- Si $n = 1$, on parle de matrice ligne.
- Si $p = 1$, on parle de matrice colonne ou de vecteur colonne.

Exemple

On considère les matrices suivantes :

$$A = \begin{pmatrix} 2 & 6 & 8 \\ 4 & 5 & 1 \\ 0 & 2 & 10 \end{pmatrix} \qquad B = \begin{pmatrix} 7 & 8 & 2 & 4 \end{pmatrix} \qquad C = \begin{pmatrix} 5 \\ -3 \\ 4 \end{pmatrix}$$

A est une matrice carrée d'ordre 3. Si on note $A = (a_{i,j})$ alors $a_{2,1} = 4$, $a_{1,2} = 6$ et $a_{3,3} = 10$.
B est une matrice ligne et C est une matrice colonne.

Pour que deux matrices $A = (a_{i,j})$ et $B = (b_{i,j})$ soient **égales**, elles doivent être de même taille (même nombre de lignes et même nombre de colonnes) et avoir les mêmes éléments (c'est-à-dire, pour tous i et j, $a_{i,j} = b_{i,j}$). Par exemple :

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \neq \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

On appelle **matrice nulle** toute matrice dont tous les éléments sont nuls.

On appelle **matrice identité d'ordre *n***, la matrice carrée d'ordre n, notée I_n et définie par :

$$I_n = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

I_n est donc la matrice telle que $a_{i,j} = 1$ si $i = j$ et $a_{i,j} = 0$ si $i \neq j$.

Exemple

$\begin{pmatrix} 0 & 0 & 0 \end{pmatrix}$, $\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ et $\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$ sont des matrices nulles.

$I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ et $I_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ sont les matrices identité d'ordres 2 et 3.

Une calculatrice graphique effectue le calcul matriciel. Pour définir une matrice sur une calculatrice Casio (Graph 35-65), sélectionner le menu RUN-MAT puis MAT(F1), ensuite le nom de la matrice (Mat A, Mat B, ..., Mat Z), ses dimensions (son nombre de lignes *m* et son nombre de colonnes *n*), et enfin ses éléments en validant chacun d'eux par EXE. Sur une calculatrice TI (82, 83, 85), appuyer sur MATRX, mettre en surbrillance EDIT, sélectionner le nom de la matrice (1:[A], 2:[B], ..., 5:[E]), valider, rentrer les dimensions de la matrice (nombre de lignes X nombre de colonnes), puis saisir les éléments en validant chacun d'eux par ENTER.

3.2 Calcul matriciel élémentaire

Soient $A = (a_{i,j})$ et $B = (b_{i,j})$ deux matrices à *n* lignes et *p* colonnes. La somme des matrices A et B est la matrice $C = (c_{i,j})$ à *n* lignes et *p* colonnes telle que pour tous *i* et *j* on ait $c_{i,j} = a_{i,j} + b_{i,j}$. On peut alors noter $C = A+B$.

Exemple

Si $A=\begin{pmatrix}3 & 1 & 0\\ 4 & -5 & 6\end{pmatrix}$ et $B=\begin{pmatrix}1 & -2 & 4\\ 8 & 2 & -7\end{pmatrix}$,

alors $A+B=\begin{pmatrix}3+1 & 1-2 & 0+4\\ 4+8 & -5+2 & 6-7\end{pmatrix}=\begin{pmatrix}4 & -1 & 4\\ 12 & -3 & -1\end{pmatrix}$.

Pour effectuer cette opération sur Casio, on tape simplement Mat A + Mat B puis EXE dans le menu RUN (Mat s'obtient directement avec SHIFT puis la touche 2). Sur TI, appuyer sur MATRX puis 1:[A], taper sur +, puis de nouveau MATRX, sélectionner 2:[B] et enfin valider le calcul par ENTER.

Propriété 3.1

Si A est une matrice et O une matrice nulle de même taille, alors $A+O=O+A=A$ (on dit qu'une matrice nulle est *élément neutre* pour l'addition matricielle).

Pour toutes matrices A, B et C de même taille, on a : $A+B=B+A$ et $A+(B+C)=(A+B)+C$, que l'on peut alors noter $A+B+C$.

Exemple

$$\begin{pmatrix}0\\ 0\end{pmatrix}+\begin{pmatrix}3\\ -1\end{pmatrix}=\begin{pmatrix}3\\ -1\end{pmatrix}+\begin{pmatrix}0\\ 0\end{pmatrix}=\begin{pmatrix}3\\ -1\end{pmatrix}$$

$$\begin{pmatrix}8 & -5\\ 1 & 2\end{pmatrix}+\begin{pmatrix}-3 & 2\\ 1 & 5\end{pmatrix}=\begin{pmatrix}-3 & 2\\ 1 & 5\end{pmatrix}+\begin{pmatrix}8 & -5\\ 1 & 2\end{pmatrix}=\begin{pmatrix}5 & -3\\ 2 & 7\end{pmatrix}$$

$$\begin{pmatrix}2 & -6\\ 3 & 0\end{pmatrix}+\begin{pmatrix}4 & 1\\ -1 & 5\end{pmatrix}+\begin{pmatrix}-3 & 1\\ 8 & -1\end{pmatrix}=\begin{pmatrix}3 & -4\\ 10 & 4\end{pmatrix}$$

Le **produit** d'une matrice $A=(a_{i,j})$ à n lignes et p colonnes par un nombre réel k est la matrice $B=(b_{i,j})$ à n lignes et p colonnes telle que pour tous i et j on ait $b_{i,j}=k\times a_{i,j}$. On note $B=kA$.

La matrice $(-1)A$ se note plus simplement $-A$ et est appelée **opposée** de A.

Exemple

Si $A=\begin{pmatrix}3 & 1 & 0\\ 4 & -5 & 6\end{pmatrix}$ et $k=2$,

alors $kA=2A=\begin{pmatrix}2\times 3 & 2\times 1 & 2\times 0\\ 2\times 4 & 2\times(-5) & 2\times 6\end{pmatrix}=\begin{pmatrix}6 & 2 & 0\\ 8 & -10 & 12\end{pmatrix}$.

Pour effectuer cette opération sur Casio, on tape 2*Mat A puis EXE dans le menu RUN. Sur TI, taper 2* puis appuyer sur MATRX, sélectionner 1:[A] puis valider le calcul par ENTER.

Soient $A = (a_{i,j})$ une matrice à n lignes et p colonnes et $B = \left(b_{i,j}\right)$ une matrice à p lignes et m colonnes. Le **produit** des matrices A et B est la matrice $C = \left(c_{i,j}\right)$ à n lignes et m colonnes telle que pour tous i et j on ait $c_{i,j} = a_{i,1} \times b_{1,j} + a_{i,2} \times b_{2,j} + ... + a_{i,p} \times b_{p,j}$. On peut alors noter $C = A \times B$ ou même $C = AB$.

Pour que l'on puisse effectuer le produit de A par B, il faut que le nombre de colonnes de A soit égal au nombre de lignes de B.

Exemple

On pose :

$$A = \begin{pmatrix} 2 & 0 \\ 5 & 3 \\ 1 & 2 \end{pmatrix} \quad \text{et} \quad B = \begin{pmatrix} 6 & 3 \\ 2 & 4 \end{pmatrix}$$

A a 3 lignes et 2 colonnes, *B* a 2 lignes et 2 colonnes. Le produit de *A* par *B* est donc une matrice à 3 lignes (nombre de lignes de *A*) et 2 colonnes (nombre de colonnes de *B*).

Calculons les éléments de la matrice *C* = *A* x *B* :

$$c_{1,1} = a_{1,1} \times b_{1,1} + a_{1,2} \times b_{2,1} = 2 \times 6 + 0 \times 2 = 12$$

(on a multiplié entre eux les nombres de la première ligne de *A* avec les nombres de la première colonne de *B*.)

$$c_{1,2} = a_{1,1} \times b_{1,2} + a_{1,2} \times b_{2,2} = 2 \times 3 + 0 \times 4 = 6$$

(on a multiplié entre eux les nombres de la première ligne de *A* avec les nombres de la deuxième colonne de *B*), etc.

On obtient alors :

$$A \times B = \begin{pmatrix} 2\times6+0\times2 & 2\times3+0\times4 \\ 5\times6+3\times2 & 5\times3+3\times4 \\ 1\times6+2\times2 & 1\times3+2\times4 \end{pmatrix} = \begin{pmatrix} 12 & 6 \\ 36 & 27 \\ 10 & 11 \end{pmatrix}$$

Si $A = \begin{pmatrix} 10 & 7 \\ 6 & 5 \end{pmatrix}$ et $B = \begin{pmatrix} 3 \\ -4 \end{pmatrix}$ alors $A \times B = \begin{pmatrix} 10\times3+7\times(-4) \\ 6\times3+5\times(-4) \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \end{pmatrix}$.

Pour effectuer cette opération sur Casio, on tape Mat A * Mat B puis EXE dans le menu RUN. Sur TI, appuyer sur MATRX puis 1:[A], taper sur X, puis de nouveau MATRX, sélectionner 2:[B] et enfin valider le calcul par ENTER.

Exemple d'application 3.1

Une société commerciale, AZER, possède deux magasins dont l'aménagement du parc informatique est le suivant :

*– **Magasin 1** : 12 PC, 5 tablettes, 10 mobiles.*

*– **Magasin 2** : 17 PC, 6 tablettes, 14 mobiles.*

On peut associer à cet équipement la matrice suivante :

$$\begin{pmatrix} 12 & 5 & 10 \\ 17 & 6 & 14 \end{pmatrix}$$

La société AZER souhaite améliorer son équipement de la façon suivante :

*– **Magasin 1** : + 3 PC, + 2 tablettes, + 2 mobiles.*

*– **Magasin 2** : + 5 PC, + 3 tablettes, + 4 mobiles.*

Ce nouvel équipement peut être associé à la matrice :

$$\begin{pmatrix} 3 & 2 & 2 \\ 5 & 3 & 4 \end{pmatrix}$$

1) Quelle opération matricielle faut-il effectuer pour déterminer le nouvel aménagement du parc informatique des deux magasins ?
Effectuer cette opération.

2) Pour acheter le nouvel équipement, la société AZER a le choix entre deux fournisseurs dont les prix sont donnés dans le tableau 3.1

Tableau 3.1

	Fournisseur 1	Fournisseur 2
PC	600,00 €	550,00 €
Tablette	180,00 €	200,00 €
Mobile	60,00 €	50,00 €

On peut associer ces prix à la matrice :

$$\begin{pmatrix} 600 & 550 \\ 180 & 200 \\ 60 & 50 \end{pmatrix}$$

Quelle opération matricielle faut-il effectuer pour déterminer le prix de l'aménagement du nouvel équipement selon le magasin et le fournisseur ?
Effectuer cette opération.

Solution. 1) On obtient alors la nouvelle répartition de l'équipement des magasins en opérant ainsi :

$$\begin{pmatrix} 12 & 5 & 10 \\ 17 & 6 & 14 \end{pmatrix} + \begin{pmatrix} 3 & 2 & 2 \\ 5 & 3 & 4 \end{pmatrix} = \begin{pmatrix} 15 & 7 & 12 \\ 22 & 9 & 18 \end{pmatrix}$$

2) On obtient les prix de l'aménagement selon les magasins et les fournisseurs en opérant comme sur la fig. 3.1.

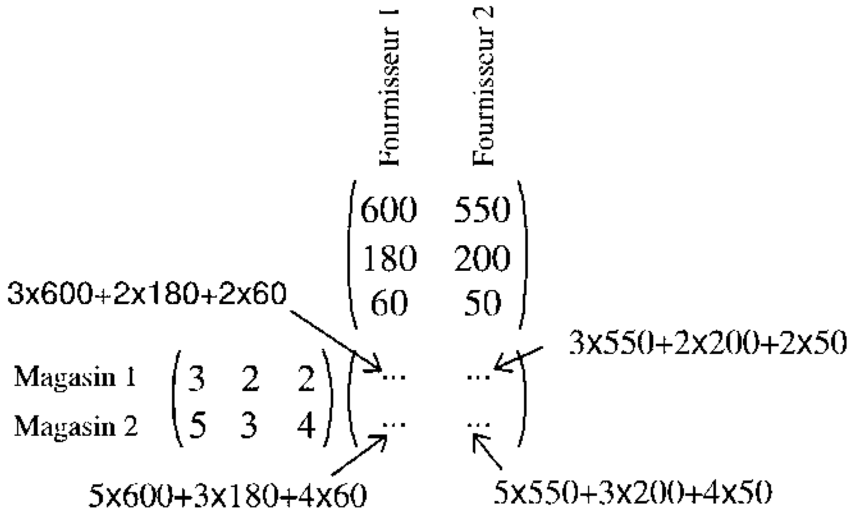


Figure 3.1

Il faut donc multiplier des matrices, et on obtient :

$$\begin{pmatrix} 3 & 2 & 2 \\ 5 & 3 & 4 \end{pmatrix} \times \begin{pmatrix} 600 & 550 \\ 180 & 200 \\ 60 & 50 \end{pmatrix} = \begin{pmatrix} 2280 & 2150 \\ 3780 & 3550 \end{pmatrix}$$

Par exemple le prix de l'investissement pour le magasin 1 est de 2 280 € avec le fournisseur 1 et de 2 150 € avec le fournisseur 2.

Si A est une matrice carrée d'ordre n alors $A \times I_n = I_n \times A = A$. On dit qu'une matrice identité est **élément neutre** pour la multiplication.

En général, on n'a pas $A \times B = B \times A$. De plus, même si un produit $A \times B$ de matrices est défini, $B \times A$ ne l'est pas forcément.

Exemple

Si $A = \begin{pmatrix} 10 & 7 \\ 6 & 5 \end{pmatrix}$ et $B = \begin{pmatrix} 6 & 3 \\ 2 & 4 \end{pmatrix}$, alors $A \times B = \begin{pmatrix} 74 & 58 \\ 46 & 38 \end{pmatrix}$ et $B \times A = \begin{pmatrix} 78 & 57 \\ 44 & 34 \end{pmatrix}$.

Propriété 3.2

Si A, B et C sont des matrices et k un nombre réel, on a les propriétés suivantes :

$(A\times B)\times C = A\times(B\times C)$ que l'on peut alors noter $A\times B\times C$

$A\times(B+C)=A\times B+A\times C$

$A\times(kB)=(kA)\times B=kA\times B$

Si A est une matrice carrée, on note $A^2, A^3, ..., A^n$ (pour $n\geq 2$), les matrices définies par : $A^2=A\times A$, $A^3=A\times A\times A, ..., A^n=A^{n-1}\times A=A\times A^{n-1}$.

Exemple

Si $A=\begin{pmatrix}3 & -1\\ 1 & 2\end{pmatrix}$ alors $A^2=\begin{pmatrix}8 & -5\\ 5 & 3\end{pmatrix}$

et $A^3=A^2\times A=\begin{pmatrix}8 & -5\\ 5 & 3\end{pmatrix}\times\begin{pmatrix}3 & -1\\ 1 & 2\end{pmatrix}=\begin{pmatrix}19 & -18\\ 18 & 1\end{pmatrix}$.

Pour effectuer cette opération sur Casio, on tape Mat A ^2 puis EXE dans le menu RUN. Sur TI, appuyer sur MATRX puis 1:[A], puis taper ^2 et valider par ENTER.

Si A et B sont deux matrices de même taille, la **différence** des matrices A et B, que l'on note $A-B$, est la matrice $A+(-1)\times B$.

Exemple

Si $A=\begin{pmatrix}5 & 0 & -8\\ 3 & 4 & 11\end{pmatrix}$ et $B=\begin{pmatrix}2 & 6 & 3\\ -4 & 4 & 1\end{pmatrix}$

alors $A-B=\begin{pmatrix}5-2 & 0-6 & -8-3\\ 3-(-4) & 4-4 & 11-1\end{pmatrix}=\begin{pmatrix}3 & -6 & -11\\ 7 & 0 & 10\end{pmatrix}$.

Si $A=\begin{pmatrix}3 & -1\\ 1 & 2\end{pmatrix}$ et $B=\begin{pmatrix}-1 & 2\\ 0 & 4\end{pmatrix}$ alors $4A-A\times B=\begin{pmatrix}12 & -4\\ 4 & 8\end{pmatrix}-\begin{pmatrix}-3 & 2\\ -1 & 10\end{pmatrix}=\begin{pmatrix}15 & -6\\ 5 & -2\end{pmatrix}$.

Pour effectuer cette opération sur Casio, on tape Mat A – Mat B puis EXE dans le menu RUN. Sur TI, appuyer sur MATRX puis 1:[A], taper sur –, puis de nouveau MATRX, sélectionner 2:[B] et enfin valider le calcul par ENTER.

3.3 Inverse d'une matrice carrée

Considérons les matrices $A=\begin{pmatrix}1 & 1\\ 2 & 4\end{pmatrix}$ et $B=\begin{pmatrix}2 & -0,5\\ -1 & 0,5\end{pmatrix}$.

$$A\times B=\begin{pmatrix}1 & 1\\ 2 & 4\end{pmatrix}\times\begin{pmatrix}2 & -0,5\\ -1 & 0,5\end{pmatrix}=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$$

$$B\times A=\begin{pmatrix}2 & -0,5\\ -1 & 0,5\end{pmatrix}\times\begin{pmatrix}1 & 1\\ 2 & 4\end{pmatrix}=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$$

On constate que $A\times B=B\times A=I_2$. Pour la matrice *A*, on a trouvé une matrice *B* telle que les produits $A\times B$ et $B\times A$ soient égaux à la matrice identité. On dit que la matrice *B* est l'**inverse** de *A*, que l'on peut noter A^{-1}. La matrice *A* est dite **inversible**.

Définition 3.1

Soit *A* une matrice carrée d'ordre *n*. S'il existe une matrice carrée *B* d'ordre *n* telle que $A\times B=B\times A=I_n$, alors on dit que la matrice *A* est **inversible** et que sa **matrice inverse** est *B*, notée A^{-1}.

Pour effectuer cette opération sur Casio, on tape Mat A ^(-1) puis EXE dans le menu RUN. Sur TI, appuyer sur MATRX puis 1:[A], puis taper ^(-1) et valider par ENTER.

Si $A\times B=B\times A=I_n$, alors *B* est l'inverse de *A* mais on peut aussi dire que *A* est l'inverse de *B*.

Quand elle existe, l'inverse d'une matrice est unique. En effet, supposons que *B* et B' soient inverses de *A*. Calculons le produit $B\times A\times B'$ de deux façons :

$$B\times A\times B'=(B\times A)\times B'=I_n\times B'=B'$$

$$\text{et } B\times A\times B'=B\times(A\times B')=B\times I_n=B$$

Cela prouve que $B=B'$ et par conséquent l'unicité de l'inverse.

En pratique, pour vérifier qu'une matrice *B* est l'inverse d'une matrice *A*, un seul calcul suffit : $A\times B$ ou $B\times A$.

Exemple

Soient $A=\begin{pmatrix}0 & 2 & -1\\ 1 & 1 & 1\\ 0 & 1 & 2\end{pmatrix}$ et $B=\begin{pmatrix}-0,2 & 1 & -0,6\\ 0,4 & 0 & 0,2\\ -0,2 & 0 & 0,4\end{pmatrix}$; $A\times B=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}=I_3$

donc *B* est l'inverse de *A*.

3.4 Résolution de systèmes à l'aide de matrices

Exemple introductif

Considérons le système (S) de deux équations à deux inconnues : $\begin{cases} 2x+3y=15 \\ 3x+4y=12 \end{cases}$

Si on pose $A=\begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix}$, $X=\begin{pmatrix} x \\ y \end{pmatrix}$ et $B=\begin{pmatrix} 15 \\ 12 \end{pmatrix}$, alors $AX=\begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix}\times\begin{pmatrix} x \\ y \end{pmatrix}=\begin{pmatrix} 2x+3y \\ 3x+4y \end{pmatrix}$.

Ainsi le système peut s'écrire $AX = B$. On a écrit le système (S) sous forme matricielle.

Théorème 3.1

Si un système de n équations à n inconnues peut s'écrire sous la forme matricielle $AX = B$ et si A est inversible, alors ce système possède une unique solution X donnée par $X = A^{-1} \times B$.

En effet, si $AX = B$ et si A est inversible, alors :

$$A^{-1} \times AX = A^{-1} \times B$$
$$\left(A^{-1} \times A\right) X = A^{-1} \times B$$
$$I \times X = A^{-1} \times B$$
$$X = A^{-1} \times B$$

Exemple

Reprenons le système (S) de l'exemple introductif. La matrice A est inversible et, avec une calculatrice, on trouve $A^{-1}=\begin{pmatrix} -4 & 3 \\ 3 & -2 \end{pmatrix}$.

Par conséquent, $X = A^{-1} \times B = \begin{pmatrix} -4 & 3 \\ 3 & -2 \end{pmatrix}\begin{pmatrix} 15 \\ 12 \end{pmatrix}=\begin{pmatrix} -24 \\ 21 \end{pmatrix}$.

Donc $x = -24$ et $y = 21$. La solution du système (S) est (–24 ; 21).

Le système $\begin{cases} x+2z=9 \\ y-3z=-4 \\ x+2y=5 \end{cases}$ s'écrit $AX=B$ avec : $A=\begin{pmatrix} 1 & 0 & 2 \\ 0 & 1 & -3 \\ 1 & 2 & 0 \end{pmatrix}$, $X=\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ et $B=\begin{pmatrix} 9 \\ -4 \\ 5 \end{pmatrix}$.

À la calculatrice, $A^{-1}=\begin{pmatrix} 1,5 & 1 & -0,5 \\ -0,75 & -0,5 & 0,75 \\ -0,25 & -0,5 & 0,25 \end{pmatrix}$ et $X=A^{-1}\times B=\begin{pmatrix} 7 \\ -1 \\ 1 \end{pmatrix}$.

Donc $x = 7$, $y = -1$ et $z = 1$. La solution du système est (7 ; –1 ; 1).

Exercices corrigés

3.1 On considère les matrices $A=(a_{i,j})$ et $B=(b_{i,j})$, carrées d'ordre 3, définies par $a_{i,j}=i$ et $b_{i,j}=i-j$. Écrire ces matrices sous la forme de tableaux de nombres.

3.2 Soit $A=\left(a_{i,j}\right)$ la matrice à 3 lignes et 4 colonnes telle que $a_{i,j}=Max(i,j)$ et $B=\left(b_{i,j}\right)$ la matrice à 2 lignes et 5 colonnes telle que $b_{i,j}=i\times j$. Écrire les matrices A et B sous la forme de tableaux de nombres.

3.3 On considère les matrices $A=\begin{pmatrix}-1 & 2\\ 1 & 0\end{pmatrix}$ et $B=\begin{pmatrix}3 & 1\\ 5 & 10\end{pmatrix}$.

a) Calculer $A+B$ et $A-B$.

b) Calculer $(A+B)+(A-B)$. Expliquer le résultat.

3.4 On considère la matrice $A=\begin{pmatrix}0{,}3 & 1\\ 4 & -2\end{pmatrix}$.

a) Déterminer $A+A$, $3A$ et $-10A$.

b) Est-il possible de trouver un nombre réel x tel que $xA = I_2$? (Justifier)

3.5 Soit A la matrice définie par $A=\begin{pmatrix}10 & 20\\ 8 & 100\end{pmatrix}$. Déterminer le plus petit entier naturel non nul n tel que tous les éléments de $1/n \times A$ soient strictement inférieurs à 0,01.

3.6 Soient $A=\begin{pmatrix}7 & 0 & -1\\ 4 & -2 & -2\end{pmatrix}$ et $B=\begin{pmatrix}-9 & 1 & 3\\ 0 & -6 & -5\end{pmatrix}$.

a) Calculer $2A-B$ et $3A-4B$.

b) Déterminer les deux matrices C et D telles que $C+D=A$ et $C-D=B$.

3.7 Un groupe pharmaceutique produit trois types de médicaments M1, M2 et M3 dans chacun de ses deux laboratoires L1 et L2. On associe la matrice A suivante à la production mensuelle (en milliers) de ces médicaments dans chaque laboratoire.

$$A=\begin{pmatrix}54 & 27 & 31\\ 12 & 9 & 10\end{pmatrix}$$

Par exemple, le groupe produit chaque mois 12 000 médicaments du type M1 dans le laboratoire L2.

a) Combien le laboratoire L1 produit-il de médicaments du type M3 par mois ?

b) On peut obtenir, à partir d'une matrice colonne B, le nombre total de médicaments produits par chaque laboratoire en effectuant le calcul $A \times B$. De quelle matrice colonne s'agit-il ?

3.8 On considère les matrices $A = \begin{pmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{pmatrix}$ et $B = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$.

a) Calculer $A \times B$ puis $A \times (2B)$.

b) Vérifier que $A \times (2B) = 2(A \times B)$.

3.9 On considère les matrices $A = \begin{pmatrix} -1 & 0 \\ 5 & 1 \\ 3 & -6 \end{pmatrix}$, $B = \begin{pmatrix} 5 & -5 \\ 1 & 4 \end{pmatrix}$ et $C = \begin{pmatrix} 0 & 3 \\ 1 & 1 \end{pmatrix}$.

a) Calculer $A \times B$ et $A \times C$.

b) Peut-on calculer $B \times A$ et $C \times A$? (Justifier)

c) Calculer $B + C$ puis $A \times (B + C)$.

d) Calculer $A \times (B + C) - A \times B - A \times C$. Expliquer le résultat.

3.10 On considère les matrices $A = \begin{pmatrix} 1 & 5 \\ -3 & 2 \end{pmatrix}$, $B = \begin{pmatrix} 4 \\ 1 \end{pmatrix}$ et $C = \begin{pmatrix} 1 & 3 \\ 5 & -2 \\ 0 & 1 \end{pmatrix}$.

Faire tous les produits possibles de deux matrices.

3.11 On considère les matrices $A = \begin{pmatrix} 7 \\ 5 \\ -2 \end{pmatrix}$, $B = \begin{pmatrix} -1 & -1 & 2 \\ -6 & 0 & 9 \\ 4 & 3 & -4 \end{pmatrix}$ et $C = \begin{pmatrix} 1 & -1 \\ 2 & 1 \\ 0 & 4 \end{pmatrix}$.

Faire tous les produits possibles de deux matrices.

3.12 On considère les matrices $A = \begin{pmatrix} 5 & 2 \\ 3 & 1 \end{pmatrix}$, $B = \begin{pmatrix} -1 & 2 \\ 3 & -5 \end{pmatrix}$ et $C = \begin{pmatrix} 2 & 0 \\ 0 & -1 \end{pmatrix}$.

Calculer $A \times B$, $B \times A$ et $A \times C \times B$.

3.13 On considère les matrices $A = \begin{pmatrix} 5 & 2 \\ 1 & 2 \end{pmatrix}$ et $B = \begin{pmatrix} -2 & 1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & -2 \end{pmatrix}$.

a) Calculer A^2 puis A^3.

b) Calculer B^2 puis B^3.

3.14 Soit $B = \begin{pmatrix} 0 & 1 & 5 \\ 0 & 0 & 6 \\ 0 & 0 & 0 \end{pmatrix}$

a) Calculer B^2 et B^3.

b) Montrer que $B^4 = 0$ (matrice nulle) puis que, pour tout $n \geq 3$, $B^n = 0$.

3.15 Une usine fabrique trois types de matériel électronique M1, M2 et M3 en assemblant des composants C1, C2 et C3 suivant la répartition du tableau 3.1.

Tableau 3.1

	M1	M2	M3
Nombre de composants C1	2	7	3
Nombre de composants C2	0	2	6
Nombre de composants C3	3	5	2

Les masses et prix unitaires des composants sont donnés par le tableau 3.2.

Tableau 3.2

	C1	C2	C3
Masse unitaire (en g)	2	1	3
Prix unitaire (en €)	15	6	4

On note $A = \begin{pmatrix} 2 & 7 & 3 \\ 0 & 2 & 6 \\ 3 & 5 & 2 \end{pmatrix}$ et $B = \begin{pmatrix} 2 & 1 & 3 \\ 15 & 6 & 4 \end{pmatrix}$.

a) Calculer $B \times A$ et interpréter ce que représentent les lignes de la matrice obtenue.

b) Le directeur de l'usine souhaite fabriquer en une journée 50 matériels M1, 70 matériels M2 et 60 matériels M3.

On pose $D = \begin{pmatrix} 50 \\ 70 \\ 60 \end{pmatrix}$.

Quelle opération matricielle permet d'obtenir le nombre de composants de chaque sorte permettant de réaliser les assemblages ? Effectuer l'opération puis donner la décomposition.

3.16 On considère les matrices $A=\begin{pmatrix}2 & 5\\ 1 & 3\end{pmatrix}$ et $B=\begin{pmatrix}3 & -5\\ -1 & 2\end{pmatrix}$.

Calculer $A\times B$ et $B\times A$. Que peut-on dire des matrices A et B ?

3.17 On considère les matrices $A=\begin{pmatrix}2 & 0 & 1\\ 0 & -1 & 2\\ 9 & 1 & 0\end{pmatrix}$ et $A'=\begin{pmatrix}-0,4 & 0,2 & 0,2\\ 3,6 & -1,8 & -0,8\\ 1,8 & -0,4 & -0,4\end{pmatrix}$.

Montrer que A' est l'inverse de A.

3.18 On considère les matrices $A=\begin{pmatrix}0 & 1 & -1\\ -3 & 4 & -3\\ -1 & 1 & 0\end{pmatrix}$ et $B=\begin{pmatrix}3 & -1 & 1\\ 3 & -1 & 3\\ 1 & -1 & 3\end{pmatrix}$.

Calculer $A\times B$. Quelle est la matrice inverse de A ?

3.19 On considère les matrices $A=\begin{pmatrix}3 & 1 & -2\\ 0 & -1 & -1\\ 2 & 1 & 2\end{pmatrix}$ et $B=\begin{pmatrix}1 & 4 & 1\\ 2 & -10 & -1\\ -2 & 1 & 1\end{pmatrix}$.

Calculer $A\times B$. Quelle est la matrice inverse de A ?

3.20 On considère les matrices $A=\begin{pmatrix}7 & 5\\ -4 & -2\end{pmatrix}$ et $B=\begin{pmatrix}-1 & -2,5\\ 2 & 3,5\end{pmatrix}$.

Calculer $A\times B$ et $B\times A$. En déduire la matrice A^{-1}.

3.21 On considère les matrices $A=\begin{pmatrix}5 & 1 & 4\\ 2 & 1 & -3\\ -3 & -1 & 1\end{pmatrix}$ et $B=\begin{pmatrix}a & -5 & -7\\ 7 & 17 & 23\\ 1 & 2 & 3\end{pmatrix}$.

a) Dans le produit $A\times B$, calculer, en fonction de a, les coefficients manquants :

$$A\times B=\begin{pmatrix}\ldots\ldots\ldots\ldots & 0 & 0\\ \ldots\ldots\ldots\ldots & 1 & 0\\ \ldots\ldots\ldots\ldots & 0 & 1\end{pmatrix}$$

b) Déterminer alors la valeur de a pour que B soit la matrice inverse de A.

3.22 On considère les matrices $A=\begin{pmatrix}2 & 1 & 3\\ 2 & 0 & -1\\ 1 & 1 & 5\end{pmatrix}$ et $B=\frac{1}{3}\begin{pmatrix}-1 & 2 & 1\\ 11 & a & -8\\ -2 & 1 & 2\end{pmatrix}$.

a) Calculer le produit $A\times B$.

b) Déterminer alors la valeur de a pour que B soit la matrice inverse de A.

3.23 Soit $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. On suppose que $ad - bc \neq 0$.

a) Montrer que la matrice $A' = \dfrac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$ est la matrice inverse de A.

b) **Application** : déterminer la matrice inverse de $A = \begin{pmatrix} 7 & 3 \\ 12 & 8 \end{pmatrix}$.

3.24 On donne $A = \begin{pmatrix} 3 & 1 \\ 4 & 1 \end{pmatrix}$.

a) Calculer $A^2 - 4A$.

b) Montrer que $A^2 - 4A = A \times (A - 4I_2)$ et en déduire que A est inversible.
Quelle est sa matrice inverse ?

3.25 Soit $A = \begin{pmatrix} -2 & -3 \\ 5 & 7 \end{pmatrix}$.

a) Calculer $A^2 - 5A$.

b) Montrer que A est inversible et déterminer A^{-1}.

3.26 Soit $A = \begin{pmatrix} 13 & -5 \\ -6 & 2 \end{pmatrix}$

a) Calculer $A^2 - 15A$

b) Montrer que A est inversible et déterminer A^{-1}.

3.27 a) Écrire le système $\begin{cases} 2x - 3y = 5 \\ -3x + 5y = -2 \end{cases}$ sous la forme matricielle $AX = B$.

Déterminer A^{-1} à la calculatrice et résoudre le système.

b) Résoudre le système $\begin{cases} 7x + 16y = 31 \\ 5x + 12y = 25 \end{cases}$ à l'aide de matrices.

c) Résoudre le système $\begin{cases} 5x - 3z = 4 \\ 3x + 4y + z = 2 \\ x - y - z = 2 \end{cases}$ à l'aide de matrices.

3.28 On considère le système S d'inconnues x, y et z : $\begin{cases} -4x+y+0{,}1z=-1 \\ 11x-3y-0{,}2z=1 \\ -6x+2y+0{,}1z=4 \end{cases}$

a) Écrire le système sous la forme $AX = B$ en précisant quelles sont les matrices A, X et B.

b) Avec une calculatrice, on a obtenu $A^{-1} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 40 & 20 & 10 \end{pmatrix}$. Résoudre le système.

3.29 Résoudre chaque système à l'aide des matrices :

a) $\begin{cases} x+y+z=23 \\ 2x+3y-z=27 \\ -x+y+4z=27 \end{cases}$ b) $\begin{cases} 5x+y+2z=25 \\ 2x-y-z=-7 \\ x+3y+z=27 \end{cases}$ c) $\begin{cases} 6x+y+3z=5 \\ 2x+z=0 \\ -x+3y-z=16 \end{cases}$

3.30 a, b et c étant trois réels, on considère le système : $\begin{cases} x+y+z=a \\ 2x+y+3z=b \\ x-y+z=c \end{cases}$

1) a) Écrire le système sous forme matricielle.
b) Exprimer x, y et z en fonction de a, b et c.

2) Utiliser les résultats de la question a) pour résoudre les systèmes suivants :

(S_1) $\begin{cases} x+y+z=12 \\ 2x+y+3z=25 \\ x-y+z=10 \end{cases}$ (S_2) $\begin{cases} x+y+z=7 \\ 2x+y+3z=18 \\ x-y+z=9 \end{cases}$ (S_3) $\begin{cases} 3x+3y+3z=27 \\ 6x+3y+9z=6 \\ 3x-3y+3z=15 \end{cases}$

3.31 Au magasin de fournitures de bureau :
– 4 stylos, 2 gommes et 3 pochettes coûtent 10,70 euros.
– 1 stylo, 1 gomme et 5 pochettes coûtent 7,80 euros.
– 2 stylos, 1 gomme et 4 pochettes coûtent 8,10 euros.

a) Traduire l'énoncé par un système puis l'écrire à l'aide de matrices.

b) Combien coûte un stylo ? une gomme ? une pochette ?

3.32 Pour une fabrication, une entreprise utilise x pièces de type 1, y pièces de type 2 et z pièces de type 3. La masse et le coût de chaque type de pièce sont donnés dans le tableau 3.3.

Tableau 3.3

	Pièce de type 1	Pièce de type 2	Pièce de type 3
Masse en grammes	2,5	2	1
Coût en euros	1	1,5	0,5

L'entreprise effectue une étude en vue d'optimiser cette fabrication. Pour cela, elle doit considérer le nombre total N de pièces employées, leur masse totale M en grammes et leur coût total C en euros.

1) Calculer N, M et C lorsque $x = 20$, $y = 30$ et $z = 35$

2) Exprimer N, M et C en fonction de x, y et z.

3) On se propose de résoudre le système $\begin{cases} x + y + z = N \\ 2{,}5x + 2y + z = M \\ x + 1{,}5y + 0{,}5z = C \end{cases}$

dont les inconnues sont x, y et z.

a) Soit $A = \begin{pmatrix} 1 & 1 & 1 \\ 2{,}5 & 2 & 1 \\ 1 & 1{,}5 & 0{,}5 \end{pmatrix}$. Avec la calculatrice, déterminer A^{-1}.

b) En déduire la solution du système en fonction de N, M et C.

4) L'étude a montré que la fabrication est optimale lorsque sont employées 140 pièces, pour une masse totale de 275 g et un coût total de 135 euros. Dans ces conditions, calculer le nombre de pièces de chaque type qui seront utilisées pour cette fabrication.

3.33 Le responsable d'un club informatique souhaite acheter des souris, des clés USB et des tapis de souris, les matériels de chaque type étant identiques.

– S'il achète 3 souris, 5 clés USB et 3 tapis, il paiera 132,10 €.
– S'il achète 6 souris, 4 clés USB et 5 tapis, il paiera 172,90 €.
– Enfin, il paiera 118,90 € pour 4 souris, 6 clés USB et 1 tapis.

a) Traduire les données par un système de trois équations dont les inconnues sont les prix en euros des trois types de matériels.

b) Résoudre ce système à l'aide des matrices.

c) Combien coûte chaque type de matériel ?

3.34 On a demandé aux 215 étudiants de BTS SIO d'une académie de dire quel est leur langage de programmation préféré parmi Python, Xcas, Java et C. Chaque étudiant devait fournir une seule réponse.

On sait que 163 étudiants ont déclaré préférer Python ou Xcas, 65 ont déclaré préférer Xcas ou C, 158 ont déclaré préférer Python ou C.

a) Traduire les données par un système de quatre équations à quatre inconnues.

b) Résoudre ce système par les matrices.

c) Quelle est la répartition des préférences pour les quatre langages ?

3.35 Dans un repère du plan, on considère les points $A(-3;-23)$, $B(2;12)$ et $C(4;-2)$ et on cherche s'il existe une parabole passant par ces trois points. Pour cela, on prend $y = ax^2 + bx + c$ comme équation générale d'une parabole.

a) Montrer que A est sur la parabole si et seulement si $9a - 3b + c = -23$.

b) Traduire le fait que les points B et C appartiennent à la parabole.

c) Résoudre le système de trois équations dont les inconnues sont a, b et c.

d) Quelle est l'équation de la parabole ?

3.36 Dans un repère du plan, on considère les points $A(-2;-7)$, $B(-1;3,5)$, $C(2;17)$ et $D(3;25,5)$. En s'inspirant de l'exercice 3.35, trouver les nombres réels a, b, c et d tels que les points A, B, C et D appartiennent à la courbe d'équation $y = ax^3 + bx^2 + cx + d$.

3.37 On considère les matrices $A = \begin{pmatrix} 2 & -1 & 1 \\ -2 & 3 & -2 \\ -4 & 4 & -3 \end{pmatrix}$ et $B = I_3 + A$.

1) Calculer A^2. En déduire que $B \times A = 2A$.

2) On admet que, pour tout entier $n \geq 1$, il existe un unique réel a_n tel que $B^n = I_3 + a_n A$.

a) Que vaut a_1 ?

b) Exprimer B^{n+1} en fonction de I_3, A et a_{n+1}, puis en fonction de I_3, A et a_n.

c) En déduire que pour tout entier $n \geq 1$, $a_{n+1} = 1 + 2a_n$.

d) Déterminer a_2, a_3 et a_4.

3) On peut démontrer (voir exercice 2.29 du chapitre sur les suites numériques), que pour tout entier $n \geq 1$, on a $a_n = 2^n - 1$. En déduire l'expression de B^n en fonction de n.

Solutions

3.1 $A = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{pmatrix}$; $B = \begin{pmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{pmatrix}$

3.2 $A=\begin{pmatrix}1&2&3&4\\2&2&3&4\\3&3&3&4\end{pmatrix}$; $B=\begin{pmatrix}1&2&3&4&5\\2&4&6&8&10\end{pmatrix}$

3.3 a) $A+B=\begin{pmatrix}2&3\\6&10\end{pmatrix}$; $A-B=\begin{pmatrix}-4&1\\-4&-10\end{pmatrix}$

b) $(A+B)+(A-B)=\begin{pmatrix}-2&4\\2&0\end{pmatrix}$

On trouve $2A$. C'était prévisible car $(A+B)+(A-B)=A+B+A-B=A+A=2A$.

3.4 a) $A+A=\begin{pmatrix}0{,}6&2\\8&-4\end{pmatrix}$; $3A=\begin{pmatrix}0{,}9&3\\12&-6\end{pmatrix}$; $-10A=\begin{pmatrix}-3&-10\\-40&20\end{pmatrix}$

b) $xA=I_2 \Leftrightarrow \begin{pmatrix}0{,}3x&x\\4x&-2x\end{pmatrix}=\begin{pmatrix}1&0\\0&1\end{pmatrix} \Leftrightarrow \begin{cases}0{,}3x=1\\x=0\\4x=0\\-2x=1\end{cases}$

Ce système n'a pas de solution, il est donc impossible de trouver x tel que $xA=I_2$.

3.5 $\dfrac{1}{n}A=\begin{pmatrix}\dfrac{10}{n}&\dfrac{20}{n}\\\dfrac{8}{n}&\dfrac{100}{n}\end{pmatrix}$. Le plus grand élément de la matrice est $100/n$. Tous les éléments de la matrice seront donc strictement inférieurs à 0,01 dès que $100/n<0{,}01$.

$$\frac{100}{n}<0{,}01 \Leftrightarrow 100<0{,}01n \Leftrightarrow \frac{100}{0{,}01}<n \Leftrightarrow 10000<n$$

Le plus petit entier cherché est donc 10001.

3.6 a) $2A-B=\begin{pmatrix}23&-1&-5\\8&2&1\end{pmatrix}$; $3A-4B=\begin{pmatrix}57&-4&-15\\12&18&14\end{pmatrix}$

b) On a les deux équations : $\begin{cases}C+D=A\\C-D=B\end{cases}$

Par addition, $(C+D)+(C-D)=A+B$, donc $2C=A+B$,

d'où $C=\dfrac{1}{2}(A+B)=\begin{pmatrix}-1&0{,}5&1\\2&-4&-3{,}5\end{pmatrix}$.

Par soustraction, $(C+D)-(C-D)=A-B$ donc $2D=A-B$,

d'où $D=\dfrac{1}{2}(A-B)=\begin{pmatrix} 8 & -0{,}5 & -2 \\ 2 & 2 & 1{,}5 \end{pmatrix}$.

3.7 a) Le laboratoire L1 produit 31 000 médicaments de type M3 par mois.

b) $B=\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

3.8 a) $A\times B=\begin{pmatrix} 3 & 4 \\ 11 & 16 \\ 19 & 28 \end{pmatrix}$; $2B=\begin{pmatrix} 2 & 4 \\ 6 & 8 \end{pmatrix}$; $A\times(2B)=\begin{pmatrix} 6 & 8 \\ 22 & 32 \\ 38 & 56 \end{pmatrix}$

b) La vérification est facile.

3.9 a) $A\times B=\begin{pmatrix} -5 & 5 \\ 26 & -21 \\ 9 & -39 \end{pmatrix}$; $A\times C=\begin{pmatrix} 0 & -3 \\ 1 & 16 \\ -6 & 3 \end{pmatrix}$

b) On ne peut pas calculer $B\times A$ et $C\times A$ car le nombre de colonnes de B et de C n'est pas égal au nombre de lignes de A.

c) $B+C=\begin{pmatrix} 5 & -2 \\ 2 & 5 \end{pmatrix}$; $A\times(B+C)=\begin{pmatrix} -5 & 2 \\ 27 & -5 \\ 3 & -36 \end{pmatrix}$

d) $A\times(B+C)-A\times B-A\times C=O$ (matrice nulle)

Le résultat était prévisible puisque $A\times(B+C)=A\times B+A\times C$.

3.10 Les produits possibles sont A^2, $A\times B$, $C\times A$ et $C\times B$. En effet, dans chaque cas, le nombre de colonnes de la première matrice est égal au nombre de lignes de la seconde.

$$A^2=\begin{pmatrix} -14 & 15 \\ -9 & -11 \end{pmatrix} \qquad A\times B=\begin{pmatrix} 9 \\ -10 \end{pmatrix}$$

$$C\times A=\begin{pmatrix} -8 & 11 \\ 11 & 21 \\ -3 & 2 \end{pmatrix} \qquad C\times B=\begin{pmatrix} 7 \\ 18 \\ 1 \end{pmatrix}$$

3.11 $B^2 = \begin{pmatrix} 15 & 7 & -19 \\ 42 & 33 & -48 \\ -38 & -16 & 51 \end{pmatrix}$ $B \times A = \begin{pmatrix} -16 \\ -60 \\ 51 \end{pmatrix}$ $B \times C = \begin{pmatrix} -3 & 8 \\ -6 & 42 \\ 10 & -17 \end{pmatrix}$

3.12 $A \times B = B \times A = I_2$ $A \times C \times B = \begin{pmatrix} -16 & 30 \\ -9 & 17 \end{pmatrix}$

3.13 a) $A^2 = \begin{pmatrix} 27 & 14 \\ 7 & 6 \end{pmatrix}$ $A^3 = \begin{pmatrix} 149 & 82 \\ 41 & 26 \end{pmatrix}$

b) $B^2 = \begin{pmatrix} 6 & -3 & -3 \\ -3 & 6 & -3 \\ -3 & -3 & 6 \end{pmatrix}$ $B^3 = \begin{pmatrix} -18 & 9 & 9 \\ 9 & -18 & 9 \\ 9 & 9 & -18 \end{pmatrix}$

3.14 a) $B^2 = \begin{pmatrix} 0 & 0 & 6 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ $B^3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} = O$

b) $B^4 = B^3 \times B = O \times B = O$ et pour tout $n \geq 3$, $B^n = B^{n-3} \times B^3 = B^{n-3} \times O = O$.

3.15 a) $B \times A = \begin{pmatrix} 13 & 31 & 18 \\ 42 & 137 & 89 \end{pmatrix}$

13, 31 et 18 sont les masses unitaires respectives (en grammes) des matériels M1, M2 et M3.

42, 137 et 89 sont les prix unitaires respectifs (en euros) des matériels M1, M2 et M3.

b) On doit calculer le produit $A \times D = \begin{pmatrix} 770 \\ 500 \\ 620 \end{pmatrix}$.

770 composants C1, 500 composants C2 et 620 composants C3 seront nécessaires.

3.16 $A \times B = B \times A = I_2$ donc A et B sont inverses l 'une de l'autre.

3.17 On calcule $A \times A'$ ou $A' \times A$. Dans les deux cas, le résultat est I_3.

3.18 $A \times B = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} = 2I_3$ donc $A \times \left(\frac{1}{2}B\right) = I_3$.

L'inverse de A est donc $\frac{1}{2}B = \begin{pmatrix} 1,5 & -0,5 & 0,5 \\ 1,5 & -0,5 & 1,5 \\ 0,5 & -0,5 & 1,5 \end{pmatrix}$.

3.19 $A \times B = \begin{pmatrix} 9 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 9 \end{pmatrix} = 9I_3$ donc $A \times \left(\frac{1}{9}B\right) = I_3$.

L'inverse de A est donc $\frac{1}{9}B = \begin{pmatrix} \frac{1}{9} & \frac{4}{9} & \frac{1}{9} \\ \frac{2}{9} & -\frac{10}{9} & -\frac{1}{9} \\ -\frac{2}{9} & \frac{1}{9} & \frac{1}{9} \end{pmatrix}$.

3.20 $A \times B = B \times A = \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix} = 3I_2$ donc $A^{-1} = \frac{1}{3}B = \begin{pmatrix} -\frac{1}{3} & -\frac{5}{6} \\ \frac{2}{3} & \frac{7}{6} \end{pmatrix}$.

3.21 a) Les coefficients manquants sont : $\begin{cases} 5a+7+4 = 5a+11 \\ 2a+7-3 = 2a+4 \\ -3a-7+1 = -3a-6 \end{cases}$

b) B est l'inverse de A si et seulement si $\begin{cases} 5a+11 = 1 \\ 2a+4 = 0 \\ -3a-6 = 0 \end{cases}$.

Ces trois équations ont la même solution : $a = -2$.

Donc : $B = A^{-1} \Leftrightarrow a = -2$

3.22 a) $A \times B = \frac{1}{3}\begin{pmatrix} 3 & 7+a & 0 \\ 0 & 3 & 0 \\ 0 & 7+a & 3 \end{pmatrix} = \begin{pmatrix} 1 & \frac{7+a}{3} & 0 \\ 0 & 1 & 0 \\ 0 & \frac{7+a}{3} & 1 \end{pmatrix}$

b) B sera la matrice inverse de A si et seulement si $(7+a)/3 = 0$ c'est-à-dire $a = -7$.

3.23 a) $A \times A' = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \times \frac{1}{ad-bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} = \frac{1}{ad-bc} \begin{pmatrix} ad-bc & -ab+ab \\ cd-cd & -cb+ad \end{pmatrix}$

$$= \frac{1}{ad-bc} \begin{pmatrix} ad-bc & 0 \\ 0 & ad-bc \end{pmatrix}$$

Donc $A \times A' = I_2$, ce qui prouve que A' est l'inverse de A.

b) $A^{-1} = \frac{1}{7\times 8 - 3\times 12} \times \begin{pmatrix} 8 & -3 \\ -12 & 7 \end{pmatrix} = \begin{pmatrix} 0,4 & -0,15 \\ -0,6 & 0,35 \end{pmatrix}$

3.24 a) $A^2 - 4A = -I_2$.

b) $A \times (A - 4I_2) = A \times A - A \times (4I_2) = A^2 - 4A$

Par conséquent, $A \times (A - 4I_2) = I_2$. Le produit de A par $A - 4I_2$ est égal à la matrice I_2, donc A est inversible et $A^{-1} = A - 4I_2 = \begin{pmatrix} 3 & 1 \\ 4 & 1 \end{pmatrix} - 4 \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} -1 & 1 \\ 4 & -3 \end{pmatrix}$.

3.25 a) $A_2 - 5A = I_2$

b) $A^2 - 5A = -I_2 \Leftrightarrow -A^2 + 5A = I_2 \Leftrightarrow A \times (-A + 5I_2) = I_2$ donc A est inversible et on a :

$$A^{-1} = -A + 5I_2 = -\begin{pmatrix} -2 & -3 \\ 5 & 7 \end{pmatrix} + 5\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 7 & 3 \\ -5 & -2 \end{pmatrix}$$

3.26 a) $A^2 - 15A = 4I_2$

b) $A^2 - 15A = 4I_2 \Leftrightarrow \frac{1}{4}(A^2 - 15A) = I_2 \Leftrightarrow A \times \left(\frac{A - 15I_2}{4} \right) = I_2$ donc A est inversible et on a :

$$A^{-1} = \frac{1}{4}(A - 15I_2) = \frac{1}{4}\begin{pmatrix} -2 & -5 \\ -6 & -13 \end{pmatrix} = \begin{pmatrix} -0,5 & -1,25 \\ -1,5 & -3,25 \end{pmatrix}$$

3.27 a) Le système s'écrit $AX = B$ avec $A = \begin{pmatrix} 2 & -3 \\ -3 & 5 \end{pmatrix}$, $X = \begin{pmatrix} x \\ y \end{pmatrix}$ et $B = \begin{pmatrix} 5 \\ -2 \end{pmatrix}$.

À la calculatrice, on obtient $A^{-1} = \begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix}$.

La solution est $X = A^{-1} \times B = \begin{pmatrix} 19 \\ 11 \end{pmatrix}$, d'où $x = 19$ et $y = 11$.

b) $x = -7$; $y = 5$

c) $x = 3$; $y = -8/3$; $z = 11/3$

3.28 a) $A = \begin{pmatrix} -4 & 1 & 0.1 \\ 11 & -3 & -0.2 \\ -6 & 2 & 0.1 \end{pmatrix}$; $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$; $B = \begin{pmatrix} -1 \\ 1 \\ 4 \end{pmatrix}$

b) $X = A^{-1} \times B = \begin{pmatrix} 4 \\ 13 \\ 20 \end{pmatrix}$

3.29 a) $x = 10, y = 5, z = 8$ b) $x = 2, y = 7, z = 4$ c) $x = 1, y = 5, z = -2$

3.30 a) Le système s'écrit $AX = B$ avec $A = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 1 & 3 \\ 1 & -1 & 1 \end{pmatrix}$, $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ et $B = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$.

b) $X = A^{-1}B$ d'où : $x = 2a - b + c$; $y = 0{,}5a - 0{,}5c$; $z = -1{,}5a + b - 0{,}5c$

2) Pour (S_1), $a = 12$, $b = 25$ et $c = 10$ donc $x = 9$, $y = 1$ et $z = 2$.

Pour (S_2), $a = 7$, $b = 18$ et $c = 9$ donc $x = 5$, $y = -1$ et $z = 3$.

En divisant tous les coefficients de (S_3) par 3, on obtient un système similaire aux deux autres : $a = 9$, $b = 2$ et $c = 5$ d'où $x = 21$, $y = 2$ et $z = -14$.

3.31 a) On appelle x, y et z les prix respectifs, en euros, d'un stylo, d'une gomme et d'une pochette.

L'énoncé se traduit par le système : $\begin{cases} 4x + 2y + 3z = 10{,}70 \\ x + y + 5z = 7{,}80 \\ 2x + y + 4z = 8{,}10 \end{cases}$

b) Un stylo coûte 1,40 euro, une gomme 0,90 euro et une pochette 1,10 euro.

3.32 1) Pour $x = 20$ pièces de type 1, $y = 30$ pièces de type 2 et $z = 35$ pièces de type 3 :

$$N = 20 + 30 + 35 = 85 \text{ (pièces)}$$
$$M = 20 \times 2{,}5 + 30 \times 2 + 35 \times 1 = 145 \text{ (grammes)}$$
$$C = 20 \times 1 + 30 \times 1{,}5 + 35 \times 0{,}5 = 82{,}50 \text{ (euros)}$$

2) $N = x + y + z \qquad M = x \times 2{,}5 + y \times 2 + z \times 1 = 2{,}5x + 2y + z$
$C = x \times 1 + y \times 1{,}5 + z \times 0{,}5 = \ x\ + 1{,}5y\ + 0{,}5z$

$$3)\ a)\quad {}^{-1} = \begin{pmatrix} -0{,}5 & 1 & -1 \\ -0{,}25 & -0{,}5 & 1{,}5 \\ 1{,}75 & -0{,}5 & -0{,}5 \end{pmatrix}$$

$$b)\quad \begin{pmatrix} \\ \\ \\ \end{pmatrix} = {}^{-1} \times \begin{pmatrix} \\ \\ \\ \end{pmatrix} \text{ donc } \begin{cases} = -0{,}5 \quad + \quad - \\ = -0{,}25 \quad -0{,}5 \quad +1{,}5 \\ = 1{,}75 \quad -0{,}5 \quad -0{,}5 \end{cases}$$

4) La fabrication est optimale lorsque $N = 140$, $M = 275$ et $C = 135$.

On a alors $x = 70$, $y = 30$ et $z = 40$ c'est-à-dire 70 pièces de type 1, 30 de type 2 et 40 de type 3.

3.33 a) On note , et le prix en euros d'une souris, d'une clé USB et d'un tapis.

x, y, z sont solutions du système : $\begin{cases} 3 \quad +5 \quad +3 \quad = 132{,}10 \\ 6 \quad +4 \quad +5 \quad = 172{,}90 \\ 4 \quad +6 \quad + \quad = 118{,}90 \end{cases}$

b) Le système s'écrit $\quad = \quad$ avec $\quad = \begin{pmatrix} 132{,}10 \\ 172{,}90 \\ 118{,}90 \end{pmatrix}$, $\quad = \begin{pmatrix} \\ \\ \\ \end{pmatrix}$ et $\quad = \begin{pmatrix} 132{,}10 \\ 172{,}90 \\ 118{,}90 \end{pmatrix}$.

$$= {}^{-1} \times \quad = \begin{pmatrix} 6{,}90 \\ 12{,}50 \\ 16{,}30 \end{pmatrix} \text{ donc } \quad = 6{,}90, \quad = 12{,}50 \text{ et } \quad = 16{,}30.$$

c) Une souris coûte 6,90 €, une clé USB 12,50 € et un tapis 16,30 €.

3.34 a) On note , , et le nombre d'étudiants préférant respectivement Python, Xcas, Java et C.

D'après l'énoncé, on a $\begin{cases} \quad + \quad = 163 \\ \quad + \quad = 65 \\ \quad + \quad = 158 \end{cases}$ et $p+x+j+c=215$

d'où le système $\begin{cases} \quad + \quad = 163 \\ \quad + \quad = 65 \\ \quad + \quad = 158 \\ \quad + \quad + \quad + \quad = 215 \end{cases}$

b) $p=128$, $x=35$, $j=22$ et $c=30$.

c) Les préférences sont donc : 128 pour Python, 35 pour Xcas, 22 pour Java et 30 pour C.

3.35 a) Dans l'équation de la parabole, on remplace x et y par les coordonnées de A :

$$-23=a\times(-3)^2+b\times(-3)+c \Leftrightarrow 9a-3b+c=-23$$

b) L'appartenance de B à la parabole se traduit par $4a+2b+c=12$. Celle de C par $16a+4b+c=-2$.

c) $a=-2$; $b=5$; $c=10$

d) L'équation de la parabole est $y=-2x^2+5x+10$.

3.36 $a=0{,}5$; $b=-1$; $c=4$; $d=9$. L'équation de la courbe est $y=0{,}5x^3-x^2+4x+9$.

3.37 1) On trouve que $A^2=A$, donc :

$$B\times A=(I_3+A)\times A=I_3\times A+A^2=A+A=2A$$

2) a) $a_1=1$

b) D'une part, $B^{n+1}=I_3+a_{n+1}A$.

D'autre part, $B^{n+1}=B^n\times B=(I_3+a_nA)\times(I_3+A)$

$$\begin{aligned}&=I_3\times I_3+I_3\times A+a_nA\times I_3+a_nA\times A\\&=I_3+A+a_nA+a_nA^2\\&=I_3+A+a_nA+a_nA\\&=I_3+(1+2a_n)A\end{aligned}$$

c) $I_3+a_{n+1}A=I_3+(1+2a_n)A \Rightarrow a_{n+1}=1+2a_n$ (par unicité)

d) $a_2=3$; $a_3=7$; $a_4=15$

3) $B^n=I_3+(2^n-1)A=\begin{pmatrix}1+2(2^n-1) & -(2^n-1) & 2^n-1\\ -2(2^n-1) & 1+3(2^n-1) & -2(2^n-1)\\ -4(2^n-1) & 4(2^n-1) & 1-3(2^n-1)\end{pmatrix}$

Logique 4

PLAN
4.1 Calcul des propositions
4.2 Calcul des prédicats
4.3 Calcul booléen

OBJECTIFS
- ➤ Introduction d'éléments de logique en lien avec l'informatique
- ➤ Se familiariser avec le calcul portant sur des énoncés

4.1 Calcul des propositions

4.1.1 Généralités

Une **proposition** est un énoncé ayant un sens et dont on peut dire avec certitude, qu'il est vrai (on dit alors que VRAI est sa **valeur de vérité**) ou qu'il est faux (on dit alors que FAUX est sa valeur de vérité).

Exemples

– Les énoncés « Grand » et « 8 » ne sont pas des propositions car il n'ont aucun sens.
– L'énoncé « Je vais gagner au loto » n'est pas une proposition car on ne peut pas dire avec certitude qu'il est vrai ou faux.
– Les énoncés « $4 = 9$ » et « $7 + 9 > 11$ » sont des propositions. Leurs valeurs de vérité sont respectivement FAUX et VRAI.
– Les énoncés, écrits en langage Python, « `6*5==30` » et « `2!=2` » sont aussi des propositions, ils signifient respectivement « 6×5 est bien égal à 30 » et « 2 est différent de 2 ». La valeur de vérité du premier est VRAI, celle du second est FAUX.

Pour voir si un énoncé est une proposition, il est important de savoir dans quel langage il est écrit. Par exemple l'énoncé « `1==2` » a bien un sens en langage Python, tandis qu'il ne veut rien dire en français. De même, en Python, une écriture du type « A=5 » ne peut pas être une proposition car elle signifie « on affecte à A la valeur 5 ».

Les propositions sont très utilisées en programmation. Considérons le programme Python suivant :

```
While N<10:
    N=N+1
    print('bonjour')
```

Il se traduit par : *Tant que la valeur de vérité de la proposition « N<10 » est VRAI, augmenter N de 1 et afficher « bonjour ».*

4.1.2 Connecteurs logiques

a) La négation d'une proposition

La **négation** d'une proposition P que l'on note **non(P)** est définie par la **table de vérité** du tableau 4.1 (on peut aussi noter $\neg P$ ou même $\overline{P}$).

Tableau 4.1

P	non(P)
FAUX	VRAI
VRAI	FAUX

La négation d'une proposition change sa valeur de vérité.

Exemples

Considérons la proposition P : « 4 < 3 ». Sa négation est non(P) : « $4 \geq 3$ ». La valeur de vérité de P est FAUX donc celle de non(P) est VRAI.

En langage Python, « non » se note « not ». Considérons par exemple la série d'instructions suivante :

```
>>> A=(2==2)     # la variable A prend la valeur (2==2)
>>> A            # appel de la valeur de A
True
>>>not(A)        # appel de la valeur de not(A)
False
```

La négation de la négation d'une proposition, c'est la proposition elle-même. Par exemple, si P : « 2 = 3 », alors sa négation est non(P) : « $2 \neq 3$ » et la négation de non(P) est non(non(P)) : « 2 = 3 ». Ceci est toujours vrai et se traduit par le fait, que pour toute proposition P, les propositions P et non(non(P)) ont la même valeur de vérité (voir tableau 4.2).

Tableau 4.2

P	non(P)	non(non(P))
FAUX	VRAI	**FAUX**
VRAI	FAUX	**VRAI**

b) Équivalence de deux propositions

Si P et Q sont deux propositions, la proposition notée P $\Leftrightarrow$ Q (lire « *P équivaut à Q* ») est définie par la table de vérité présentée dans le tableau 4.3.

Tableau 4.3

P	Q	P $\Leftrightarrow$ Q
FAUX	FAUX	VRAI
FAUX	VRAI	FAUX
VRAI	FAUX	FAUX
VRAI	VRAI	VRAI

Deux propositions ne sont équivalentes que si elles ont la même valeur de vérité.

Exemples

– La proposition « 2 + 3 = 5 $\Leftrightarrow$ 5 < 6 » est vraie (équivalence de deux propositions vraies), de même que « 8 $\geq$ 10 $\Leftrightarrow$ 4 = 3 » (équivalence de deux propositions fausses).
– La proposition « 5 + 7 = 12 $\Leftrightarrow$ 2 = 3 » est fausse (équivalence de deux propositions dont l'une est vraie et l'autre fausse).
– En Python, l'équivalence se note aussi « == ». Testons, sur la console Python, la valeur de vérité de quelques propositions :

```
>>> (5<6)==(1==4)    # 5<6 est vrai et 1==4 est faux
False
>>> (6<5)==(1==4)    # 6<5 est faux et 1==4 est faux
True
```

Montrons, à l'aide d'une table de vérité, que la proposition (P $\Leftrightarrow$ non(non(P))) est vraie, quelle que soit la valeur de vérité de la proposition P (voir tableau 4.4).

Tableau 4.4

P	non(P)	non(non(P))	P $\Leftrightarrow$ non(non(P))
FAUX	VRAI	FAUX	VRAI
VRAI	FAUX	VRAI	VRAI

c) Conjonction de deux propositions

Si P et Q sont deux propositions, la proposition notée « P et Q » est définie par la table de vérité du tableau 4.5 (on peut aussi noter P $\wedge$ Q).

Tableau 4.5

P	Q	P et Q
FAUX	FAUX	FAUX
FAUX	VRAI	FAUX
VRAI	FAUX	FAUX
VRAI	VRAI	VRAI

« P et Q » n'est vraie que si P et Q sont vraies.

Exemples

– La proposition « 14 – 3 = 11 et 2 > 3 » est fausse (conjonction d'une proposition vraie et d'une proposition fausse). La proposition « 100 est pair et $100 = 10^2$ » est vraie (conjonction de deux propositions vraies).

– En Python, « et » se note « and ». Testons, sur la console Python, la valeur de vérité de quelques propositions (« != » signifie « est différent de »).

```
>>> (4==4) and (6!=6)   # 4==4 est vrai et 6!=6 est faux
False
>>> (4<5) and (6<10)    # 4<5 est vrai et 6<10 est vrai
True
```

d) Disjonction de deux propositions

Si P et Q sont deux propositions, la proposition notée « P ou Q » est définie par la table de vérité ci-après (on peut aussi noter $P \vee Q$).

Tableau 4.6

P	Q	P ou Q
FAUX	FAUX	FAUX
FAUX	VRAI	VRAI
VRAI	FAUX	VRAI
VRAI	VRAI	VRAI

« P ou Q » n'est vraie que si au moins une des propositions P ou Q est vraie.

Exemples

– La proposition « 14 – 3 = 11 ou 2 > 3 » est vraie (disjonction d'une proposition vraie et d'une proposition fausse). La proposition « 100 est pair ou $100 = 10^2$ » est aussi vraie (disjonction de deux propositions vraies).

– En Python, « ou » se note « or ». Testons, sur la console Python, la valeur de vérité de quelques propositions :

```
>>> (2!=2) or (5<7)     # 2!=2 est faux et 5<7 est vrai
True
>>> (7<3) or (6==7)     # 7<3 est faux et 6==7 est faux
False
```

e) Implication

Si P et Q sont deux propositions, la proposition notée « P ⇒ Q » (lire « P implique Q ») est définie par la table de vérité du tableau 4.7.

Tableau 4.7

P	Q	P ⇒ Q
FAUX	FAUX	VRAI
FAUX	VRAI	VRAI
VRAI	FAUX	FAUX
VRAI	VRAI	VRAI

Le faux implique « n'importe quoi » mais le vrai n'implique que le vrai.

Exemples

– La proposition « 5 < 5 ⇒ 5 = 5 » est vraie (le faux implique le vrai), tandis que la proposition « 5 = 5 ⇒ 5 < 5 » est fausse (le vrai n'implique pas le faux).
– En Python, « ⇒ » se note « <= ». On a donc `True<=True`, `False<=True`, `False<=False`, mais on n'a pas `True<=False`. Testons, sur la console Python, la valeur de vérité de quelques propositions :

```
>>> (3==2) <= (5<7)   # 3==2 est faux et 5<7 est vrai
True
>>> (2<4) <= (3>5)    # 2<4 est vrai et 3>5 est faux
False
```

4.1.3 Propriétés

Exemples

Montrons que la proposition ((P et Q) ⇔ (Q et P)) est vraie, quelles que soient les valeurs de vérité des propositions P et Q.

Tableau 4.8

P	Q	P et Q	Q et P	((P et Q) ⇔ (Q et P))
FAUX	FAUX	FAUX	FAUX	**VRAI**
FAUX	VRAI	FAUX	FAUX	**VRAI**
VRAI	FAUX	FAUX	FAUX	**VRAI**
VRAI	VRAI	VRAI	VRAI	**VRAI**

Montrons à l'aide d'une table de vérité que la proposition suivante est une tautologie, c'est-à-dire une proposition toujours vraie, quelles que soient les valeurs de vérité des propositions P, Q et R :

((P ou (Q et R)) ⇔ ((P ou Q) et (P ou R))

Pour condenser la table, nous noterons T cette proposition, F au lieu de FAUX et V au lieu de VRAI.

Tableau 4.9

P	Q	R	Q et R	P ou (Q et R)	P ou Q	P ou R	(P ou Q) et (P ou R)	T
F	F	F	F	F	F	F	F	**V**
F	F	V	F	F	F	V	F	**V**
F	V	F	F	F	V	F	F	**V**
F	V	V	V	V	V	V	V	**V**
V	F	F	F	V	V	V	V	**V**
V	F	V	F	V	V	V	V	**V**
V	V	F	F	V	V	V	V	**V**
V	V	V	V	V	V	V	V	**V**

De la même manière, on pourrait démontrer l'ensemble des points de la propriété 4.1.

Propriété 4.1

Quelles que soient les valeurs de vérité des propositions P, Q et R, les propositions suivantes sont vraies (ce sont des tautologies) :

– (P et Q) ⇔ (Q et P) (commutativité de l'opérateur et)
– (P ou Q) ⇔ (Q ou P) (commutativité de l'opérateur ou)
– ((P et Q) et R) ⇔ (P et (Q et R)) (associativité de l'opérateur et)
– ((P ou Q) ou R) ⇔ (P ou (Q ou R)) (associativité de l'opérateur ou)
– (P ou (Q et R)) ⇔ (P ou Q) et (P ou R) (distributivité de ou par rapport à et)
– (P et (Q ou R)) ⇔ (P et Q) ou (P et R) (distributivité de et par rapport à ou)
– (P ⇒ Q) ⇔ (non(P) ou Q)
– non(P et Q) ⇔ ((non(P)) ou (non Q)) (loi de Morgan)
– non(P ou Q) ⇔ ((non(P)) et (non Q)) (loi de Morgan)

L'utilisation de la propriété précédente permet de simplifier l'écriture de certaines propositions.

Exemple

La proposition « On n'a pas : $x = 1$ ou $x > 4$ » peut se noter non($x = 1$ ou $x > 4$), donc d'après une loi de Morgan, elle peut s'écrire (non($x = 1$) et non($x > 4$)), d'où plus simplement « $x \neq 1$ et $x \leq 4$ ».

4.2 Calcul des prédicats

Le symbole $\forall$ signifie « pour tout » et est appelé **quantificateur universel**.
Le symbole $\exists$ signifie « il existe » et est appelé **quantificateur existentiel**.

Exemples

« $\forall x \in \mathbb{R}, x^2 \geq 0$ » se lit « pour tout réel x, $x^2 \geq 0$ ».
« $\exists x \in \mathbb{N}, n^2 = n$ » se lit « il existe un entier naturel n tel que $n^2 = n$ ».

Dans les énoncés « $x^2 = 9$ », « $x < y$ », « n est divisible par 3 » on trouve des variables (x, y et n).

Ces énoncés ne sont pas des propositions car ils n'ont aucune valeur de vérité. Cependant, en ajoutant des quantificateurs, chacun de ces énoncés peut devenir une proposition. Prenons par exemple le prédicat « $x^2 = 9$ » et considérons les énoncés :

$$\text{P} : « \forall x \in \mathbb{R}, x^2 = 9 » \qquad \text{Q} : « \exists x \in \mathbb{N}, x^2 = 9 »$$

P et Q sont bien des propositions puisque ces énoncés ont un sens et une valeur de vérité : Q est vraie puisque, par exemple on a $3^2 = 9$, tandis que P est fausse car $x^2 = 9$ n'est pas vrai pour tout réel x (par exemple $5^2 \neq 9$).

Une **variable** est un élément (nombre, matrice, etc.) pouvant prendre plusieurs valeurs.

Un **prédicat** est un énoncé sans valeur de vérité, dans lequel intervient au moins une variable, et qui devient une proposition par ajout de quantificateurs.

Exemples

– La proposition « $\exists x \in \mathbb{R}, x < 1$ » qui signifie « *il existe un nombre réel x tel que x < 1* » est vraie puisque par exemple $0 < 1$. Cette proposition est constituée d'un quantificateur existentiel et du prédicat « $x < 1$ ».
– Considérons les propositions P : « $\exists x \in \mathbb{R}, x^2 = 3$ » et Q : « $\exists x \in \mathbb{N}, x^2 = 3$ », dans lesquelles intervient le prédicat $x^2 = 3$. La proposition P est vraie puisque, par exemple $\sqrt{3}^2 = 3$, tandis que Q est fausse. En effet, 3 n'est le carré d'aucun nombre entier.
– Soient S et T deux propositions construites autour du prédicat « $x < y$ » (prédicat à deux variables) :
S : « $\forall x \in \mathbb{R}, \exists y \in \mathbb{R}, x < y$ » T : « $\exists y \in \mathbb{R}, \forall x \in \mathbb{R}, x < y$ »
La proposition S signifie : « *pour tout réel x, il existe un réel y tel que x < y* ». Celle-ci est vraie puisque, quel que soit le nombre réel x, on peut toujours trouver un réel y qui lui est supérieur (il suffit de prendre $y = x + 1$).
La proposition T signifie « *il existe un réel y tel que pour tout réel x, x < y* ». Celle-ci est fausse car il n'existe aucun nombre réel supérieur à tous les autres et à lui-même.

L'ordre dans lequel interviennent les quantificateurs $\forall$ et $\exists$ a une importance.
On ne peut pas le changer sans changer le sens de la proposition.

On ne peut pas changer, dans une proposition, l'ordre des quantificateurs $\forall$ et $\exists$ sans changer son sens, mais on peut librement intervertir des quantificateurs de même nature. Par exemple une proposition du type « $\forall x > 0. \forall y \in \mathbb{R}$, etc. » peut s'écrire « $\forall y \in \mathbb{R}, \forall x > 0$, etc.» et une proposition du type « $\exists n \in \mathbb{N}, \exists x > 1$, etc. » peut s'écrire « $\exists x > 1, \exists n \in \mathbb{N}$, etc. ».

Exemples

– La négation de « *Tous les chats sont gris* » est « *Il existe un chat non gris* » et la négation de « *Il existe un chat gris* » est « *Tous les chats sont non gris* ».
– De même, la négation de « $\exists\, x \in \mathbb{R},\ x^2 = 9$ » est « $\forall\, x \in \mathbb{R},\ x^2 \neq 9$ » et la négation de « $\exists\, x \in \mathbb{R},\ x^2 \neq 9$ » est « $\forall\, x \in \mathbb{R},\ x^2 = 9$ ».

Plus généralement, on observe la propriété 4.2.

Propriété 4.2

Si $p(x)$ est un prédicat, alors :
– la négation de la proposition « $\forall\, x,\ p(x)$ » est « $\exists\, x$, non $(p(x))$ » ;
– la négation de la proposition « $\exists\, x,\ p(x)$ » est « $\forall\, x$, non $(p(x))$ ».

Exemples

– La négation de la proposition P : « $\forall\, n \in \mathbb{N},\ 2^n < 100$ » est « $\exists\, n \in \mathbb{N},\ 2^n \geq 100$ ». La proposition P est fausse puisque toutes les puissances de 2 ne sont pas inférieures à 100. Sa négation est donc vraie (on peut le vérifier en prenant par exemple $n = 10$ puisque $2^{10} = 1\,024 \geq 100$).
– La négation de la proposition Q : « $\exists\, n \in \mathbb{N},\ n/3 \in \mathbb{N}$ » est « $\forall\, n \in \mathbb{N},\ n/3 \notin \mathbb{N}$ ». La proposition Q est vraie car par exemple $6 \in \mathbb{N}$ et $6/3 = 2 \in \mathbb{N}$. Sa négation est donc fausse (elle signifie qu'aucun entier naturel n'est divisible par 3).

Quelle est la négation de la proposition « $\forall\, x \in \mathbb{R}, \exists\, y \in \mathbb{R}, x < y$ » ?

En appliquant les règles de négation précédentes, on obtient successivement : « $\exists\, x \in \mathbb{R}$, non($\exists\, y \in \mathbb{R},\ x < y$) » puis « $\exists\, x \in \mathbb{R}, \forall\, y \in \mathbb{R}$, non($x < y$) », d'où « $\exists\, x \in \mathbb{R}, \forall\, y \in \mathbb{R}, x \geq y$ ».

Propriété 4.3

On obtient la négation d'une proposition formée d'une suite de quantificateurs suivie d'un prédicat en changeant partout $\forall$ par $\exists$ et $\exists$ par $\forall$ puis en changeant le prédicat par sa négation.

Exemple

La négation de la proposition P : « $\forall\, n \in \mathbb{N}, \exists\, p \in \mathbb{N},\ p = n/2$ » est nonP : « $\exists\, n \in \mathbb{N}$, $\forall\, p \in \mathbb{N},\ p \neq n/2$ ».
P est fausse puisqu'elle exprime que tout entier naturel est divisible par 2. Sa négation est donc vraie ; elle exprime qu'il existe un entier naturel n qui ne soit pas divisible pas 2 (par exemple 3 n'est pas divisible par 2).

Pour démonter une affirmation, on peut montrer qu'elle est vraie, mais il peut être plus simple de montrer que sa négation est fausse et pour montrer qu'une affirmation est fausse, il peut être aussi plus simple de montrer que sa négation est vraie. Par exemple, pour démontrer que l'affirmation « $\forall\, x \in \mathbb{R},\ x^3 \neq 8$ » est fausse, on peut montrer que sa négation « $\exists\, x \in \mathbb{R},\ x^3 = 8$ » est vraie en remarquant que $2^3 = 8$ (on dit qu'il s'agit d'un contre-exemple).

4.3 Calcul booléen

4.3.1 Exemple introductif

a) Premier montage

On considère la portion de circuit électrique suivante :

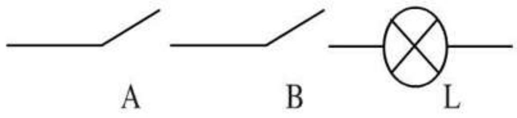


Figure 4.1 - A et B sont deux interrupteurs et L est une lampe.

On crée trois variables numériques a, b et l ne pouvant prendre que les valeurs 0 ou 1 en décidant que :

- $a = 0$ si l'interrupteur A est ouvert (le courant ne passe pas) et $a = 1$ si A est fermé.
- $b = 0$ si B est ouvert, $b = 1$ si B est fermé.
- $l = 0$ si la lampe L est éteinte, $l = 1$ si L est allumée.

Pour ce montage, on a la table présentée dans le tableau 4.10.

Tableau 4.10

a	b	l
0	0	0
0	1	0
1	0	0
1	1	1

On voit que $l = 1$ seulement lorsque a **ET** b valent 1.

b) Deuxième montage

On considère maintenant le montage suivant :

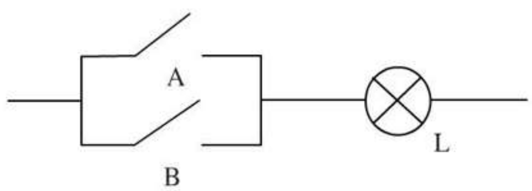


Figure 4.2

Pour ce montage, la table est présentée dans le tableau 4.11.

Tableau 4.11

a	*b*	*l*
0	0	0
0	1	1
1	0	1
1	1	1

On voit que $l = 1$ lorsque a **OU** b valent 1.

c) Bilan

Dans ces deux cas, on a défini des variables ne prenant que deux valeurs : 0 et 1. On dit que ce sont des **variables booléennes.**

Dans le premier montage, l est le **produit** des variables a et b, ce que l'on écrit $l = a{\cdot}b$ ou plus simplement $l = ab$.

Dans le deuxième montage, l est la **somme** des variables a et b, ce que l'on écrit $l = a + b$.

4.3.2 Algèbre de Boole

Considérons :

- un ensemble dont les éléments sont des variables (souvent notées a, b, c, …) ne pouvant prendre que deux valeurs : 0 et 1.
- et trois opérations appelées **addition**, **multiplication** et **complémentation** définies dans le tableau 4.12.

Tableau 4.12

a	b	a + b
0	0	0
0	1	1
1	0	1
1	1	1

addition

a	b	ab
0	0	0
0	1	0
1	0	0
1	1	1

multiplication

a	$\overline{a}$
0	1
1	0

complémentation

On dit que l'ensemble de ces variables, muni de ces trois opérations, a une structure **d'algèbre de Boole**. Les variables sont appelées **variables booléennes**.

Un lien évident avec la logique montre que l'addition correspond à OU, la multiplication à ET et la complémentation à NON.

Propriété 4.4

Dans une algèbre de Boole :
– L'addition et la multiplication sont commutatives : $a+b=b+a$; $ab=ba$.
– L'addition et la multiplication sont associatives : $(a+b)+c=a+(b+c)$; $(ab)c=a(bc)$.
– La multiplication et l'addition sont distributives l'une par rapport à l'autre : $a(b+c)=ab+ac$; $a+bc=(a+b)(a+c)$.
– 0 est élément neutre de l'addition : $0+a=a+0=a$.
– 1 est élément neutre de la multiplication : $1a=a1=a$.
– 1 est élément absorbant de l'addition : $a+1=1+a=1$.
– 0 est élément absorbant de la multiplication : $0a=a0=0$.
– Quel que soit a : $a+\overline{a}=1$; $a\overline{a}=0$.

Les démonstrations de la distributivité seront proposées dans l'exercice 4.20, les autres propriétés sont très simples à démontrer.

Exemple

Montrons que 0 est élément neutre de l'addition :
Si a = 0, alors a + 0 = 0 + 0 = 0 et 0 + a = 0 + 0 = 0. Le résultat est égal à la valeur de a.
Si a = 1, alors a + 0 = 1 + 0 = 1 et 0 + a = 0 + 1 = 1. Le résultat est encore égal à la valeur de a.
Donc, quel que soit a, on a 0 + a = a + 0 = a.

Dans une expression, la complémentation est prioritaire sur la multiplication, qui est elle-même prioritaire sur l'addition.

Si on doit calculer $a+b\overline{c}$ pour $a=0$, $b=1$ et $c=0$, on commence par calculer $\overline{c}=\overline{0}=1$.

Puis on calcule le produit $b\overline{c}=1\cdot1=1$ et enfin la somme $a+b\overline{c}=0+1=1$.

$a+a=a$ et plus généralement $a+a+...+a=a$.

$aa=a$ et plus généralement $aa...a=a$.

Ces formules montrent qu'en calcul booléen, il n'y a ni multiple ni puissance.

4.3.3 Lois de Morgan

Propriété 4.5 (lois de Morgan)

Quelles que soient les variables a et b d'une algèbre de Boole, on a : $\overline{a+b}=\overline{a}\cdot\overline{b}$ et $\overline{a\cdot b}=\overline{a}+\overline{b}$.

Les démonstrations peuvent se faire avec des tables de vérité comme illustré par les tableaux 4.13 et 4.14.

Les résultats identiques dans les colonnes colorées du tableau 4.13 montrent que $\overline{a+b}=\overline{a}\,\overline{b}$ et dans le tableau 4.14, que $\overline{ab}=\overline{a}+\overline{b}$.

Tableau 4.13

a	b	$a+b$	$\overline{a+b}$	$\overline{a}$	$\overline{b}$	$\overline{a}\,\overline{b}$
0	0	0	**1**	1	1	**1**
0	1	1	**0**	1	0	**0**
1	0	1	**0**	0	1	**0**
1	1	1	**0**	0	0	**0**

Tableau 4.14

a	b	ab	$\overline{ab}$	$\overline{a}$	$\overline{b}$	$\overline{a}+\overline{b}$
0	0	0	**1**	1	1	**1**
0	1	0	**1**	1	0	**1**
1	0	0	**1**	0	1	**1**
1	1	1	**0**	0	0	**0**

4.3.4 Tableaux de Karnaugh

Les tableaux de Karnaugh permettent de représenter facilement des expressions booléennes. Dans le cadre du programme, on se limitera à deux ou trois variables.

a) Cas de deux variables

Le tableau de Karnaugh comprend quatre cases, correspondant aux quatre produits $ab, a\overline{b}, \overline{a}b, \overline{a}\,\overline{b}$.

Tableau 4.15

ab	$a\overline{b}$
$\overline{a}b$	$\overline{a}\,\overline{b}$

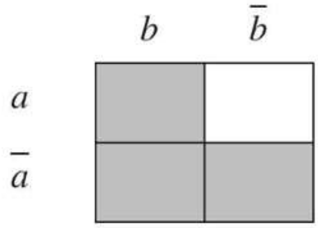
	b	$\overline{b}$
a		
$\overline{a}$		

La première ligne correspond à $ab + a\overline{b} = a(b+\overline{b}) = a \cdot 1 = a$, la deuxième ligne correspond à $\overline{a}$.

De même, la première colonne est b et la deuxième est $\overline{b}$. Pour représenter une expression dans un tableau de Karnaugh, on colore les cases concernées par l'expression.

> ***Exemples***
> L'expression $\overline{a}+b$ est représentée par le tableau 4.16, dans lequel on colore la ligne $\overline{a}$ et la colonne b.

Tableau 4.16



b) Cas de trois variables

Le tableau de Karnaugh comprend huit cases, correspondant aux huit produits : abc, $ab\bar{c}$, $a\bar{b}c$, $a\bar{b}\bar{c}$, $\bar{a}bc$, $\bar{a}b\bar{c}$, $\bar{a}\,\bar{b}\,c$, $\bar{a}\,\bar{b}\,\bar{c}$.

Tableau 4.17

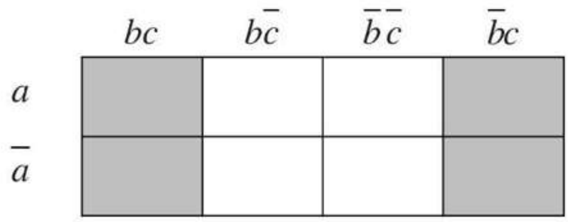
abc	$ab\bar{c}$	$a\bar{b}\,\bar{c}$	$a\bar{b}c$
$\bar{a}bc$	$\bar{a}\,b\bar{c}$	$\bar{a}\,\bar{b}\,\bar{c}$	$\bar{a}\,\bar{b}\,c$

	bc	$b\bar{c}$	$\bar{b}\,\bar{c}$	$\bar{b}c$
a				
$\bar{a}$				

Exemples

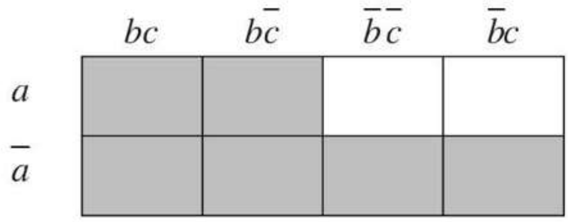
L'expression c est représentée par le tableau 4.18.

Tableau 4.18



Pour l'expression $\bar{a}+b$, on réunit la ligne $\bar{a}$ et les colonnes b (+ équivaut à OU et à réunion).

Tableau 4.19



Pour $\bar{a}\,c$, on prend l'intersection de la ligne $\bar{a}$ avec les colonnes c (. équivaut à ET et à intersection).

Tableau 4.20

	bc	$b\bar{c}$	$\bar{b}\bar{c}$	$\bar{b}c$
a				
$\bar{a}$				

Inversement, un tableau de Karnaugh permet de retrouver une expression (Fig. 4.3).

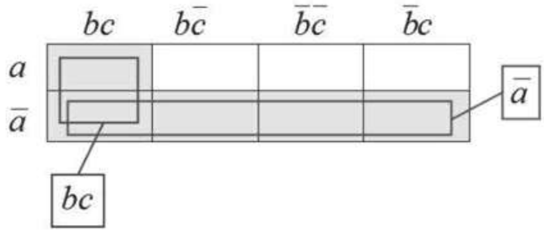


Figure 4.3

Il s'agit de l'expression $\bar{a} + bc$

TD – Expression booléenne

La connexion à un site Internet nécessite la saisie d'un mot de passe comportant de 8 à 12 caractères. Ces caractères peuvent être des lettres majuscules de l'alphabet français, ou des chiffres, ou des caractères spéciaux (tels que &, *, /, § etc).

Un mot de passe est valide si l'une au moins des trois conditions suivantes est réalisée :

- Il comporte au moins trois chiffres et trois caractères spéciaux.
- Il comporte au moins cinq lettres.
- Il comporte moins de trois chiffres mais au moins cinq lettres et trois caractères spéciaux.

Partie A – Reconnaître si un mot de passe est valide

a) Parmi les mots de passe suivants, quels sont ceux qui sont valides ?

H32EXZ&K5= LUC230598** 123(M*K<4

b) Alice veut créer un mot de passe avec quatre lettres, quatre chiffres et quatre caractères spéciaux. Ce mot de passe sera-t-il accepté ? Et un mot de passe de huit lettres ?

Partie B – Écriture d'une expression booléenne

On définit trois variables booléennes a, b et c de la façon suivante :

- $a = 1$ si le mot de passe contient au moins trois chiffres, sinon $a = 0$.
- $b = 1$ si le mot de passe contient au moins cinq lettres, sinon $b = 0$.
- $c = 1$ si le mot de passe contient au moins trois caractères spéciaux, sinon $c = 0$.
  ainsi que la variable A telle que $A = 1$ si le mot de passe est valide, $A = 0$ sinon.

a) Traduire chacune des trois conditions de validité d'un mot de passe à l'aide des variables a, b et c. En déduire l'expression de A.

b) Représenter A avec un tableau de Karnaugh. En déduire une expression simplifiée de A.

c) Par calculs, retrouver la forme simplifiée de A.

d) Exprimer par une phrase la condition pour qu'un mot de passe soit valide.

Partie C – Les mots de passe non valides

a) En utilisant le tableau de Karnaugh, déterminer l'expression de $\overline{A}$.

b) Retrouver le résultat par calculs.

c) Exprimer par une phrase la condition pour qu'un mot de passe soit refusé.

Solution du TD

Partie A – Reconnaître si un mot de passe est valide

a) H32EXZ&K5= est valide car il a cinq lettres.

LUC230598** ne remplit aucune des trois conditions donc il n'est pas valide.

123(M*K<4 est valide car il remplit la première condition (quatre chiffres et trois caractères spéciaux).

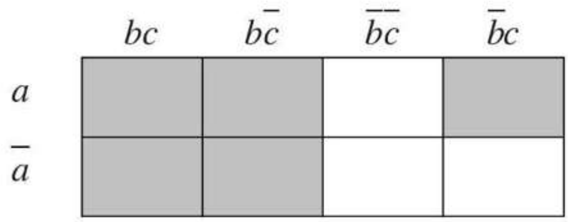
b) Un mot de passe avec quatre lettres, quatre chiffres et quatre caractères spéciaux est valide, car la première condition est remplie. Un mot de passe de huit lettres convient aussi car la deuxième condition est vérifiée.

Partie B – Écriture d'une expression booléenne

a) Les trois conditions s'expriment respectivement ac, b et $\overline{a}bc$. Donc $A = ac + b + \overline{a}bc$.

b)

Tableau 4.21



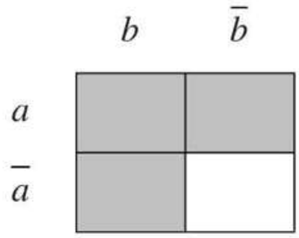
Le tableau 4.21 montre que l'on peut écrire $A = b + a\bar{b}c$, ou plus simplement, pour minimiser le nombre de symboles, $A = b + ac$.

c) $A = ac + b + \bar{a}bc$

$$\begin{aligned} &= c(a + \bar{a}b) + b && (1) \\ &= c(a + b) + b && (2) \\ &= ac + bc + b && (3) \\ &= ac + b(1 + c) && \\ &= ac + b && (4) \end{aligned}$$

(1) en factorisant par c ; (2) en utilisant un tableau de Karnaugh à deux variables (voir tableau 4.22) ; (3) par distributivité de la multiplication ; (4) car $1+c=1$.

Tableau 4.22 – **Tableau montrant que** $a + \bar{a}b = a + b$.



d) Un mot de passe est valide lorsqu'il a au moins cinq lettres ou qu'il a au moins trois chiffres et au moins trois caractères spéciaux.

Partie C – Les mots de passe non valides

a) Pour trouver l'expression de $\overline{A}$ grâce au tableau, il faut considérer les cases restées blanches : la troisième colonne correspond à $\bar{b}\bar{c}$ et les deux dernières cases de la deuxième ligne correspondent à $\bar{a}\bar{b}$ donc $\overline{A} = \bar{a}\bar{b} + \bar{b}\bar{c}$.

b) $A = ac + b$

$$\Rightarrow \overline{A} = \overline{ac + b}$$

$$\Rightarrow \overline{A} = \overline{ac} \cdot \overline{b} \qquad (1)$$

$$\Rightarrow \overline{A} = \left(\overline{a} + \overline{c}\right)\overline{b} \qquad (2)$$

$$\Rightarrow \overline{A} = \overline{a}\,\overline{b} + \overline{b}\,\overline{c}$$

(1) d'après une loi de Morgan ; (2) d'après l'autre loi de Morgan.

c) Un mot de passe est refusé lorsqu'il a moins de cinq lettres et moins de trois chiffres, ou moins de cinq lettres et moins de trois caractères spéciaux.

Exercices corrigés

4.1 Écrire sans implication la proposition « $x>1 \Rightarrow y = 3$ ».

4.2 On considère la proposition P : « $\forall\, n \in \mathbb{N},\ \exists p \in \mathbb{N},\ n = 2^p$ ».
Énoncer la négation de P puis donner, en justifiant, la valeur de vérité de P.

4.3 L'affirmation « la présente affirmation est fausse » est-elle une proposition ?

4.4 On note P et Q les affirmations suivantes :

- P = « Paul aime le foot »
- Q = « Paul aime les maths »

Représenter les affirmations suivantes sous forme symbolique en utilisant P, Q et des connecteurs logiques.

- A = « Paul aime le foot mais pas les maths »
- B = « Paul n'aime ni le foot, ni les maths »
- C = « Paul aime le foot ou il aime les maths et pas le foot »
- D = « Paul aime les maths et le foot ou il aime les maths mais pas le foot »

4.5 Quelles sont les valeurs de vérité des propositions suivantes :

- A = « $\pi = 5$ et $2 + 3 = 5$ »
- B = « $\pi = 5$ ou $2 + 3 = 5$ »
- C = « $\pi \approx 3{,}14 \Rightarrow 5 + 6 = 11$ »
- D = « $\pi = 5 \Rightarrow 2 + 3 = 5$ »
- E = « $4 = 5 \Rightarrow$ A est vraie » (A est la proposition citée plus haut)
- F = « $5 + 5 = 10 \Leftrightarrow \pi = 11$ »

4.6 Quelles sont les valeurs de vérité des propositions suivantes :

A = « $11 > 0$ et $3 < 2$ »

B = « $11 > 0$ ou $3 < 2$ »

C = « $3 > 6$ ou $6 > 20$ »

D = « $3 < 2 \Rightarrow 5 = 5$ »

E = « $4 \neq 1 \Rightarrow 4 = 1$ »

F = « $4 < 5 \Leftrightarrow 10 = 1 + 9$ »

4.7 Montrer, à l'aide d'une table de vérité, que les propositions suivantes sont des tautologies, c'est-à-dire qu'elles sont vraies quelles que soient les valeurs de vérité des propositions P et Q.

a) non(P et Q) $\Leftrightarrow$ (nonP ou nonQ) b) (P $\Rightarrow$ Q) $\Leftrightarrow$ (non P ou Q)

4.8 Montrer, à l'aide d'une table de vérité, que les propositions suivantes sont des tautologies, c'est-à-dire qu'elles sont vraies qu'elles que soient les valeurs de vérité des propositions qui les composent.

a) non(P ou Q) $\Leftrightarrow$ (nonP et non Q)

b) (P et (Q ou R)) $\Leftrightarrow$ ((P et Q) ou (P et R))

4.9 Vrai ou faux ? (Justifier)

a) $\exists\, x \in \mathbb{R},\, x^2 = x$ c) $\exists\, x \in \mathbb{R},\, (x+1)^2 = x^2 + 2x + 1$

b) $\forall\, x \in \mathbb{R},\, x/3 = (1/3)x$ d) $\forall\, x \in \mathbb{R},\, x^2 + 1 < 101$

4.10 Énoncer deux propositions de valeur de vérités différentes dans lesquelles apparaît le prédicat « $x = 2x + 1$ ».

4.11 Dans chacun des cas suivants, énoncer deux propositions dans lesquelles apparaît le prédicat donné : l'une vraie, l'autre fausse.

a) $x^2 \geq x$ b) $y = 2x$

4.12 Vrai ou faux ? (Justifier)

a) $\forall\, x \in \mathbb{R},\, \exists\, y \in \mathbb{R},\, y = x - 1$ b) $\exists\, x \in \mathbb{R},\, \forall\, y \in \mathbb{R},\, y = x - 1$

4.13 Vrai ou faux ? (Justifier)

a) $\forall\, x \in \mathbb{R},\, \forall\, n \in \mathbb{N},\, x + nx = (n+1)x$

b) $\exists\, x \in \mathbb{R},\, \exists\, n \in \mathbb{N},\, x + nx = (n+1)x$

c) $\exists\, n \in \mathbb{N},\, \exists\, p \in \mathbb{N},\, p - n$ est divisible par 2

d) $\forall\, n \in \mathbb{N},\, \forall\, p \in \mathbb{N},\, p - n$ est divisible par 2

4.14 Pour chaque proposition, donner sa valeur de vérité, énoncer sa négation puis donner la valeur de vérité de celle-ci.

a) $\exists\, x \in \mathbb{R},\, 3x = 2$

b) $\forall\, x \in \mathbb{R},\, x = x + 1$

c) $\forall\, x \in \mathbb{R},\, \forall\, y \in \mathbb{R},\, x \leq y$

4.15 Pour chaque proposition, donner sa valeur de vérité, énoncer sa négation puis donner la valeur de vérité de celle-ci.

a) $\exists\, x \in \mathbb{R},\, \forall\, y \in \mathbb{R},\, x^2 = y$

b) $\forall\, x \in \mathbb{R},\, \exists\, y \in \mathbb{R},\, x^2 = y$

4.16 Pour les valeurs de a, b et c données, calculer les expressions A, B et C :

$$A=\overline{a}+b+\overline{c},\qquad B=a\overline{b}+c,\qquad C=(a+b)\overline{c}$$

a) $a=0, b=1, c=1$

b) $a=1, b=1, c=1$

c) $a=1, b=1, c=0$

4.17 Démontrer à l'aide d'une table de vérité les formules suivantes :

a) $a+ab=a$ b) $a(a+b)=a$

4.18 Démontrer à l'aide d'une table de vérité les formules suivantes :

a) $a+\overline{a}b=a+b$ b) $(a+b)(a+\overline{b})=a$

4.19 Démontrer à l'aide d'une table de vérité les formules suivantes :

a) $a+b+\overline{a}\cdot\overline{b}=1$ b) $(a+b)\overline{a}\cdot\overline{b}=0$

4.20 Démontrer à l'aide d'une table de vérité les formules suivantes :

$a(b+c)=ab+ac$ (distributivité de la multiplication)

$a+bc=(a+b)(a+c)$ (distributivité de l'addition)

4.21 Démontrer par calculs les formules suivantes :

a) $a+ab=a$ b) $a(a+b)=a$

4.22 Démontrer par calculs les formules suivantes :

a) $a+\overline{a}b=a+b$ b) $(a+b)(a+\overline{b})=a$

4.23 Démontrer par calculs les formules suivantes :

a) $a+b+\overline{a}\cdot\overline{b}=1$ b) $(a+b)\overline{a}\cdot\overline{b}=0$

4.24 Démontrer à l'aide d'un tableau de Karnaugh les formules suivantes :

a) $a+\overline{a}b=a+b$ b) $a+b+\overline{a}\cdot\overline{b}=1$

4.25 a) Représenter par un tableau de Karnaugh les expressions :

$A=ab+\overline{c}$ et $B=\overline{a}+bc$

b) Utiliser le tableau pour déterminer les expressions de $\overline{A}$ et $\overline{B}$.

4.26 a) Représenter par un tableau de Karnaugh les expressions :

$$A=a\overline{c}+\overline{b}c+abc\qquad\text{et}\qquad B=\overline{a}b+\overline{b}c+a\overline{c}$$

b) Utiliser le tableau pour déterminer les expressions de $\overline{A}$ et $\overline{B}$.

4.27 Soit $A = ab + \overline{c}$ et $B = \overline{a} + bc$. Montrer par calculs que $\overline{A} = \overline{a}c + \overline{b}c$ et que $\overline{B} = a\overline{c} + a\overline{b}$.

4.28 Par calculs, démontrer la formule $(a+b)(a+c) = a + bc$.

4.29 On considère la loi * définie par la table de vérité du tableau :

Tableau 4.23

a	b	$a*b$
0	0	0
0	1	1
1	0	1
1	1	0

a) En logique, quel est l'équivalent de cette loi ?

b) À l'aide d'une table de vérité, montrer que $a*b = \overline{a}\cdot b + a\cdot\overline{b}$

c) Démontrer par calculs que $a*b = (a+b)\cdot(\overline{a}+\overline{b})$.

4.30 Soit $A = \overline{a}\cdot\overline{b} + a\overline{b} + b$. Montrer qua A=1 de deux façons : table de vérité et calculs.

4.31 Soit $A = ab + abc + a\overline{b}$. Montrer qua A=a de deux façons : tableau de Karnaugh et calculs.

4.32 Soit $A = \overline{a}\cdot\overline{b}\cdot\overline{c} + a\overline{b}c + \overline{a}\cdot\overline{b} + a\overline{b}\cdot\overline{c}$. Simplifier A avec un diagramme de Karnaugh, puis vérifier par des calculs.

4.33 On définit une loi $*$ par $a*b = \overline{a}\cdot b + a\cdot\overline{b}$

a) Montrer que $a * a = 0$

b) Calculer $a*\overline{a}$

c) Calculer $a*1$

d) Montrer que 0 est élément neutre pour la loi $*$

4.34 On désigne par $\downarrow$ la loi définie par $a\downarrow b = \overline{a+b}$ où a et b sont des variables booléennes.

a) Que valent $a\downarrow 0$ et $a\downarrow 1$?

b) Soit $A = (a\downarrow b) + (a\downarrow a)$. Simplifier A par le calcul.

c) Soit a, b et c trois variables booléennes. A-t-on $(a\downarrow b)\downarrow c = a\downarrow(b\downarrow c)$?

4.35 Dans un grand magasin, le service clientèle a organisé une classification des clients en trois catégories :

- Si le client achète un article, il est classé en catégorie A et $a = 1$, sinon $a = 0$.
- Si le client échange ou rend un article, il est classé en catégorie E et $e = 1$ sinon $e = 0$.

- Si le client demande des renseignements, il est classé en catégorie R et $r = 1$ sinon $r = 0$.

Soit $F = a \cdot \overline{e} \cdot r + a \cdot \overline{e} \cdot \overline{r}$

a) Faire le diagramme de Karnaugh de F.

b) En déduire une forme simplifiée de F.

c) Retrouver la forme simplifiée de F par calcul.

d) Quel type de client est un client de type $\overline{F}$? Est-ce un client peu intéressant, assez intéressant ou très intéressant pour le magasin ? Donner l'écriture la plus simple de $\overline{F}$.

4.36 Une société désire recruter en interne des collaborateurs pour sa filiale en Asie. Pour chaque employé, on définit les variables booléennes suivantes :

$a = 1$ s'il a plus de cinq ans d'ancienneté dans l'entreprise.

$b = 1$ s'il possède un BTS SIO.

$c = 1$ s'il parle couramment l'anglais.

La direction des ressources humaines décide que pourront postuler les employés :

- qui satisfont aux trois conditions ;
- ou qui ont moins de cinq ans d'expérience mais qui maîtrisent l'anglais ;
- ou qui ne maîtrisent pas l'anglais mais ont un BTS SIO.

a) Écrire une expression booléenne E traduisant les critères de sélection de la direction.

b) Représenter l'expression E par un tableau de Karnaugh.

c) À l'aide du tableau, donner une expression simplifiée de E.

d) Retrouver le résultat par calcul.

e) Déduire des questions 3 ou 4 une version simplifiée des critères de la direction.

4.37 On considère l'expression $E = \overline{a}\,\overline{c} + b\overline{c} + a\overline{b} + \overline{a}\,\overline{b}c$.

a) Simplifier l'écriture de $\overline{E}$ à l'aide d'un diagramme de Karnaugh et en déduire que $E = \overline{b} + \overline{c}$. Retrouver par calcul la forme simplifiée de E.

b) Dans un organisme qui aide des personnes au chômage à retrouver un emploi, on considère pour ces personnes les trois variables booléennes a, b et c définies ainsi :

- $a = 1$ si la personne a 45 ans ou plus, sinon $a = 0$.
- $b = 1$ si la personne est au chômage depuis un an ou plus, sinon $b = 0$.
- $c = 1$ si la personne a déjà suivi une qualification l'année précédente, sinon $c = 0$.

Une formation sera mise en place pour les personnes vérifiant au moins un des critères suivants :

- Avoir 45 ans ou plus et être au chômage depuis moins d'un an.
- Avoir moins de 45 ans et ne pas avoir suivi de formation l'année précédente.

- Être au chômage depuis un an ou plus et ne pas avoir suivi de formation l'année précédente.
- Avoir moins de 45 ans, être au chômage depuis moins d'un an et avoir suivi une formation l'année précédente.

Les personnes qui ne répondent à aucun de ces quatre critères pourront participer à un stage d'insertion en entreprise.

1) Écrire l'expression booléenne F en fonction de a, b et c qui traduit le fait que la personne pourra suivre cette formation.

2) En déduire les personnes qui ne pourront pas participer à la formation et qui participeront donc à un stage d'insertion en entreprise.

4.38 Une entreprise décide de choisir de nouveaux chefs de service parmi ses employés en se servant des variables booléennes suivantes :

- $a = 1$ si et seulement si l'employé a plus de 10 ans d'ancienneté dans l'entreprise.
- $b = 1$ si et seulement si l'employé arrive souvent en retard.
- $c = 1$ si et seulement si l'employé a des relations difficiles avec ses collègues.

L'entreprise fait une première sélection parmi ses employés en considérant les critères suivants : « l'employé est ponctuel et s'entend bien avec ses collègues » ou « l'employé est dans l'entreprise depuis au moins 10 ans ».

a) Donner l'expression booléenne E correspondant à un employé qui respecte les conditions pour devenir chef de service.

b) Donner l'expression de $\overline{E}$ à l'aide du tableau de Karnaugh. Retrouver le résultat par calculs.

4.39 Le gérant d'un magasin de vente de matériel d'occasion décide de réaliser une enquête sur les critères de choix des clients concernant l'achat d'ordinateurs.

Il examine trois critères, associés à trois variables booléennes a, b et c.

- La variable a concerne l'ancienneté. $a = 0$ si l'ordinateur a moins d'un an, $a = 1$ s'il a plus d'un an.
- La variable b concerne l'état. $b = 0$ si l'ordinateur est un peu abimé, $b = 1$ s'il est en bon état.
- La variable c concerne la fiabilité. $c = 0$ si la marque est réputée peu fiable, $c = 1$ sinon.

Après dépouillement, il apparaît que les clients achètent un ordinateur si :

- il a moins d'un an et que sa marque est réputée fiable ;
- ou s'il a plus d'un an mais qu'il est en bon état ;
- ou si sa marque est réputée peu fiable mais qu'il a moins d'un an.

a) Traduire par une variable booléenne E l'ensemble des critères d'achat.

b) Faire la table de Karnaugh de E et donner une expression simplifiée de E. Traduire par une phrase cette expression simplifiée.

c) Montrer par calculs, que $E = \overline{a} + b$. En déduire une expression de $\overline{E}$ et traduire cette expression par une phrase.

Solutions

4.1 On sait que (P $\Rightarrow$ Q) $\Leftrightarrow$ ((non(P)) ou Q) est toujours vrai (c'est une tautologie), donc :

$$\begin{aligned}(x > 1 \Rightarrow y = 3) &\Leftrightarrow (\text{non}(x > 1) \text{ ou } y = 3)\\ &\Leftrightarrow (x \leq 1 \text{ ou } y = 3)\end{aligned}$$

La proposition « $x > 1 \Rightarrow y = 3$ » peut donc s'écrire « $x \leq 1$ ou $y = 3$ ».

4.2 nonP : « $\exists\, n \in \mathbb{N}, \forall\, p \in \mathbb{N}, n \neq 2^p$ ». Par exemple pour $n = 3$, tout entier p vérife $2^p \neq 3$ car 3 n'est pas une puissance de 2. Cela démontre que nonP est vraie, donc que P est fausse.

4.3 L'affirmation « *la présente affirmation est fausse* » n'est pas une proposition car on ne peut pas lui attribuer une valeur de vérité. En effet, si la valeur de vérité de l'affirmation est VRAI, c'est que ce qu'elle affirme est vrai, donc que l'affirmation est fausse, c'est-à-dire que sa valeur de vérité est FAUX. De même, si la valeur de vérité de l'affirmation est FAUX, c'est que ce qu'elle affirme est faux, donc que l'affirmation est vraie, c'est-à-dire que sa valeur de vérité est VRAI.

4.4 A = (P et nonQ) B = (nonP et nonQ) ;

C = (P ou (Q et nonP)) D = ((P et Q) ou (Q et nonP))

4.5 A : FAUX ($\pi = 5$ est faux, 2 + 3 = 5 est vrai et « FAUX et VRAI » vaut FAUX)

B : VRAI (« FAUX ou VRAI » vaut VRAI)

C : VRAI ($\pi \approx 3{,}14$ est vrai, 5 + 6 = 11 est vrai et « VRAI $\Rightarrow$ VRAI » vaut VRAI)

D : VRAI ($\pi = 5$ est faux, 2 + 3 = 5 est vrai et « FAUX $\Rightarrow$ VRAI » vaut VRAI)

E : VRAI (4 = 5 est faux, « A est vraie » est faux et « FAUX $\Rightarrow$ FAUX » vaut VRAI)

F : FAUX (5 + 5 = 10 est vrai, $\pi = 11$ est faux et « VRAI $\Leftrightarrow$ FAUX » vaut FAUX)

4.6 A : FAUX (11 > 0 est vrai, 3 < 2 est faux et « VRAI et FAUX » vaut FAUX)

B : VRAI (« VRAI ou FAUX » vaut VRAI)

C : FAUX (3 > 6 est faux, 6 > 20 est faux, et « FAUX ou FAUX » vaut FAUX)

D : VRAI ($3 < 2$ est faux, $5 = 5$ est vrai, et « FAUX $\Rightarrow$ VRAI » vaut VRAI)

E : FAUX ($4 \neq 1$ est vrai, $4 = 1$ est faux et « VRAI $\Rightarrow$ FAUX » vaut FAUX)

F : VRAI ($4 < 5$ est vrai, $10 = 1 + 9$ est vrai et « VRAI $\Leftrightarrow$ VRAI » vaut VRAI)

4.7 a) Notons T la proposition « non(P et Q) $\Leftrightarrow$ (nonP ou nonQ) ».

Tableau 4.24

P	Q	P et Q	non(Q et P)	NonP	nonQ	nonP ou non Q	T
FAUX	FAUX	FAUX	VRAI	VRAI	VRAI	VRAI	**VRAI**
FAUX	VRAI	FAUX	VRAI	VRAI	FAUX	VRAI	**VRAI**
VRAI	FAUX	FAUX	VRAI	FAUX	VRAI	VRAI	**VRAI**
VRAI	VRAI	VRAI	FAUX	FAUX	FAUX	FAUX	**VRAI**

b) Notons T la proposition « (P $\Rightarrow$ Q) $\Leftrightarrow$ (nonP ou Q) ».

Tableau 4.25

P	Q	P$\Rightarrow$Q	NonP	nonP ou Q	T
FAUX	FAUX	VRAI	VRAI	VRAI	**VRAI**
FAUX	VRAI	VRAI	VRAI	VRAI	**VRAI**
VRAI	FAUX	FAUX	FAUX	FAUX	**VRAI**
VRAI	VRAI	VRAI	FAUX	VRAI	**VRAI**

4.8 a) Notons T la proposition « non(P ou Q) $\Leftrightarrow$ (nonP et nonQ) »

Tableau 4.26

P	Q	P ou Q	non(P ou Q)	nonP	nonQ	nonP et nonQ	T
FAUX	FAUX	FAUX	VRAI	VRAI	VRAI	VRAI	**VRAI**
FAUX	VRAI	VRAI	FAUX	VRAI	FAUX	FAUX	**VRAI**
VRAI	FAUX	VRAI	FAUX	FAUX	VRAI	FAUX	**VRAI**
VRAI	VRAI	VRAI	FAUX	FAUX	FAUX	FAUX	**VRAI**

La dernière colonne du tableau montre que T est toujours vraie.

b) Notons T la proposition : « (P et (Q ou R)) $\Leftrightarrow$ ((P et Q) ou (P et R)) ».

Tableau 4.27

P	Q	R	Q ou R	P et (Q ou R)	P et Q	P et R	(P et Q) ou (P et R)	T
FAUX	FAUX	FAUX	FAUX	FAUX	FAUX	FAUX	FAUX	**VRAI**
FAUX	FAUX	VRAI	VRAI	FAUX	FAUX	FAUX	FAUX	**VRAI**
FAUX	VRAI	FAUX	VRAI	FAUX	FAUX	FAUX	FAUX	**VRAI**
FAUX	VRAI	VRAI	VRAI	FAUX	FAUX	FAUX	FAUX	**VRAI**
VRAI	FAUX	FAUX	FAUX	FAUX	FAUX	FAUX	FAUX	**VRAI**
VRAI	FAUX	VRAI	VRAI	VRAI	FAUX	VRAI	VRAI	**VRAI**
VRAI	VRAI	FAUX	VRAI	VRAI	VRAI	FAUX	VRAI	**VRAI**
VRAI	VRAI	VRAI	VRAI	VRAI	VRAI	VRAI	VRAI	**VRAI**

La dernière colonne du tableau montre que T est toujours vraie.

4.9 a) Vrai : il existe bien un réel x tel que $x^2 = x$ puisque, par exemple, $1^2 = 1$.

b) Vrai : c'est une propriété du calcul fractionnaire.

c) Vrai : puisque l'égalité est vraie pour tout réel (c'est une identité remarquable), il existe bien un réel vérifiant cette égalité (il suffit d'en choisir un au hasard).

d) Faux : tous les réels ne vérifient pas cette inégalité puisque par exemple, $12^2 + 1 = 145$ et 145 n'est pas inférieur à 101.

4.10 Proposition vraie : $\exists\, x \in \mathbb{R},\ x = 2x + 1$ (en effet, $-1 = 2 \times (-1) + 1$).

Proposition fausse : $\forall\, x \in \mathbb{R},\ x = 2x + 1$ (en effet, par exemple 0 ne vérifie pas $0 = 2 \times 0 + 1$).

4.11 a) Proposition vraie : $\exists\, x \in \mathbb{R},\ x^2 \geq x$ (par exemple, $x = 5$).

Proposition fausse : $\forall\, x \in \mathbb{R},\ x^2 \geq x$.

b) Proposition vraie : $\forall\, x \in \mathbb{R}, \exists\, y \in \mathbb{R},\ y = 2x$ (cela veut dire que tout nombre réel x possède un double $2x$).

Proposition fausse : $\exists\, x \in \mathbb{R}, \forall\, y \in \mathbb{R},\ y = 2x$.

4.12 a) L'énoncé signifie « *Pour tout réel x il existe un réel y tel que y = x − 1* ». C'est vrai car, pour tout réel x, $x - 1$ est aussi un réel.

b) L'énoncé signifie « *Il existe au moins un réel x tel que pour tout réel y on ait y = x − 1* ». C'est faux car, sinon tous les réels seraient égaux au même nombre $x - 1$.

4.13 a) Vrai, c'est une factorisation par x.

b) Vrai, par exemple avec $x = 2$ et $n = 5$, on a $2 + 5 \times 2 = (5 + 1) \times 2$.

c) Vrai, par exemple, avec $n = 7$ et $p = 15$, $p - n = 15 - 7 = 8$ et 8 est divisible par 2.

d) Faux, $n = 7$ et $p = 12$ fournit un contre exemple : $p - n = 5$ qui n'est pas divisible par 2.

4.14 a) Valeur de vérité : VRAI (car $x = 2/3$ vérifie $3x = 2$)

Négation : $\forall x \in \mathbb{R}, 3x \neq 2$ (valeur de vérité : FAUX)

b) Valeur de vérité : FAUX (car par exemple, on n'a pas $2 = 2 + 1$)
Négation : $\exists x \in \mathbb{R}, x \neq x + 1$ (valeur de vérité : VRAI)

c) Valeur de vérité : FAUX (sinon tout réel serait tel que tous les autres réels lui sont supérieurs ou égaux).
Négation : $\exists x \in \mathbb{R}, \exists y \in \mathbb{R}, x > y$ (valeur de vérité : VRAI)

4.15 a) Faux. Si y est strictement négatif, il n'existe pas x réel tel que $x^2 = y$ car x^2 est toujours positif.

Négation : $\forall x \in \mathbb{R}, \exists y \in \mathbb{R}, x^2 \neq y$. Elle est vraie.

b) Vrai, car tout réel possède un carré.

Négation : $\exists x \in \mathbb{R}, \forall y \in \mathbb{R}, x^2 \neq y$. Elle est fausse.

4.16 a) $A = \bar{0} + 1 + \bar{1} = 1 + 1 + 0 = 1$ $\quad B = 0 \cdot \bar{1} + 1 = 0 \cdot 0 + 1 = 1$

$C = (0 + 1) \cdot \bar{1} = 1 \cdot 0 = 0$

b) $A = \bar{1} + 1 + \bar{1} = 0 + 1 + 0 = 1$ $\quad B = 1 \cdot \bar{1} + 1 = 1 \cdot 0 + 1 = 1$

$C = (1 + 1) \cdot \bar{1} = 1 \cdot 0 = 0$

c) $A = \bar{1} + 1 + \bar{0} = 0 + 1 + 1 = 1$ $\quad B = 1 \cdot \bar{1} + 0 = 1 \cdot 0 + 0 = 0$

$C = (1 + 1) \cdot \bar{0} = 1 \cdot 1 = 1$

4.17

Tableau 4.28 – **À gauche réponse à la question a), À droite réponse à la question b).**

a	b	ab	$a+ab$
0	0	0	**0**
0	1	0	**0**
1	0	0	**1**
1	1	1	**1**

a	b	$a+b$	$a(a+b)$
0	0	0	**0**
0	1	1	**0**
1	0	1	**1**
1	1	1	**1**

Les colonnes colorées étant identiques, on en déduit que $a = a + ab$ et que $a = a(a + b)$.

4.18

Tableau 4.29 – **Réponse à la question a).**

a	b	$\overline{a}$	$\overline{a}\cdot b$	$a+\overline{a}b$	$a+b$
0	0	1	0	**0**	**0**
0	1	1	1	**1**	**1**
1	0	0	0	**1**	**1**
1	1	0	0	**1**	**1**

Tableau 4.30 – **Réponse à la question b).**

a	b	$\overline{b}$	$a+b$	$a+\overline{b}$	$(a+b)\,(a+\overline{b})$
0	0	1	0	1	**0**
0	1	0	1	0	**0**
1	0	1	1	1	**1**
1	1	0	1	1	**1**

4.19

Tableau 4.31 – **Réponse à la question a). Posons** $A = a+b+\overline{a}\cdot\overline{b}$**.**

a	b	$\overline{a}$	$\overline{b}$	$\overline{a}\cdot\overline{b}$	$a+b$	***A***
0	0	1	1	1	0	**1**
0	1	1	0	0	1	**1**
1	0	0	1	0	1	**1**
1	1	0	0	0	1	**1**

Tableau 4.31 – **Réponse à la question a). Posons** $B = (a+b)\overline{a}\cdot\overline{b}$**.**

a	b	$\overline{a}$	$\overline{b}$	$\overline{a}\cdot\overline{b}$	$a+b$	***B***
0	0	1	1	1	0	**0**
0	1	1	0	0	1	**0**
1	0	0	1	0	1	**0**
1	1	0	0	0	1	**0**

Donc, quelles que soient les valeurs de a et de b, on a : $a+b+\overline{a}\overline{b}=1$ et $(a+b)\overline{a}\overline{b}=0$.

4.20 Dans chaque tableau, l'égalité des colonnes colorées permet de prouver la formule.

Tableau 4.32

a	b	c	$b+c$	$a(b+c)$	ab	ac	$ab+ac$
0	0	0	0	**0**	0	0	**0**
0	0	1	1	**0**	0	0	**0**
0	1	0	1	**0**	0	0	**0**
0	1	1	1	**0**	0	0	**0**
1	0	0	0	**0**	0	0	**0**
1	0	1	1	**1**	0	1	**1**
1	1	0	1	**1**	1	0	**1**
1	1	1	1	**1**	1	1	**1**

Tableau 4.33

a	b	c	bc	$a+bc$	$a+b$	$a+c$	$(a+b)(a+c)$
0	0	0	0	**0**	0	0	**0**
0	0	1	0	**0**	0	1	**0**
0	1	0	0	**0**	1	0	**0**
0	1	1	1	**1**	1	1	**1**
1	0	0	0	**1**	1	1	**1**
1	0	1	0	**1**	1	1	**1**
1	1	0	0	**1**	1	1	**1**
1	1	1	1	**1**	1	1	**1**

4.21 a) $a+ab = a\cdot 1 + ab = a(1+b) = a\cdot 1 = a$

b) $a(a+b) = aa + ab = a + ab = a$ (d'après le calcul précédent)

4.22 a) Par distributivité de l'addition :

$$\begin{aligned} a+\overline{a}b &= (a+\overline{a})(a+b) \\ &= 1(a+b) \\ &= a+b \end{aligned}$$

b) En appliquant deux fois la distributivité de la multiplication :

$$\begin{aligned} (a+b)\left(a+\overline{b}\right) &= aa + a\overline{b} + ab + b\overline{b} \\ &= a + a\overline{b} + ab + 0 \\ &= a + a(b+\overline{b}) \\ &= a + a\cdot 1 \\ &= a + a \\ &= a \end{aligned}$$

4.23 a)

$$\begin{aligned} a+b+\overline{a}\,\overline{b} &= a+b+\overline{a+b} && (1)\\ &= c+\overline{c} && (2)\\ &= 1 \end{aligned}$$

(1) d'après une loi de Morgan ; (2) en posant $a+b=c$.

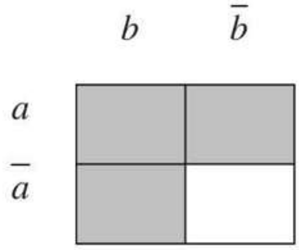
b)

$$\begin{aligned} (a+b)\overline{a}\,\overline{b} &= (a+b)\overline{(a+b)} && (1)\\ &= c\overline{c} && (2)\\ &= 0 \end{aligned}$$

(1) d'après une loi de Morgan ; (2) en posant $a+b=c$.

4.24 a) On colore la ligne *a*, puis la case $\overline{a}b$. Cela revient à avoir coloré la ligne *a* et la colonne *b*.

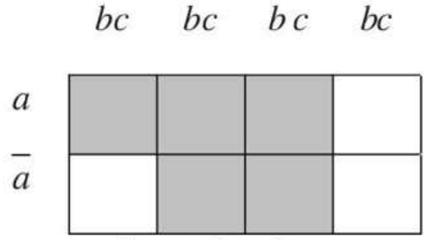
Tableau 4.34



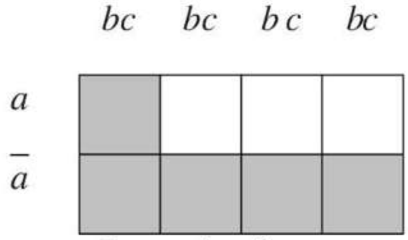
b) On colore la ligne *a* et la colonne *b*, puis la case $\overline{a}\,\overline{b}$. L'ensemble du tableau est coloré, ce qui correspond à l'expression 1.

4.25 a)

Tableau 4.35



Expression *A*

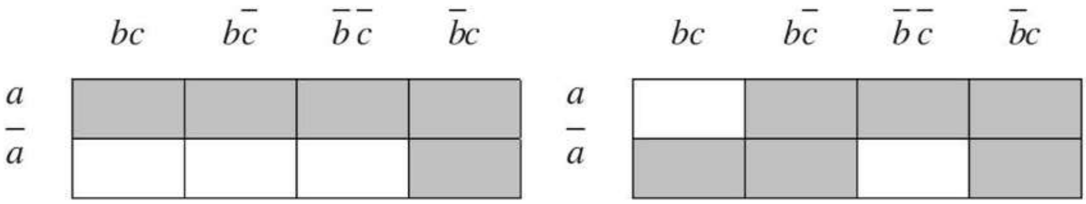


Expression *B*

b) $\overline{A}=\overline{a}bc+\overline{b}c$ ou $\overline{A}=\overline{a}c+\overline{b}c$ $\overline{B}=a\overline{c}+a\overline{b}c$ ou $\overline{B}=a\overline{c}+a\overline{b}$

4.26 a)

Tableau 4.36



b) $\overline{A} = \overline{a}b + \overline{a}\,\overline{b}\,\overline{c}$ ou $\overline{A} = \overline{a}b + \overline{a}\,\overline{c}$ $\qquad$ $\overline{B} = abc + \overline{a}\,\overline{b}\,\overline{c}$

4.27

$$\begin{aligned} \overline{A} &= \overline{ab+\overline{c}} = \overline{ab}\cdot\overline{\overline{c}} && (1)\\ &= \left(\overline{a}+\overline{b}\right)c && (2)\\ &= \overline{a}c+\overline{b}c \end{aligned}$$

(1) d’après une loi de Morgan ; (2) d’après une loi de Morgan.

$$\overline{B} = \overline{\overline{a}+bc} = \overline{\overline{a}}\cdot\overline{bc} = a(\overline{b}+\overline{c}) = a\overline{b}+a\overline{c} = a\overline{c}+a\overline{b}$$

4.28

$$\begin{aligned} (a+b)(a+c) &= aa+ac+ba+bc = a+ac+ab+bc\\ &= a(1+b+c)+bc = a\cdot 1+bc = a+bc \end{aligned}$$

4.29 a) Cette loi est équivalente à OU EXCLUSIF.

b)

Tableau 4.37

a	b	$a*b$	$\overline{a}b$	$a\overline{b}$	$\overline{a}b+a\overline{b}$
0	0	**0**	0	0	**0**
0	1	**1**	1	0	**1**
1	0	**1**	0	1	**1**
1	1	**0**	0	0	**0**

c)

$$\begin{aligned} (a+b)\cdot\left(\overline{a}+\overline{b}\right) &= a\overline{a}+a\overline{b}+\overline{a}b+b\overline{b}\\ &= 0+a\overline{b}+\overline{a}b+0\\ &= a\overline{b}+\overline{a}b \end{aligned}$$

Donc $a*b = (a+b)\cdot\left(\overline{a}+\overline{b}\right)$.

4.30

Tableau 4.38

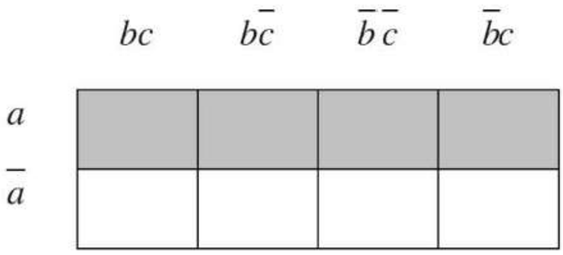
a	b	$\overline{a}\,\overline{b}$	$a\overline{b}$	$\overline{a}\,\overline{b}+a\overline{b}+b$
0	0	1	0	1
0	1	0	0	1
1	0	0	1	1
1	1	0	0	1

Par calculs :

$$\begin{aligned} A &= \overline{b}(\overline{a}+a)+b \\ &= \overline{b}\cdot 1+b \\ &= \overline{b}+b \\ &= 1 \end{aligned}$$

4.31 Pour ab, on colore les deux premières cases de la ligne a. abc est une case déjà colorée, et pour $a\overline{b}$ on colore les deux dernières cases de la ligne a. On obtient $A = a$.

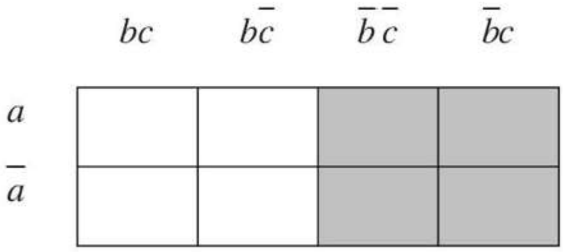
Tableau 4.39



Par calculs : $A = ab + abc + a\overline{b} = ab(1+c) + a\overline{b} = ab + a\overline{b} = a(b+\overline{b}) = a\cdot 1 = a$

4.32

Tableau 4.40



D'après le tableau, $A = \overline{b}$.

$$\begin{aligned} A &= \overline{b}\,\overline{c}(\overline{a}+a)+\overline{b}(ac+\overline{a}) \\ &= \overline{b}\,\overline{c}\cdot 1+\overline{b}(ac+\overline{a}) \\ &= \overline{b}\,(\overline{c}+\overline{a}+ac) \\ &= \overline{b}(\overline{ac}+ac) \\ &= \overline{b}\cdot 1=\overline{b} \end{aligned}$$

4.33 a) $a * a = \overline{a}a + a\overline{a} = 0 + 0 = 0$

b) $a * \overline{a} = \overline{a}\,\overline{a} + a\overline{\overline{a}} = \overline{a} + a = 1$

c) $a * 1 = \overline{a} \cdot 1 + a \cdot \overline{1} = \overline{a} + a \cdot 0 = \overline{a}$

d) $a * 0 = \overline{a} \cdot 0 + a \cdot \overline{0} = 0 + a \cdot 1 = a$ et de même $0 * a = a$ donc 0 est élément neutre pour la loi $*$.

4.34 a) $a \downarrow 0 = \overline{a+0} = \overline{a}$ et $a \downarrow 1 = \overline{a+1} = \overline{1} = 0$

b)

$$
\begin{aligned}
A &= (a \downarrow b) + (a \downarrow a) = \overline{a+b} + \overline{a+a} = \overline{a+b} + \overline{a} \\
&= \overline{a}\,\overline{b} + \overline{a} = \overline{a}(\overline{b} + 1) = \overline{a} \cdot 1 \\
&= \overline{a}
\end{aligned}
$$

c) $(a \downarrow b) \downarrow c = \overline{a+b} \downarrow c = \overline{\overline{a+b} + c} = (a+b)\overline{c} = a\overline{c} + b\overline{c}$

$a \downarrow (b+c) = \overline{a + \overline{\overline{b+c}}} = \overline{a}(b+c) = \overline{a}b + \overline{a}c$

Donc $(a \downarrow b) \downarrow c \neq a \downarrow (b \downarrow c)$

4.35 a)

Tableau 4.41

	er	$e\overline{r}$	$\overline{e}\,\overline{r}$	$\overline{e}r$
a				
$\overline{a}$				

b) D'après le tableau, $F = a\overline{e}$

c) $F = a\overline{e}(r + \overline{r}) = a\overline{e} \cdot 1 = a\overline{e}$

d) Un client de type F est un client qui achète et n'échange pas d'article.

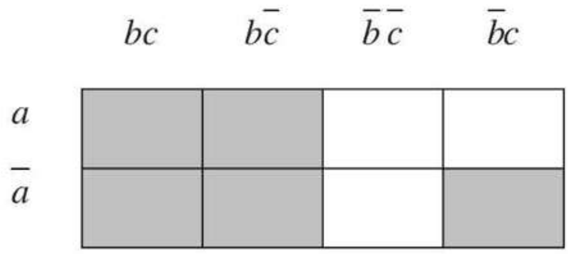
Un client de type $\overline{F}$ est un client qui n'achète pas ou qui échange un article : ce type de client est peu intéressant.

$$\overline{F} = \overline{a\overline{e}} = \overline{a} + e$$

4.36 a) $E = abc + \overline{a}c + \overline{c}b$

b)

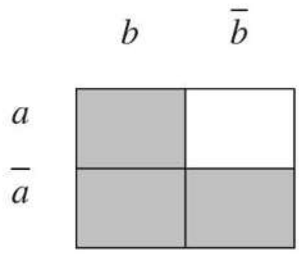
Tableau 4.42



c) Le bloc de quatre cases colorées sur la gauche du tableau correspond aux deux colonnes b. La première et la quatrième case de la deuxième ligne correspondent à $\overline{a}c$ donc $E = b + \overline{a}c$.

d) $E = abc + \overline{a}c + \overline{c}b = c(ab + \overline{a}) + b\overline{c} = c\left(\overline{a} + b\right) + b\overline{c}$. En effet, $ab + \overline{a} = \overline{a} + b$ d'après le tableau de Karnaugh (tableau 4.43).

Tableau 4.43

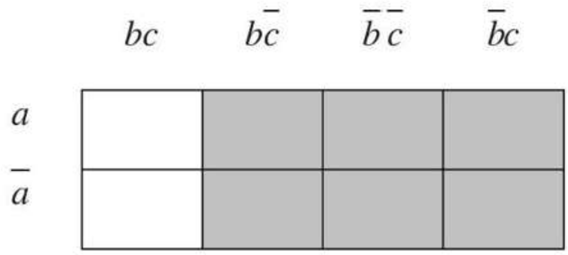


On continue le calcul de E : $E = \overline{a}c + bc + b\overline{c} = \overline{a}c + b\left(c + \overline{c}\right) = \overline{a}c + b$.

e) Les critères simplifiés sont donc : avoir un BTS SIO ou moins de cinq ans d'ancienneté mais maîtriser l'anglais.

4.37 a) Puisque les cases colorées correspondent à E et les non colorées à $\overline{E}$, on voit que $\overline{E} = bc$.

Tableau 4.44



On en déduit que $E = \overline{bc} = \overline{b} + \overline{c}$.

b) 1) On traduit chaque critère avec des variables booléennes :

- Avoir 45 ans ou plus et être au chômage depuis moins d'un an : $a\bar{b}$.
- Avoir moins de 45 ans et ne pas avoir suivi de formation l'année précédente : $\bar{a}\bar{c}$.
- Être au chômage depuis un an ou plus et ne pas avoir suivi de formation l'année précédente : $b\bar{c}$.
- Avoir moins de 45 ans, être au chômage depuis moins d'un an et avoir suivi une formation l'année précédente : $\bar{a}\bar{b}c$.

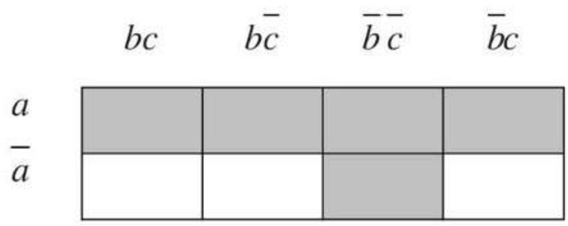
Ainsi $F=\bar{a}\bar{c}+b\bar{c}+a\bar{b}+\bar{a}\bar{b}c$. On constate que $F=E$.

2) Les personnes qui ne pourront pas participer à la formation sont celles qui correspondent à la variable $\bar{F}$. Comme $\bar{F}=bc$, ce sont celles qui ont plus d'un an de chômage et qui ont suivi une qualification l'année précédente.

4.38 a) $E=\bar{b}\bar{c}+a$

b)

Tableau 4.45

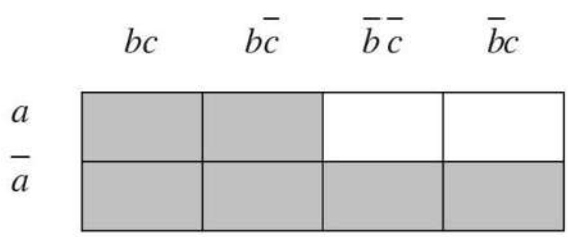


D'après le tableau, $\bar{E}=\bar{a}b+\bar{a}c$. Par calculs, $E=\bar{b}\bar{c}+a$ donc $\bar{E}=\overline{\bar{b}\bar{c}+a}=\overline{\bar{b}\bar{c}}\cdot\bar{a}=(b+c)\bar{a}=\bar{a}b+\bar{a}c$.

4.39 a) $E=\bar{a}c+ab+\bar{c}\bar{a}$

b)

Tableau 4.46



D'après le tableau, $E=\bar{a}+b$. Un client achète un ordinateur s'il a moins d'un an ou s'il est en bon état.

c) $E=\bar{a}c+ab+\bar{c}\bar{a}=\bar{a}(c+\bar{c})+ab=\bar{a}+ab=\bar{a}+b$ (fin de calcul déjà rencontrée).

On en déduit que $\bar{E}=\overline{\bar{a}+b}=a\bar{b}$: un client n'achète pas un ordinateur s'il a plus d'un an et qu'il est en mauvais état.

ENSEMBLES 5

PLAN
5.1 Langage ensembliste
5.2 Relations binaires
5.3 Applications d'un ensemble dans un ensemble

OBJECTIFS
- ➤ Interprétation en termes ensemblistes de la disjonction, de la conjonction de deux relations.
- ➤ Comprendre les relations entre ensembles et leur utilisation.

5.1 LANGAGE ENSEMBLISTE

5.1.1 Généralités

Un ensemble peut être défini en **extension** (on donne alors la liste de ses éléments) ou en **compréhension** (on donne alors une propriété caractéristique de ses éléments, formulée en français ou mathématiquement).

Exemples

Certains ensembles portent des noms qui leur sont réservés : l'ensemble des entiers naturels est $\mathbb{N}$, l'ensemble des nombres réels est $\mathbb{R}$. L'ensemble vide se note $\varnothing$ ou $\{\ \}$.
Soit $A = \{0;3;6;9;12;15;18\}$. A est défini en extension. Pour dire que 12 est un élément de A, on écrit $12 \in A$. Par contre, $13 \notin A$.
On peut aussi dire que A est l'ensemble des multiples de 3 qui sont inférieurs à 20. A est alors défini en compréhension. On peut aussi définir A en compréhension avec une formulation plus mathématique : $A = \{3k,\ k \in \mathbb{N},\ 0 \leq k \leq 6\}$.

Si un ensemble E a un nombre fini d'éléments, on appelle **cardinal** de E le nombre d'éléments de E. On note ce nombre **card(*E*)**.

Exemples

Dans l'exemple précédent, card(A) = 7.
$\mathbb{N}$ et $\mathbb{R}$ sont des ensembles infinis, ils n'ont pas de cardinaux. Card($\varnothing$) = 0.

Soit A et B deux **ensembles**. On dit que A est **inclus** dans B, et on note $A \subset B$, quand tout élément de A est aussi élément de B. Donc : $A \subset B \Leftrightarrow \forall\, x \in A,\, x \in B$.

Si $A \subset B$, on dit que A est un sous-ensemble ou une partie de B. L'ensemble des parties d'un ensemble E se note $\mathcal{P}(E)$. Quel que soit l'ensemble E, on a $E \subset E$ et $\varnothing \subset E$.

Soit A un sous-ensemble de E. On appelle complémentaire de A dans E, que l'on note $\mathcal{C}_E A$ ou $\overline{A}$, l'ensemble de tous les éléments de E qui n'appartiennent pas à A.

$$x \in \mathcal{C}_E A \Leftrightarrow (x \in E \wedge x \notin A)$$

Exemples

Dans $\mathbb{N}$, le complémentaire de l'ensemble des nombres pairs est l'ensemble des nombres impairs.
Dans $\mathbb{R}$, le complémentaire de l'ensemble $[1;+\infty[$ est $]-\infty\,;1[$.

Soit A et B deux ensembles. L'**intersection** de A et de B, que l'on note $A \cap B$, est l'ensemble des éléments qui appartiennent à la fois à A et à B. Donc :

$$x \in A \cap B \Leftrightarrow (x \in A \wedge x \in B)$$

La réunion de A et de B, que l'on note $A \cup B$, est l'ensemble de tous les éléments qui appartiennent à A ou à B (ou non exclusif). Donc :

$$x \in A \cup B \Leftrightarrow (x \in A \vee x \in B)$$

Exemples

$A = \{0;3;6;9;12;15\}$ $\quad$ $B = \{0;4;8;12;16\}$
$A \cap B = \{0;12\}$ $\quad$ $A \cup B = \{0;3;4;6;8;9;12;15;16\}$

Propriété 5.1

Quels que soient les ensembles A, B et C :
$A \cap B = B \cap A$ $\quad$ $A \cup B = B \cup A$
$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ $\quad$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

Ces propriétés montrent que l'intersection et la réunion sont commutatives et distributives l'une par rapport à l'autre.

Propriété 5.2

Soit A une partie d'un ensemble E et soit $\overline{A}$ son complémentaire dans E :
$A \cap \varnothing = \varnothing$ $\quad$ $A \cup \varnothing = A$ $\quad$ $A \cap \overline{A} = \varnothing$ $\quad$ $A \cup \overline{A} = E$

5.1.2 Produit cartésien

Dans le tableau 5.1, les huit cases peuvent être repérées par un couple formé par le numéro de la ligne, suivi du numéro de la colonne. Ainsi la case colorée en gris est repérée par le couple (1;3).

Tableau 5.1

	1	2	3	4
1				
2				

On obtient alors les huit couples : (1;1), (1;2), (1;3), (1;4), (2;1), (2;2), (2;3), (2;4). Ces huit couples sont tous ceux que l'on peut former en mettant un élément de l'ensemble $E = \{1;2\}$ en première position et un élément de $F = \{1;2;3;4\}$ en deuxième position.

L'ensemble de ces huit couples forment un ensemble appelé **produit cartésien** de E et F, noté $E \times F$.

On aurait pu choisir de repérer chaque case par un couple formé par le numéro de la colonne, suivi du numéro de la ligne. On obtient alors huit couples qui forment le produit cartésien de F et de E.

Ces deux produits cartésiens ne sont pas égaux, même s'ils ont le même nombre d'éléments.

Exemple

$(1;3) \in E \times F$ mais $(1;3) \notin F \times E$.

Soit E et F deux ensembles. Le produit cartésien de E et F, que l'on note $E \times F$, est l'ensemble de tous les couples formés par un élément de E en première position, et un élément de F en deuxième position.

$$E \times F = \{(x\,;y),\ x \in E,\ y \in F\}$$

De plus, si E et F sont des ensembles finis, $\text{card}(E \times F) = \text{card}(E) \times \text{card}(F)$.

On peut faire le produit cartésien d'un ensemble avec lui-même. Ce produit cartésien pourra se noter E^2. Cette notation permet de simplifier certains quantificateurs :

- Au lieu d'écrire « $\forall\, x \in \text{E}, \forall\, y \in \text{E}$ » , on peut écrire « $\forall\, (x;y) \in E^2$ ».
- Pour dire qu'il existe deux entiers naturels, on pourra écrire « $\exists\, (x;y) \in \mathbb{N}^2$ ».

On peut généraliser la notion de produit cartésien à plus de deux ensembles avec la définition suivante.

Soit $E_1, E_2, ..., E_n$ n ensembles. Le produit cartésien $E_1 \times E_2 \times ... \times E_n$ est l'ensemble de tous les n-uplets formés par un élément de E_1 en première position, un élément de E_2 en deuxième position, …, un élément de E_n en n-ième position.

$$E_1 \times E_2 \times ... \times E_n = \{(x_1, x_2, ..., x_n),\ x_1 \in E_1, x_2 \in E_2, ... , x_n \in E_n\}$$

De plus, si tous les ensembles sont finis :

$$\text{card}(E_1 \times E_2 \times ... \times E_n) = \text{card}(E_1) \times \text{card}(E_2) \times ... \times \text{card}(E_n)$$

5.2 Relations binaires

5.2.1 Exemple introductif

On examine l'implantation de trois PME informatiques, que l'on notera a, b et c, dans quatre départements, que l'on notera A, C, H et P. Notons E et F les ensembles $\{a, b, c\}$ et $\{A, C, H, P\}$. La situation peut être décrite de plusieurs façons :

• Avec un tableau simple (tableau 5.2) :

Tableau 5.2

Nom de la PME	a	b	c
Départements où elle est présente	A, P	H, P	C

• Avec un tableau à double entrée (tableau 5.3) :

Tableau 5.3

	A	**C**	**H**	**P**
a	oui	non	non	oui
b	non	non	oui	oui
c	non	oui	non	non

• Avec une matrice : en ligne, de haut en bas, les trois PME dans l'ordre a, b, c ; en colonne, de gauche à droite, les quatre départements dans l'ordre A, C, H et P. On code 0 pour non, 1 pour oui.

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

• Avec un diagramme sagittal :

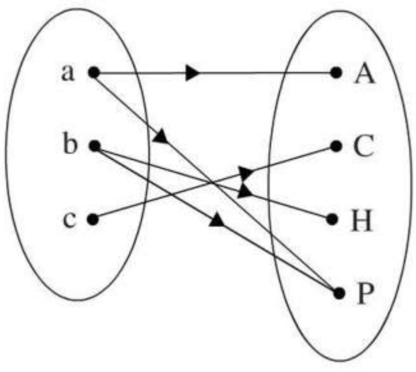


Figure 5.1

- Avec une liste de couples : (a, A), (a, P), (b, H), (b, P), (c, C) qui est une partie du produit cartésien $E \times F$.

Dans cet exemple, nous avons défini une relation « est présente dans le département », entre certains éléments de E (appelé ensemble de départ) et certains éléments de F (appelé ensemble d'arrivée). L'ensemble des couples en relation est appelé **graphe** de la relation.

Si on note $\mathcal{R}$ cette relation, on écrit alors $x \mathcal{R} y$ pour dire que la PME x est présente dans le département y. Par exemple, $a \mathcal{R} P$.

Soit E et F deux ensembles non vides. Une relation binaire $\mathcal{R}$ de E vers F est définie par la donnée du graphe, qui est une partie du produit cartésien $E \times F$: un couple $(x;y)$ appartient au graphe si et seulement si $x \mathcal{R} y$, avec $(x;y) \in E \times F$

E est **l'ensemble de départ**, F est **l'ensemble d'arrivée.**

5.2.2 Relation binaire sur un ensemble

Soit E un ensemble non vide. On peut définir une relation binaire $\mathcal{R}$ sur E par la donnée de son graphe, qui est une partie du produit cartésien E^2. $(x;y)$ appartient au graphe si et seulement si $x \mathcal{R} y$, avec $(x;y) \in E^2$.

Exemples

Deux exemples simples de relations binaires sur des ensembles :

– Dans $\mathbb{R}$, l'inégalité : $x \mathcal{R} y \Leftrightarrow x \leq y$.

– Dans $\mathbb{N}$, la congruence modulo n : $x \mathcal{R} y \Leftrightarrow x \equiv y[n]$ (n est un entier naturel non nul).

Soit $\mathcal{R}$ une relation définie sur un ensemble E non vide :

- $\mathcal{R}$ est réflexive si et seulement si : $\forall\, x \in E, x \mathcal{R} x$.
- $\mathcal{R}$ est symétrique si et seulement si : $\forall\, (x;y) \in E^2, x \mathcal{R} y \Leftrightarrow y \mathcal{R} x$.
- $\mathcal{R}$ est antisymétrique si et seulement si : $\forall\, (x;y) \in E^2, (x \mathcal{R} y \wedge y \mathcal{R} x) \Rightarrow x = y$.
- $\mathcal{R}$ est transitive si et seulement si : $\forall\, (x;y;z) \in E^3, (x \mathcal{R} y \wedge y \mathcal{R} z) \Rightarrow x \mathcal{R} z$.

Exemples

Dans $\mathbb{R}$, la relation d'inégalité :
$x \mathcal{R} y \Leftrightarrow x \leq y$ est réflexive, antisymétrique et transitive.
Dans $\mathbb{N}$, la relation de congruence modulo n :
$x \mathcal{R} y \Leftrightarrow x \equiv y[n]$ est réflexive, symétrique et transitive.

Soit $\mathcal{R}$ une relation définie sur un ensemble E non vide. $\mathcal{R}$ est une relation d'équivalence si et seulement si elle est réflexive, symétrique et transitive.

Exemples

Dans $\mathbb{N}$, la relation de congruence modulo n : $x \mathcal{R} y \Leftrightarrow x \equiv y[n]$ est une relation d'équivalence.
Dans $\mathbb{R}$, la relation d'inégalité : $x \mathcal{R} y \Leftrightarrow x \leq y$ n'est pas une relation d'équivalence car elle n'est pas symétrique.

Soit $\mathcal{R}$ une relation définie sur un ensemble E non vide. $\mathcal{R}$ est une relation d'ordre si et seulement si elle est réflexive, antisymétrique et transitive.

Exemples

Dans $\mathbb{R}$, la relation d'inégalité : $x \mathcal{R} y \Leftrightarrow x \leq y$ est une relation d'ordre.
Dans $\mathbb{N}$, la relation de congruence modulo n : $x \mathcal{R} y \Leftrightarrow x \equiv y[n]$ n'est pas une relation d'ordre car elle n'est pas antisymétrique.

Soit $\mathcal{R}$ une relation d'ordre définie sur un ensemble E non vide. $\mathcal{R}$ est une relation d'ordre total si et seulement si $\forall\ (x;y) \in E^2, (x \mathcal{R} y \vee y \mathcal{R} x)$. Sinon, $\mathcal{R}$ est une relation d'ordre partiel.

Exemple

Dans $\mathbb{R}$, la relation d'inégalité : $x \mathcal{R} y \Leftrightarrow x \leq y$ est une relation d'ordre total car, quels que soient les réels x et y, on a soit $x \leq y$ soit $y \leq x$.

5.3 Applications d'un ensemble dans un ensemble

Une **application d'un ensemble E vers un ensemble F** est une relation binaire de E vers F qui, à tout élément de E, associe un **unique** élément de F.

On note $f: \begin{array}{l} E \to F \\ x \mapsto y \end{array}$ et $f(x) = y$. On dit que y est l'image de x et x est un antécédent de y.

Exemple

Soit $E = \{0;1;2;3\}$, $F = \{0;1;2;3;4;5\}$ et f : $\begin{array}{l} E \to F \\ x \to x^2 \end{array}$.

L'image de 0 est 0, l'image de 1 est 1, l'image de 2 est 4 mais 3 n'a pas d'image car $3^2 \notin$ F. f n'est pas une application de E dans F. Par contre, f est une application de E dans $\mathbb{N}$.

5.3.1 Image directe – Image réciproque

Soit f une application de E dans F et soit A une partie de E. L'image directe de A par f est l'ensemble des images des éléments de A. Cet ensemble se note $f(A)$.

$$f(A) = \{f(x), x \in A\}$$

Exemple

Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = 3x + 1$.
Soit $A = \{0;4;5\}$. L'image directe de A est $f(A) = \{1;13;16\}$.
Soit $A = [2;7]$. $2 \leq x \leq 7 \Leftrightarrow 7 \leq 3x + 1 \leq 22$ donc $f(A) = [7;22]$.

Soit f une application de E dans F et soit B une partie de F. L'**image réciproque** de B par f est l'ensemble des éléments de E qui ont leur image dans B. Cet ensemble se note $f^{-1}(B)$.

$$f^{-1}(B) = \{x \in E, f(x) \in B\}$$

Exemple

Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = 3x + 1$.
Soit $B = [10;16]$, $10 \leq 3x + 1 \leq 16 \Leftrightarrow 9 \leq 3x \leq 15 \Leftrightarrow 3 \leq x \leq 5$ donc $f^{-1}(B) = [3;5]$.

5.3.2 Injection – Surjection – Bijection

Soit f une application de E dans F. f est une **injection** (ou **application injective**) si et seulement si tout élément de F possède au plus un antécédent, ce qui revient à dire que deux éléments distincts de E ont deux images distinctes.

$$\forall x \in E, \forall x' \in E, x \neq x' \Rightarrow f(x) \neq f(x')$$

Exemples

Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = x^2$. f n'est pas injective car on peut trouver des nombres distincts qui ont la même image : par exemple, 3 et -3 ont la même image 9.
Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = 2x - 1$. Pour montrer que f est injective, on va utiliser la contraposée « $f(x) = f(x') \Rightarrow x = x'$ ».
$f(x) = f(x') \Rightarrow 2x - 1 = 2x' - 1 \Rightarrow 2x = 2x' \Rightarrow x = x'$ donc f est injective.

Soit f une application de E dans F. f est une **surjection** (ou **application surjective**) si et seulement si tout élément de F possède au moins un antécédent.

$$\forall y \in F, \exists x \in E, y = f(x)$$

Exemples

Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = x^2$. f n'est pas une surjection car tout nombre strictement négatif n'a pas d'antécédent.
Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = 2x - 1$. f est une surjection car tout nombre réel y possède un antécédent qui est $(y + 1)/2$.

Soit f une application de E dans F. f est une **bijection** (ou **application bijective**) si et seulement si f est à la fois une injection et une surjection.

Exemple

L'application f de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = 2x - 1$ est une bijection puisqu'on a vu qu'elle était une injection et une surjection.

La caractéristique d'une bijection, c'est que chaque élément de l'ensemble de départ a une **seule image**, et chaque élément de l'ensemble d'arrivée a un **seul antécédent**.

En informatique, les injections et les bijections ont un rôle important dans le codage d'informations : en effet, chaque « objet » doit avoir son propre code, car il est très souvent indispensable qu'un même code ne puisse pas être attribué à deux objets distincts.

5.3.3 Composition d'applications

Soit f une application de E dans F et g une application de F dans G. L'application **composée de f par g** est l'application de E dans G qui, à tout élément de E, associe le nombre $g[f(x)]$. Cette application se note $g \circ f$.

$$\forall\, x \in \text{E},\ g \circ f(x) = g[f(x)]$$

Exemple

Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = x^2$ et g l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $g(x) = 0{,}5x + 4$. L'image de 3 par la composée de f par g est :
$g \circ f(3) = g[f(3)] = g(3^2) = g(9) = 0{,}5 \times 9 + 4 = 8{,}5$
$\forall\, x \in \mathbb{R},\ g \circ \text{f}(x) = g[f(x)] = g(x^2) = 0{,}5x^2 + 4.$
L'application $g \circ f$ est donc définie de $\mathbb{R}$ dans $\mathbb{R}$ par $g \circ f(x) = 0{,}5x^2 + 4$.

Nous admettrons les propriétés suivantes.

Propriété 5.3

La composée de deux injections est une **injection**.

La composée de deux surjections est une **surjection**.

La composée de deux bijections est une **bijection**.

5.3.4 Bijection réciproque

Considérons f de $E = \{1;2;3\}$ dans $F = \{a;b;c\}$ définie par : $f(1) = a$; $f(2) = c$; $f(3) = b$. C'est une bijection puisque chaque élément de E a une seule image (donc f est une application), et chaque élément de F a un seul antécédent.

On peut alors définir une **bijection réciproque de f**, de F dans E, que l'on note f^{-1}, telle que :

$$f^{-1}(a) = 1\ ;\ f^{-1}(b) = 3\ ;\ f^{-1}(c) = 2$$

Cette application f^{-1} est aussi une bijection.

Soit f une bijection de E dans F. Il existe une **bijection réciproque** de F dans E, notée f^{-1} telle que :

$$\forall\, (x;y) \in E \times F, f(x) = y \Leftrightarrow f^{-1}(y) = x$$

Il ne faut pas confondre $f^{-1}(B)$ et $f^{-1}(y)$. Dans le premier cas, B est un ensemble et $f^{-1}(B)$ est l'image réciproque de B par f : c'est donc un ensemble. Dans le deuxième cas, y est un élément, f^{-1} désigne une bijection réciproque et $f^{-1}(y)$ est alors l'unique antécédent de y par f : c'est donc un élément.

Propriété 5.4

Soit f une bijection de E dans F et g une bijection de F dans G.
Alors $g \circ f$ est une bijection de E dans G qui possède une bijection réciproque $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$.

Le schéma de la figure 5.2 aide à comprendre les bijections entre ensembles et à retenir la formule précédente.

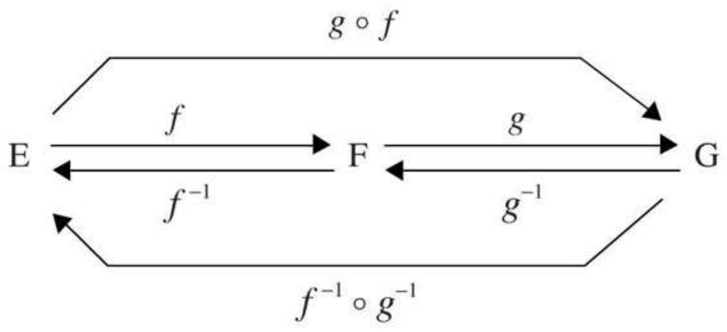


Figure 5.2

TD – Relation binaire dans un ensemble

On donne le dessin de la fig. 5.3, représentant des liens entre des éléments a, b, c et d.

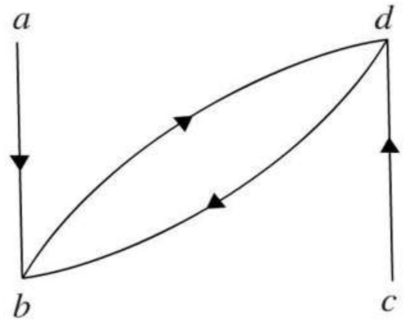


Figure 5.3

Partie A

On considère le dessin comme étant la représentation d'une relation binaire, que l'on note $\mathcal{R}$, définie sur l'ensemble $S = \{a;b;c;d\}$.

a) Écrire tous les éléments qui sont en relation sous la forme $x \mathcal{R} y$, avec $(x;y) \in S^2$.

b) La relation $\mathcal{R}$ est elle réflexive ?

c) La relation $\mathcal{R}$ est elle symétrique ?

d) La relation $\mathcal{R}$ est elle transitive ?

e) Au minimum, quelles flèches doit-on ajouter pour obtenir la représentation d'une relation réflexive ?

f) Même question avec une relation symétrique.

g) Même question avec une relation transitive.

h) Comment doit-on compléter le dessin pour obtenir une relation d'équivalence ?

Partie B

On considère maintenant le dessin comme étant la représentation graphique d'une fonction, que l'on note f, définie sur l'ensemble S.

a) Expliquer pourquoi f est une application.

b) f est elle une injection ?

c) f est elle une surjection ?

d) Quelle est l'image directe de $\{a;b\}$ par f ?

e) Quelle est l'image réciproque de $\{a;b\}$ par f ?

f) Déterminer $f^{-1}[f(\{a;b\})]$ et $f[f^{-1}(\{a;b\})]$.

Partie C

Quel que soit le réel x positif, on appelle partie entière de x, que l'on note $E(x)$, le plus grand entier naturel inférieur ou égal à x. Par exemple : $E(5{,}1) = 5$; $E(10) = 10$; $E(3{,}14) = 3$.

Dans l'ensemble $S = \{3{,}1;2{,}4;4{,}2;4{,}8\}$, on définit une relation binaire $\mathcal{R}$ par :

$$\forall\ (x;y) \in S^2,\ x\,\mathcal{R}\,y \Leftrightarrow (x \neq y) \wedge (2x + y \in \mathbb{N} \vee E(x) = E(y))$$

a) Montrer que $2{,}4\ \mathcal{R}\ 4{,}2$. Est-ce que $2{,}4\ \mathcal{R}\ 4{,}8$?

b) Déterminer tous les couples d'éléments de S qui sont en relation.

c) Faire une représentation graphique de la relation.

d) La relation $\mathfrak{R}$ est-elle réflexive ? Symétrique ? Transitive ?

Solution du TD

Partie A

a) Une flèche relie a à b donc $a\,\mathcal{R}\,b$. On a également $b\,\mathcal{R}\,d$, $d\,\mathcal{R}\,b$ et $c\,\mathcal{R}\,d$.

b) $\mathcal{R}$ n'est pas réflexive car aucun élément n'est en relation avec lui-même (en fait, il suffit qu'il y ait un élément qui ne soit pas en relation avec lui-même).

c) $\mathcal{R}$ n'est pas symétrique. On a, par exemple, $a\,\mathcal{R}\,b$ sans avoir $b\,\mathcal{R}\,a$.

d) $\mathcal{R}$ n'est pas transitive. En effet, on a $a\,\mathcal{R}\,b$ et $b\,\mathcal{R}\,d$ mais on n'a pas $a\,\mathcal{R}\,d$.

e) Il faut ajouter une boucle sur chaque élément.

f) Dès qu'il y a une flèche entre deux éléments, il faut ajouter la flèche « retour ». Il faut donc relier b à a, et d à c.

g) Quand deux flèches se suivent, il faut ajouter la flèche « directe ». Comme on a $a \mathcal{R} b$ et $b \mathcal{R} d$, il faut ajouter la flèche de a à d. De même, puisque l'on a $c \mathcal{R} d$ et $d \mathcal{R} b$, il faut ajouter la flèche $c \mathcal{R} b$.

h) En ajoutant d'abord les boucles, puis les flèches « retour », puis les flèches « directes », on constate qu'il faut relier chaque élément à lui-même et aux trois autres (ce qui fera 16 flèches).

Partie B

a) La flèche qui relie a à b signifie que $f(a) = b$. On a aussi $f(b) = d$, $f(d) = b$, $f(c) = d$. Ainsi, chaque élément a une unique image, la fonction f est donc une application dans S.

b) f n'est pas une injection car a et d ont la même image (b et c aussi).

c) f n'est pas une surjection car a n'a pas d'antécédent (c non plus).

d) $f(a) = b$, $f(b) = d$ donc l'image directe de $\{a;b\}$ est $f(\{a;b\}) = \{b;d\}$.

e) a n'a pas d'antécédent, b a pour antécédents a et d, donc $f^{-1}(\{a;b\}) = \{a;d\}$.

f) $f^{-1}[f(\{a;b\})] = f^{-1}(\{b;d\}) = \{a;b;c;d\} = S$

$f[f^{-1}(\{a;b\})] = f(\{a;d\}) = \{b\}$

Partie C

a) 2,4 est en relation avec 4,2 car $2{,}4 \neq 4{,}2$ et $2 \times 2{,}4 + 4{,}2 = 9$, qui est un entier naturel. Mais 2,4 n'est pas en relation avec 4,8 car on n'a ni $2 \times 2{,}4 + 4{,}8 \in \mathbb{N}$ ni $E(2{,}4) = E(4{,}8)$.

b) On a déjà vu que $2{,}4 \mathcal{R} 4{,}2$. On a aussi $4{,}2 \mathcal{R} 4{,}8$; $4{,}8 \mathcal{R} 4{,}2$ et $3{,}1 \mathcal{R} 4{,}8$.

c)

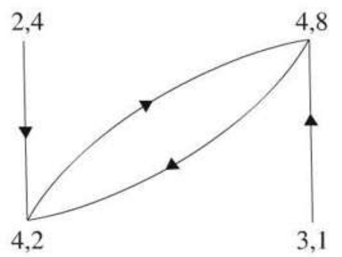


Figure 5.4

d) On retrouve le même graphique que celui étudié en partie A, à part le nom des sommets. La relation $\mathcal{R}$ n'est ni réflexive, ni symétrique, ni transitive.

Exercices corrigés

5.1 Donner en extension l'ensemble A des multiples de 3 compris entre 20 et 40. Définir A en compréhension avec une formulation mathématique.

5.2 Soit $A = \{1;2;4;8;16;32\}$. Définir A en compréhension avec une phrase de français, puis avec une formulation mathématique.

5.3 Soit A l'ensemble des multiples de 7 compris entre 100 et 1 000. Quel est le cardinal de C ?

5.4 Donner en extension l'ensemble A des valeurs affichées par cet algorithme :

```
Variables : I, J entiers
Début
  Pour I de 1 à 3 Faire
    Pour J de 0 à 2 Faire
      Afficher 10I+J
    FinPour
  FinPour
Fin
```

5.5 Donner en extension l'ensemble A des valeurs affichées par cet algorithme :

```
Variables : I, J entiers
Début
  Pour I de 1 à 3 Faire
    Pour J de 0 à 2 Faire
      Afficher I×J
    FinPour
  FinPour
Fin
```

5.6 On considère les ensembles suivants : $E = \{1;2;3;4;5;6;7\}$, $A = \{1;3;5;7\}$, $B = \{2;4;6\}$ et $C = \{1;3;6\}$.

a) Déterminer $A \cap B$, $A \cap C$, $B \cap C$.

b) Déterminer $A \cup B$, $A \cup C$, $B \cup C$.

c) Déterminer $\overline{A}$, complémentaire de A dans E. Déterminer ensuite $\overline{B}$ et $\overline{C}$.

5.7 $A = \{1;4;5;6;8;9\}$, $B = \{2;3;4;5;9\}$ et $C = \{1;2;4;7;10\}$

a) Déterminer $B \cup C$ puis $A \cap (B \cup C)$.

b) Déterminer $A \cap B$, $A \cap C$ puis $(A \cap B) \cup (A \cap C)$.

c) Quelle propriété vient-on d'illustrer ?

5.8 Soit $E=\{0;1\}$. Déterminer $\mathcal{P}(E)$.

5.9 Soit $E=\{a;b;c\}$. Déterminer $\mathcal{P}(E)$.

5.10 A et B étant deux parties d'un ensemble E, on définit l'ensemble $A \Delta B$ (lire A delta B), appelé différence symétrique de A et B, par :

$$A \Delta B = \{x \in E / x \in A \cup B \wedge x \notin A \cap B\}$$

a) Représenter $A \Delta B$ sur un diagramme. Faire la table de vérité de $A \Delta B$.

b) Dans l'ensemble des entiers naturels, on considère les ensembles suivants :
$A = \{1;2;3;4;5;6\}$, $B = \{1;3;5;7\}$ et $C = \{3;4;5;8\}$.
Comparer les ensembles $A \Delta (B \Delta C)$ et $(A \Delta B) \Delta C$.

c) À l'aide d'une table de vérité, montrer que, quels que soient les ensembles A, B et C, on a $A \Delta (B \Delta C) = (A \Delta B) \Delta C$.

5.11 On considère les ensembles $E = \{1;2\}$ et $F = \{a;b;c\}$. Montrer que $E \times F$ et $F \times E$ ont le même cardinal mais que ce sont deux ensembles différents.

5.12 On donne l'algorithme suivant :

```
Variables : i, j, k, S (entiers)
Début
 S prend la valeur 0
 Pour i de 1 à 3 Faire
    Pour j de 1 à 50 Faire
       Pour k de 1 à 12 Faire
          S prend la valeur S+1
       FinPour
    FinPour
 FinPour
 Afficher S
Fin
```

Quelle valeur affiche l'algorithme ? Écrire S comme le cardinal d'un produit cartésien d'ensembles à préciser.

5.13 A et B étant deux ensembles, on définit l'ensemble $A \setminus B$ (lire A moins B) et appelé différence de A et de B, comme étant l'ensemble des éléments de A qui ne sont pas dans B.

a) Écrire une définition de $A \setminus B$ à l'aide de symboles mathématiques. Faire la table de vérité de $A \setminus B$.

b) Soit $A = \{1;2;3;4;5;6\}$, $B = \{1;3;5;7\}$ et $C = \{3;4;5;8\}$. Déterminer $(A \setminus B)$ puis $(A \setminus B) \setminus C$. Déterminer $(B \setminus C)$ puis $A \setminus (B \setminus C)$.
Est-ce que $(A \setminus B) \setminus C = A \setminus (B \setminus C)$?

c) Déterminer $A \setminus (B \cap C)$ puis $(A \setminus B) \cup (A \setminus C)$.

d) À l'aide d'une table de vérité, montrer que, quels que soient les ensembles A, B et C, on a :

$$A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$$

5.14 On dit que des parties non vides d'un ensemble E forment une partition de E lorsque leur réunion est égale à E et qu'elles sont deux à deux disjointes.

a) Soit $E = \{a;b;c;d;e;f;g\}$. Donner un exemple de partition de E en trois parties.

b) Soit $F = \{a;b;c;d\}$. Combien peut-on faire de partitions de F en deux parties ?

Pour chacun des exercices de **5.15** à **5.20**, étudier si la relation $\mathcal{R}$ est réflexive, symétrique, antisymétrique, transitive. Éventuellement, dire si c'est une relation d'équivalence ou d'ordre (total ou partiel).

5.15 Dans $\mathbb{R}$, $x \mathcal{R} y \Leftrightarrow x < y$.

5.16 Dans $\mathbb{N}$, $x \mathcal{R} y \Leftrightarrow x$ est un diviseur de y.

5.17 Dans $\mathbb{N}$, $x \mathcal{R} y \Leftrightarrow x + y$ est multiple de 2.

5.18 Soit E un ensemble non vide. Dans $\mathcal{P}(E)$, $A \mathcal{R} B \Leftrightarrow A \subset B$.

5.19 Soit E un ensemble non vide. Dans $\mathcal{P}(E)$, $A \mathcal{R} B \Leftrightarrow A \cap B = \varnothing$.

5.20 Dans $\mathbb{R}$, $x \mathcal{R} y \Leftrightarrow xy \geq 0$.

5.21 Sur l'ensemble des nombres de l'arbre (appelé arbre binaire) présenté fig. 5.5, on définit une relation binaire par $x \mathcal{R} y$ si et seulement si $x = y$ ou on peut passer de x à y (ou de y à x) par un chemin qui descend toujours à droite.

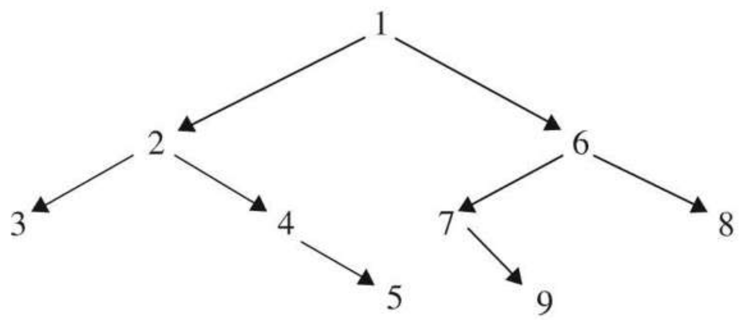


Figure 5.5

a) Expliquer pourquoi, simplement d'après sa définition, on peut affirmer que $\mathcal{R}$ est réflexive et symétrique.

b) Montrer que $\mathcal{R}$ est transitive.

5.22 Soit f l'application de E dans F définie par le diagramme de la fig. 5.6.

a) f est elle injective ? Surjective ?

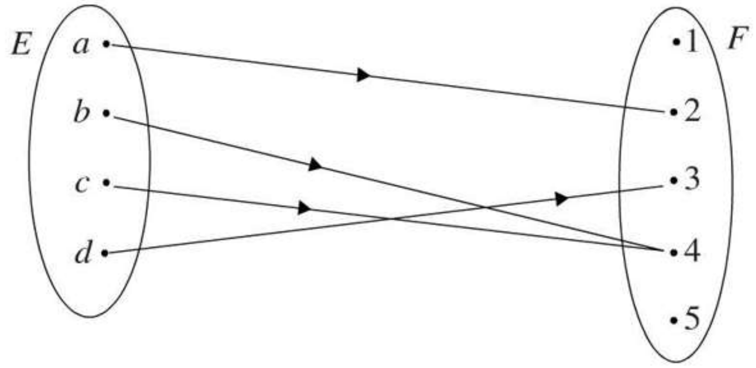


Figure 5.6

b) Soit $A = \{a;b;c\}$ et $A' = \{c;d\}$. Déterminer $f(A)$ et $f(A')$.

c) Déterminer $f(E)$.

d) Soit $B = \{1;2;3\}$ et $B' = \{4\}$. Déterminer $f^{-1}(B)$ et $f^{-1}(B')$.

5.23 $E = \{a;b;c;d\}$, $F = \{1;2;3;4;5\}$. f est l'application de E vers F telle que $f(a) = 2, f(b) = 3, f(c) = 5, f(d) = 3$.

a) f est-elle injective ? Surjective ? Bijective ?

b) Quelle est l'image directe de $A = \{b;c;d\}$ par f ?

c) Quelle est l'image réciproque de $B = \{1;2;3\}$ par f ?

5.24 $E = \{0;1;2;3;4;5;6;7\}$, $F = \{0;1;2;3\}$. f est l'application de E dans F, qui à tout élément de E, associe son reste dans la division euclidienne par 3.

a) f est-elle une injection ? Une surjection ?

b) Soit $A = \{1;3;4\}$, déterminer $f^{-1}[f(\mathrm{A})]$. Est-ce que $f^{-1}[f(\mathrm{A})] = A$?

c) Soit $B = \{2;3\}$, déterminer $f\,[f^{-1}(B)]$. Est-ce que $f\,[f^{-1}(B)] = B$?

5.25 Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$ définie par $f(x) = 4x + 10$.

a) f est-elle une injection ?

b) f est-elle une surjection ?

c) f est-elle une bijection ?

d) Déterminer l'image directe de $[2;3]$et de $[0;+\infty[$.

e) Déterminer l'image réciproque de $[0;+\infty[$.

5.26 $E = \{a;b;c;d\}$, $F = \{1;2;3\}$, $\mathrm{G} = \{\alpha;\beta;\gamma\}$. On définit les applications f de E vers F et g de F vers G de la façon suivante : $f(a) = 2$, $f(b) = 1$, $f(c) = 3$, $f(d) = 2$, $g(1) = \gamma$, $g(2) = \alpha$, $g(3) = \beta$.

a) Les applications f et g sont-elles des injections ? Des surjections ? Des bijections ?

b) Définir l'application $g \circ f$.

c) Peut-on définir l'application réciproque de f ? de g ?

5.27 $E = \{0;1;2;3;4;5;6\}$. On définit les applications f et g de E dans E de la façon suivante :

- $\forall\, x \in E$, $f(x)$ est le reste de la division euclidienne de $3x + 2$ par 7.
- $\forall\, x \in E$, $g(x)$ est le reste de la division euclidienne de $4x + 2$ par 7.

a) Remplir le tableau 5.4.

Tableau 5.4

x	0	1	2	3	4	5	6
$f(x)$							
$g(x)$							
$g \circ f(x)$							

b) Expliquer pourquoi on peut définir les applications réciproques de f, de g et de $g \circ f$.

c) Que vaut $(g \circ f)^{-1}(6)$? Calculer $f^{-1} \circ g^{-1}(6)$ Le résultat était-il prévisible ?

5.28 Soit f et g les applications de $\mathbb{R}$ dans $\mathbb{R}$ définies par $f(x) = 3x$ et $g(x) = 2x + 5$. Définir les applications $g \circ f$ et $f \circ g$.

5.29 Soit f et g les applications de $\mathbb{R}$ dans $\mathbb{R}$ définies par $f(x) = 2x$ et $g(x) = x(x + 1)$. Définir les applications $g \circ f$ et $f \circ g$.

5.30 Soit P le plan rapporté à un repère. Dans ce plan P, on considère l'application qui, à tout point M de coordonnées $(x;y)$ associe le point M' de coordonnées $(x + y\,;\ x - y)$.

a) Soit $\Omega = \{A;B;C;D;E\}$ l'ensemble des cinq points de la figure 5.7. Déterminer l'image de Ω.

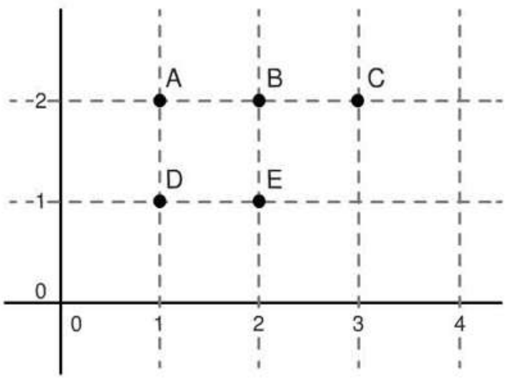


Figure 5.7

b) Soit $F(4;3)$. Déterminer $f^{-1}(\{F\})$.

c) Montrer que, quel que soit le point M se trouvant sur l'axe des abscisses, alors son image M' se trouve sur une droite dont on donnera l'équation.

5.31 On définit une application f de $E = \{-2;-1;0;1;2\}$ dans $F = \{0;1;2;3;4\}$ par $f(x) = x^2$. Dans E, on considère les parties $A = \{-2;-1;0\}$ et $A' = \{0;1\}$.

a) Est-ce que $f(A \cap A') = f(A) \cap f(A')$?

b) Est-ce que $f(A \cup A') = f(A) \cup f(A')$?

c) Est-ce que $f\left(\overline{A}\right) = \overline{f(A)}$?

5.32 On définit une application f de $E = \{-2;-1;0;1;2\}$ dans $F = \{0;1;2;3;4\}$ par $f(x) = x^2$. Dans F, on considère les parties $B = \{0;1;2\}$ et $B' = \{2;4\}$.

a) Est-ce que $f^{-1}(B \cap B') = f^{-1}(B) \cap f^{-1}(B')$?

b) Est-ce que $f^{-1}(B \cup B') = f^{-1}(B) \cup f^{-1}(B')$?

c) Est-ce que $f^{-1}\left(\overline{B}\right) = \overline{f^{-1}(B)}$?

5.33 Pour s'échanger des messages codés, Alice et Bob utilisent leur clavier téléphonique. Le chiffre 2 sert à coder les lettres A, B, C ; le chiffre 3 sert à coder les lettres D, E, F etc.

a) Quel nombre Alice va-t-elle envoyer à Bob pour lui dire BRAVO ?

b) Bob est-il sûr de comprendre ?

c) Quelle propriété de l'application : lettre $\mapsto$ chiffre n'est pas respectée, qui permettrait de décoder le message de façon certaine ?

d) Proposer une adaptation de la méthode permettant d'avoir un décodage unique.

5.34 a) Pour un certain type de codage, on a besoin de coder les lettres A, B, C, D. On commence par attribuer un nombre à chaque lettre : A = 0, B = 1, C = 2, D = 3. Puis on multiplie par 2 le nombre et on ajoute 3. On cherche ensuite le reste de la division euclidienne par 4. À ce reste correspond alors une lettre qui est la lettre cryptée. Comment seront cryptées les lettres A, B, C, D ? Ce type de codage est-il acceptable ?

b) La méthode précédente n'étant pas satisfaisante, on décide de modifier le procédé. On multiplie par 3 le nombre et on ajoute 2. Et on cherche encore le reste de la division euclidienne par 4. Vérifier que l'on a un codage bijectif.

5.35 À chaque lettre, à part le W que l'on fusionne avec le V, on attribue un nombre suivant la correspondance du tableau 5.5.

On code chaque lettre par un couple $(q;r)$, où q et r sont le quotient et le reste dans la division euclidienne par 5 du nombre associé à la lettre. Pour simplifier, on écrira le code sous forme d'un nombre à deux chiffres obtenu en concaténant les nombres q et r.

Tableau 5.5

lettre	A	B	C	D	E	F	G	H	I	J	K	L	M
nombre	0	1	2	3	4	5	6	7	8	9	10	11	12
lettre	N	O	P	Q	R	S	T	U	V,W	X	Y	Z	
nombre	13	14	15	16	17	18	19	20	21	22	23	24	

a) Montrer que la lettre J est codée 14.

b) Comment se codera le mot BRAVO ?

c) Montrer que ce codage est injectif.

Solutions

5.1 $A = \{21;24;27;30;33;36;39\} = \{3k, k \in \mathbb{N}, 7 \leq k \leq 13\}$.

5.2 A est l'ensemble des six premières puissances de 2. $A = \{2k, k \in \mathbb{N}, 0 \leq k \leq 5\}$.

5.3 Le plus petit multiple de 7 supérieurs à 100 est $7 \times 15 = 105$ et le plus grand inférieur à 1 000 est $7 \times 142 = 994$. Donc $A = \{7 \times 15; 7 \times 16; 7 \times 17; ...; 7 \times 141; 7 \times 142\}$. Par conséquent, $\text{card}(A) = 142 - 15 + 1 = 128$.

5.4 $A = \{10;11;12;20;21;22;30;31;32\}$

5.5 Les valeurs successives affichées sont : 0, 1, 2, 0, 2, 4, 0, 3, 6. Donc $A = \{0;1;2;3;4;6\}$.

5.6 a) $A \cap B = \{\ \}$, $A \cap C = \{1;3\}$, $B \cap C = \{6\}$

b) $A \cup B = E$, $A \cup C = \{1;3;5;6;7\}$, $B \cup C = \{1;2;3;4;6\}$.

c) $\overline{A} = \{2;4;6\} = B$, $\overline{B} = A$, $\overline{C} = \{2;4;5;7\}$

5.7 a) $B \cup C = \{1;2;3;4;5;7;9;10\}$, $A \cap (B \cup C) = \{1;4;5;9\}$

b) $A \cap B = \{4;5;9\}$, $A \cap C = \{1;4\}$, $(A \cap B) \cup (A \cap C) = \{1;4;5;9\}$.

c) On vient d'illustrer la formule $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ qui est la distributivité de l'intersection par rapport à la réunion.

5.8 E possède deux éléments donc les parties de E possèdent soit 0, soit 1 soit 2 éléments. Ce sont donc l'ensemble vide, les singletons $\{0\}$ et $\{1\}$, et E lui-même. Donc $\mathcal{P}(E) = \{\varnothing, \{0\}, \{1\}, E\}$.

5.9 $\mathcal{P}(E) = \{\varnothing, \{a\}, \{b\}, \{c\}, \{a;b\}, \{a;c\}, \{b;c\}, E\}$

5.10 a) $A \Delta B$ est la partie colorée de la figure 5.8. La table de vérité est présentée par le tableau 5.6.

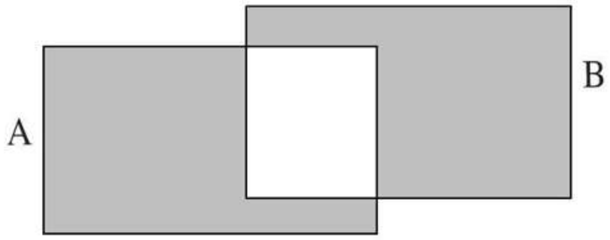


Figure 5.8

Tableau 5.6

A	B	$A \Delta B$
0	0	0
0	1	1
1	0	1
1	1	0

b) $B \Delta C = \{1;4;7;8\}$, $A \Delta (B \Delta C) = \{2;3;5;6;7;8\}$

$A \Delta B = \{2;4;6;7\}$, $(A \Delta B) \Delta C = \{2;3;5;6;7;8\}$

Avec les trois ensembles donnés, $A \Delta (B \Delta C) = (A \Delta B) \Delta C$.

c)

Tableau 5.7

A	B	C	$B \Delta C$	$A \Delta (B \Delta C)$	$A \Delta B$	$(A \Delta B) \Delta C$
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	1	1	1
0	1	1	0	0	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	0	0	0
1	1	1	0	1	0	1

Les colonnes colorées sont les mêmes donc $A \Delta (B \Delta C) = (A \Delta B) \Delta C$ quels que soient les ensembles A, B et C.

5.11 $\text{card}(E \times F) = \text{card}(E) \times \text{card}(F) = 2 \times 3 = 6$ et de même $\text{card}(F \times E) = \text{card}(F) \times \text{card}(E) = 6$.
Mais $(1;a) \in E \times F$ alors que $(1;a) \notin F \times E$ donc $E \times F \neq F \times E$.

5.12 La valeur affichée sera $S = 3 \times 50 \times 12 = 1\,800$. Si $E = \{1;2;3\}$, $F = \{1;2;3;...;49;50\}$, $G = \{1;2;...;12\}$ alors $S = \text{card}(E) \times \text{card}(F) \times \text{card}(G)$

5.13 a) $A \setminus B = \{x, x \in A \wedge x \notin B\}$

Tableau 5.8

A	B	A\B
0	0	0
0	1	0
1	0	1
1	1	0

b) $(A \setminus B) = \{2;4;6\}$ $(A \setminus B) \setminus C = \{2;6\}$

$(B \setminus C) = \{1;7\}$ $A \setminus (B \setminus C) = \{2;3;4;5;6\}$

On constate que $(A \setminus B) \setminus C \neq A \setminus (B \setminus C)$

c) $B \cap C = \{3;5\}$ $A \setminus (B \cap C) = \{1;2;4;6\}$

$(A \setminus B) = \{2;4;6\}$ et $(A \setminus C) = \{1;2;6\}$ donc $(A \setminus B) \cup (A \setminus C) = \{1;2;4;6\}$.

Sur cet exemple, on constate que $A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$.

d)

Tableau 5.9

A	B	C	$B \cap C$	$A \setminus (B \cap C)$	$A \setminus B$	$A \setminus C$	$(A \setminus B) \cup (A \setminus C)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1
1	0	1	0	1	1	0	1
1	1	0	0	1	0	1	1
1	1	1	1	0	0	0	0

D'après la table de vérité, quels que soient les ensembles A, B et C :

$$A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$$

5.14 a) Il suffit de répartir les 6 éléments de E dans trois sous-ensembles, chaque élément ne se trouvant que dans un seul sous-ensemble, sans laisser un sous-ensemble vide. Par exemple, on peut prendre $E_1 = \{1;2\}$, $E_2 = \{3;4;5\}$, $E_3 = \{6\}$.

b) On peut faire des partitions avec une partie avec un seul élément, l'autre partie en ayant alors nécessairement trois : il y a quatre façons de choisir l'élément qui sera seul, donc on a quatre partitions « un élément + trois éléments ». On peut aussi faire des partitions « 2 éléments + 2 éléments ». Il y a trois façons de choisir l'élément avec lequel sera a : avec b, avec c ou avec d. En tout, il y a donc $4 + 3 = 7$ partitions de F en deux parties.

5.15 $\mathcal{R}$ n'est pas réflexive car on n'a pas $x \mathcal{R} x$ pour tout x réel (par exemple, on n'a pas $1 < 1$).

$\mathcal{R}$ n'est pas symétrique car $x \mathcal{R} y$ n'implique pas $y \mathcal{R} x$ (par exemple, $2 < 3$ mais on n'a pas $3 < 2$).

$\mathcal{R}$ est transitive car, pour tous nombres x, y et z, si on a $x < y$ et $y < z$ alors $x < z$.

5.16 $\mathcal{R}$ est réflexive car tout nombre entier naturel est diviseur de lui-même.

$\mathcal{R}$ est antisymétrique car si x divise y et y divise x, alors $x = y$.

$\mathcal{R}$ est transitive car si x divise y et y divise z, alors x divise z.

La relation $\mathcal{R}$ est donc une relation d'ordre. Et c'est un ordre partiel, car pour deux entiers naturels quelconques x et y, on n'a pas forcément x qui divise y ou y qui divise x.

5.17 $\mathcal{R}$ est réflexive car pour tout entier naturel x, $x + x = 2x$ est multiple de 2.

$\mathcal{R}$ est symétrique car si $x + y$ est multiple de 2, alors $y + x$ aussi.

$\mathcal{R}$ est transitive car si $x + y$ est multiple de 2 et $y + z$ aussi, alors $x + y = 2k$, avec $k \in \mathbb{N}$, $y + z = 2k'$, avec $k' \in \mathbb{N}$, et on a alors $x + y + y + z = 2k + 2k'$ donc $x + z = 2(k + k' - y)$ qui est un multiple de 2.

La relation $\mathcal{R}$ est donc une relation d'équivalence.

5.18 $\mathcal{R}$ est réflexive car tout ensemble est inclus dans lui-même.

$\mathcal{R}$ est antisymétrique car si $A \subset B$ et $B \subset A$ alors $A = B$.

$\mathcal{R}$ est transitive car si $A \subset B$ et $B \subset C$ alors $A \subset C$.

La relation $\mathcal{R}$ est donc une relation d'ordre. L'ordre est partiel, car pour deux parties quelconques de $\mathcal{P}(E)$, il n'y en a pas forcément une incluse dans l'autre.

5.19 $\mathcal{R}$ n'est pas réflexive car pour tout A, $A \cap A = A \neq \varnothing$.

$\mathcal{R}$ est symétrique car si $A \cap B = \varnothing$ alors $B \cap A = \varnothing$.

$\mathcal{R}$ n'est pas transitive car $A \cap B = \varnothing$ et $B \cap C = \varnothing$ n'implique pas nécessairement $A \cap C = \varnothing$ (contre-exemple avec $A = \{1;2\}$, $B = \{3;4\}$, $C = \{1;5\}$).

5.20 $\mathcal{R}$ est réflexive car pour tout x réel, $x^2 \geq 0$.

$\mathcal{R}$ est symétrique car si $xy \geq 0$ alors $yx \geq 0$.

$\mathcal{R}$ est transitive car si $xy \geq 0$ et $yz \geq 0$ alors $xyyz \geq 0$ d'où $xy^2z \geq 0$ puis $xz \geq 0$.

5.21 a) Soit $E = \{1;2;3;4;5;6;7;8;9\}$. Tout élément de E est en relation avec lui-même d'après la partie « $x = y$ » de la définition, donc $\mathcal{R}$ est réflexive. Si $x \mathcal{R} y$ alors $y \mathcal{R} x$ d'après la partie « … de x à y (ou de y à x) … » de la définition donc $\mathcal{R}$ est symétrique.

b) On doit montrer que $\forall (x;y;z) \in E^3$, $x \mathcal{R} y$ et $y \mathcal{R} z \Rightarrow x \mathcal{R} z$. Les cas où deux nombres sont égaux sont évidents. Il suffit d'examiner tous les cas où les trois nombres sont différents. On a $1 \mathcal{R} 6$ et $6 \mathcal{R} 8 \Rightarrow 1 \mathcal{R} 8$ et $2 \mathcal{R} 4$ et $4 \mathcal{R} 5 \Rightarrow 2 \mathcal{R} 5$ donc $\mathcal{R}$ est transitive. On peut donc dire que $\mathcal{R}$ est une relation d'équivalence.

5.22 a) f n'est pas injective car b et c, qui sont deux éléments distincts de E, ont la même image. f n'est pas surjective car des éléments de F n'ont pas d'antécédent (les éléments 1 et 5).

b) $f(a) = 2$, $f(b) = 4$, $f(c) = 4$, donc $f(A) = \{2;4\}$.

$f(c) = 4$, $f(d) = 3$, donc $f(A') = \{3;4\}$.

c) $f(E) = \{2;3;4\}$.

d) 1 n'a pas d'antécédent, 2 a pour antécédent a et 3 a pour antécédent d donc $f^{-1}(B) = \{a;d\}$.

3 a pour antécédent d et 4 a pour antécédents b et c donc $f^{-1}(B') = \{b;c;d\}$.

5.23 a) f est une injection car deux éléments distincts de E ont des images distinctes. f n'est pas une surjection car des éléments de F n'ont pas d'antécédent (1 et 4). f n'étant pas surjective, elle n'est pas une bijection.

b) $f(A) = \{3;5\}$

c) $f^{-1}(B) = \{a;b;d\}$

5.24 $f(0) = f(3) = f(6) = 0$, $f(1) = f(4) = f(7) = 1$, $f(2) = f(5) = 2$.

a) f n'est pas injective car on peut trouver des éléments distincts de E qui ont la même image. f n'est pas surjective car 3 n'a pas d'antécédent.

b) $f(A) = \{0;1\}$ et $f^{-1}[f(\mathrm{A})] = \{0;1;3;4;6;7\}$. Donc $f^{-1}[f(\mathrm{A})] \neq A$.

c) $f^{-1}(B) = \{2\,;5\}$ et $f\,[f^{-1}(B)] = \{2\}$. Donc $f\,[f^{-1}(B)] \neq B$.

5.25 a) On utilise la contraposée de la définition :

$$\forall (x;y) \in \mathbb{R}^2, f(x) = f(x') \Leftrightarrow 4x + 10 = 4x' + 10 \Leftrightarrow 4x = 4x' \Leftrightarrow x = x'$$

f est donc une injection.

b) $\forall\, y \in \mathbb{R}$, $y = 4x + 10 \Leftrightarrow y - 10 = 4x \Leftrightarrow x = (y - 10)/4 = 0{,}25y - 2{,}5$

Tout réel y possède donc un antécédent qui est $0{,}25y - 2{,}5$. f est donc une surjection.

c) f est une bijection puisqu'elle est à la fois une injection et une surjection.

d) $x \in [2\,;3] \Leftrightarrow 2 \leq x \leq 3 \Leftrightarrow 8 \leq 4x \leq 12 \Leftrightarrow 18 \leq 4x + 10 \leq 22 \Leftrightarrow 4x + 10 \in [18\,;22]$.
Ainsi, $f([2;3]) = [18;22]$.

$x \in [0;+\infty[\ \Leftrightarrow 0 \leq x \Leftrightarrow 0 \leq 4x \Leftrightarrow 10 \leq 4x + 10 \Leftrightarrow 4x + 10 \in [10;+\infty[$.
Ainsi, $f([0;+\infty[) = [10;+\infty[$.

e) $\forall\, y \in \mathbb{R}$, $4x + 10 \geq 0 \Leftrightarrow 4x \geq -1 \Leftrightarrow x \geq -2{,}5 \Leftrightarrow x \in [2{,}5\,;+\infty[$.
Ainsi, $f^{-1}([0;+\infty[) = [-2{,}5;+\infty[$.

5.26 a) f n'est pas une injection car les éléments a et d ont la même image. f est une surjection car tous les éléments de F ont au moins un antécédent. Puisque f n'est pas injective, ce n'est pas une bijection.

g est une bijection car chaque élément de F a une seule image et chaque élément de G a un seul antécédent.

b) $g \circ f$ est l'application de E dans G, définie par le tableau 5.10.

Tableau 5.10

x	a	b	c	d
$g \circ f(x)$	α	γ	β	α

(exemple de calcul : $g \circ f(a) = g[f(a)] = g(2) = \alpha$)

c) On ne peut pas définir l'application réciproque de f car f n'est pas bijective. Par contre, on peut définir l'application réciproque g^{-1} de g, et cette réciproque est elle aussi bijective.

g^{-1} est définie de G dans F par $g^{-1}(\alpha) = 2$, $g^{-1}(\beta) = 3$, $g^{-1}(\gamma) = 1$.

5.27 a)

Tableau 5.11

x	0	1	2	3	4	5	6
$f(x)$	2	5	1	4	0	3	6
$g(x)$	1	5	2	6	3	0	4
$g \circ f(x)$	2	0	5	3	1	6	4

b) D'après le tableau 5.11, on voit que f et g sont des bijections, donc elles possèdent des bijections réciproques. La composée de deux bijections est une bijection, donc $g \circ f$ est une bijection, qui possède une bijection réciproque.

c) $g \circ f(5) = 6$ donc $(g \circ f)^{-1}(6) = 5$ et $f^{-1} \circ g^{-1}(6) = f^{-1}\lfloor g^{-1}(6) \rfloor = f^{-1}(3) = 5$.

On constate que $(g \circ f)^{-1}(6) = f^{-1} \circ g^{-1}(6)$. Ce résultat était prévisible puisque c'est une propriété générale : la bijection réciproque de $g \circ f$ est $f^{-1} \circ g^{-1}$.

5.28 $\forall x \in \mathbb{R}, g \circ f(x) = g(3x) = 2(3x) + 5 = 6x + 5$

$\forall x \in \mathbb{R}, f \circ g(x) = f(2x + 5) = 3(2x + 5) = 6x + 15$

5.29 $\forall x \in \mathbb{R}, g \circ f(x) = g(2x) = 2x(2x + 1) = 4x^2 + 2x$

$\forall x \in \mathbb{R}, f \circ g(x) = f(x(x + 1)) = 2x(2x + 1) = 2x^2 + 2x$

5.30 a) A a pour coordonnées $(1;2)$. L'image de A est le point A' de coordonnées $(1 + 2;1 - 2) = (3;1)$. De même, on lit les coordonnées des autres points : $B(2;2)$, $C(3;2)$, $D(1;1)$, $E(2;1)$ et on calcule les coordonnées des points images : $B'(4;0)$, $C'(5;1)$, $D'(2;0)$, $E'(3;1)$.

$$f(\Omega) = \{A';B';C';D';E'\}$$

b) $(x+y;\, x-y) = (4;\, 3) \Leftrightarrow \begin{cases} x+y=4 \\ x-y=3 \end{cases} \Leftrightarrow \begin{cases} x=3.5 \\ y=0.5 \end{cases}$

$f^{-1}(\{F\}) = \{G(3,5;0,5)\}$

c) Si M appartient à l'axe des abscisses, ses coordonnées sont $(x;0)$. Alors M' a pour coordonnées $(x;x)$ et appartient toujours à la droite d'équation $y = x$.

5.31 a) $A \cap A' = \{0\}$ donc $f(A \cap A') = f(\{0\}) = \{0\}$.

$f(A) = \{0;1;4\}, f(A') = \{0;1\}$ donc $f(A) \cap f(A') = \{0;1\}$.

Donc $f(A \cap A') \neq f(A) \cap f(A')$.

b) $A \cup A' = \{-2;-1;0\,;1\}$ donc $f(A \cup A') = \{0;1;4\}$.

Et $f(A) \cup f(A') = \{0;1;4\}$ donc pour cet exemple, $f(A \cup A') = f(A) \cup f(A')$.

c) $\overline{A} = \{1;2\}$ donc $f(\overline{A}) = \{1;4\}$.

$f(A) = \{0;1;4\}$ donc $\overline{f(A)} = \{2;3\}$donc $f(\overline{A}) \neq \overline{f(A)}$.

5.32 a) $B \cap B' = \{2\}$ donc $f^{-1}(B \cap B') = f^{-1}(\{2\}) = \varnothing$

$f^{-1}(B) = \{-1;0;1\}, f^{-1}(B') = \{-2;2\}$ donc $f^{-1}(B) \cap f^{-1}(B') = \varnothing$.

Pour cet exemple d'application, $f^{-1}(B \cap B') = f^{-1}(B) \cap f^{-1}(B')$.

b) $B \cup B' = \{0;1;2;4\}$ donc $f^{-1}(B \cup B') = \{-2;-1;0;1;2\} = E$. On trouve également $f^{-1}(B) \cup f^{-1}(B') = E$. Pour cet exemple, $f^{-1}(B \cup B') = f^{-1}(B) \cup f^{-1}(B')$.

c) $\overline{B} = \{-2;-1\}$ donc $f^{-1}(\overline{B}) = \varnothing$. $f^{-1}(B) = \{-1;0;1\}$ donc $\overline{f^{-1}(B)} = \{-2;2\}$.

Donc $f^{-1}(\overline{B}) \neq \overline{f^{-1}(B)}$.

5.33 a) Alice va envoyer 27 286.

b) Non, car il peut interpréter le 2 comme un A, un B, un C.

c) Le codage n'est pas injectif (et par conséquent pas bijectif).

d) On peut écrire : $A = 2$, $B = 22$, $C = 222$, D = *3*, $E = 33$, $F = 333$, etc.

5.34 a) Ce codage n'est pas acceptable car il n'est pas injectif, et *a fortiori* pas bijectif.

Tableau 5.12

Lettre	A	B	C	D
Valeur x	0	1	2	3
$2x + 3$	3	5	7	9
Reste	3	1	3	1
Code	D	B	D	B

b) Cette fois, on obtient un codage bijectif.

Tableau 5.13

Lettre	A	B	C	D
Valeur x	0	1	2	3
$2x + 3$	2	5	8	11
Reste	2	1	0	3
Code	C	B	A	D

5.35 a) J = 9. Dans la division euclidienne par 5, le quotient est 1 et le reste est 4. Donc le code est (1;4) que l'on écrit plus simplement 14.

b) On trouve facilement que B se code 01, R se code 32, A se code 00, V se code 41 et O se code 24 donc BRAVO se code 0132004124.

c) Ce codage est injectif puisque, pour chaque nombre entier, le couple (quotient ; reste) est unique.

Le fait de fusionner V et W permet d'avoir 25 lettres, ce qui peut servir pour présenter la grille de codage sous la forme d'un tableau (voir tableau 5.14). Chaque lettre est alors repérée par un couple (ligne ; colonne).

Tableau 5.14

	0	1	2	3	4
0	A	B	C	D	E
1	F	G	H	I	J
2	K	L	M	N	O
3	P	Q	R	S	T
4	U	V,W	X	Y	Z

GRAPHES ET ORDONNANCEMENT

6

PLAN

6.1 Représentations d'un graphe
6.2 Chemins d'un graphe
6.3 Niveau des sommets d'un graphe sans circuit
6.4 Méthode MPM d'ordonnancement d'un graphe

OBJECTIFS

➤ Savoir mettre en œuvre des algorithmes de recherche de niveaux, d'optimisation et de constructions de graphes ordonnés.

6.1 Représentations d'un graphe

Un site internet est composé de cinq pages notées A, B, C, D et E. En un clic, on peut passer d'une page à certaines autres selon les possibilités suivantes :

- De la page A, on peut passer en un clic aux pages C et E.
- De la page B, on peut passer aux pages A et D.
- Depuis la page C, on peut accéder à la page B ou rester sur C.

Quand on est sur la page D, on peut seulement aller sur la page C et de la page E, on ne peut aller que sur la page A.

On peut modéliser cette situation avec un graphique sagittal (fig. 6.1), constitué de l'ensemble $S = \{A;B;C;D;E\}$ des **sommets**, et de flèches orientées appelées **arcs.**

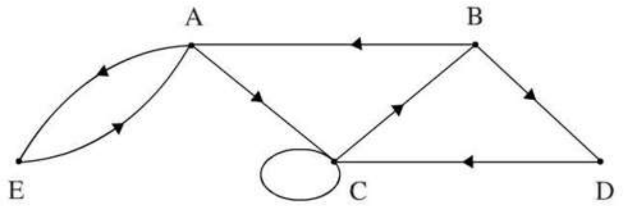


Figure 6.1

On peut représenter chaque flèche par un couple (X,Y) où X est le point de départ de la flèche et Y son point d'arrivée. On obtient ainsi une liste de couples, qui est une partie du produit cartésien S^2 : $(A;C)$, $(A;E)$, $(B;A)$, $(B;D)$, $(C;B)$, $(C;C)$, $(D;C)$, $(E;A)$.

On peut aussi faire un tableau donnant, pour chaque page, les pages qui peuvent être atteintes par un seul clic. On fait alors le tableau des **successeurs** (tableau 6.1, à gauche). Une autre possibilité est de faire le tableau des **prédécesseurs** (tableau 6.1, à droite), c'est-à-dire un tableau donnant, pour chaque page, les pages qui ont permis d'y arriver avec un seul clic.

Tableau 6.1

Page (sommet)	Successeurs
A	C, E
B	A, D
C	B, C
D	C
E	A

Page (sommet)	Prédécesseurs
A	B, E
B	C
C	A, C, D
D	B
E	A

On peut aussi faire un tableau à double entrée en codant 1 s'il existe une possibilité en un seul clic, d'aller d'une page de départ (origine) à une page d'arrivée (extrémité), et en codant 0 s'il n'y a pas possibilité de passage.

Tableau 6.2

		Extrémité				
		A	B	C	D	E
Origine	A	0	0	1	0	1
	B	1	0	0	1	0
	C	0	1	1	0	0
	D	0	0	1	0	0
	E	1	0	0	0	0

De façon évidente, le tableau 6.2 peut être associé à la matrice carrée d'ordre 5 suivante, appelée matrice d'adjacence (fig. 6.2).

$$\begin{array}{c|ccccc} & A & B & C & D & E \\ A & 0 & 0 & 1 & 0 & 1 \\ B & 1 & 0 & 0 & 1 & 0 \\ C & 0 & 1 & 1 & 0 & 0 \\ D & 0 & 0 & 1 & 0 & 0 \\ E & 1 & 0 & 0 & 0 & 0 \end{array}$$

Figure 6.2

6.1.1 Vocabulaire des graphes

Un **graphe simple orienté** est défini par un ensemble fini $S = \{a_1 ;a_2 ;... ;a_n\}$ dont les éléments sont appelés **sommets**, et par un ensemble G de couples appartenant au produit cartésien S^2 dont les éléments sont appelés **arcs**.

Un couple $(a_i;a_j)$ appartient à G si et seulement si, sur le diagramme sagittal, on peut relier le sommet a_i au sommet a_j par une flèche orientée.

Le sommet a_i de l'arc est appelé **origine** et le sommet a_j est appelé **extrémité**. Si l'origine et l'extrémité sont confondues, l'arc est une **boucle**.

Soit $(a_i;a_j)$ un arc d'un graphe G :

- Le sommet a_j est un **successeur** de a_i et a_i **est un prédécesseur** de a_j.
- L'ensemble des successeurs d'un sommet a_i se note $\Gamma^+ (a_i)$.
- L'ensemble des prédécesseurs d'un sommet a_j se note $\Gamma^- (a_j)$.

Exemple

Reprenons l'exemple étudié en début de chapitre :
- Les successeurs de A sont C et E donc $\Gamma^+ (A) = \{C;E\}$. De même, $\Gamma^+ (B) = \{A;D\}$.
- Les prédécesseurs de A sont B et E donc $\Gamma^- (A) = \{B;E\}$. De même, $\Gamma^- (B) = \{C\}$.

6.1.2 Matrice d'adjacence d'un graphe

Soit G un graphe orienté ayant n sommets $a_1, a_2, ..., a_n$. La **matrice d'adjacence** du graphe est la matrice carrée $M = (m_{i,j})$ d'ordre n telle que : $m_{i,j} = 1$ si et seulement si $(a_i;a_j)$ est un arc de G, et $m_{i,j} = 0$ sinon.

Exemple

Soit $M = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

la matrice d'adjacence d'un graphe définie sur $S = \{A;B;C;D\}$.

La première ligne montre que $\Gamma^+(A) = \{C;D\}$. La troisième ligne montre que C n'a pas de successeur. On peut ainsi reconstituer le graphe :

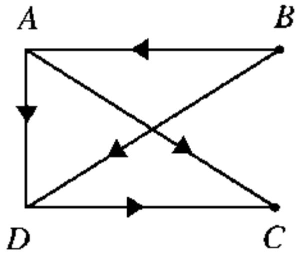


Figure 6.3

Le nombre de 1 sur une ligne est le nombre de successeurs du sommet correspondant à la ligne. Concernant les colonnes, le nombre de 1 est le nombre de prédécesseurs du sommet qui correspond à la colonne.

6.2 Chemins d'un graphe

Reprenons le graphe du paragraphe 6.1.

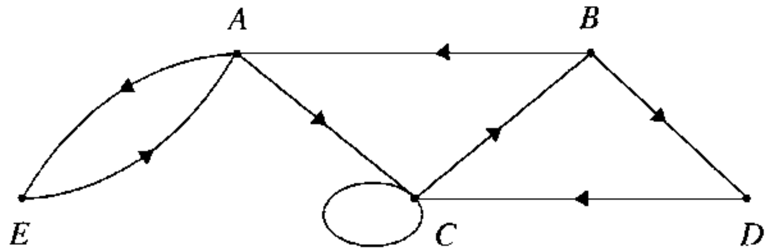


Figure 6.4

En suivant des arcs, on peut aller du sommet E au sommet B en passant par les sommets A puis C.

La suite ordonnée (E,A,C,B) est un **chemin** constitué de trois arcs. Ce chemin est de longueur 3. On peut remarquer que chaque arc de ce chemin peut être considéré comme un chemin de longueur 1.

(A,B,D) n'est pas un chemin car il n'y a pas d'arc qui permet de passer de A à B.

(D,C,B,D) est un chemin dont le premier sommet est le même que le dernier sommet : c'est un **circuit**.

(E,A,B,C,D) est un chemin qui passe par tous les sommets, une fois et une seule : on dit que c'est un **chemin hamiltonien.**

6.2.1 Chemin et longueur

Dans un graphe orienté, un **chemin** est une suite ordonnée de sommets dans laquelle chaque sommet, à part le premier, est un successeur du sommet qui le précède.

Un **chemin hamiltonien** est un chemin qui passe une seule fois par tous les sommets du graphe.

Un **circuit** est un chemin dans lequel le premier et le dernier sommet sont identiques.

Exemple

Considérons le graphe présenté fig. 6.5.

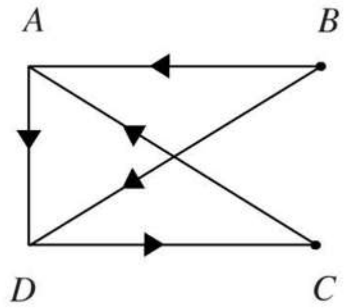


Figure 6.5

(B,D,C) et (B,A,D) sont des chemins, (A,C,D) n'en est pas un.
(A,D,C,A) est un circuit car c'est un chemin, et le premier sommet est le même que le dernier.
(B,A,D,C) est un chemin hamiltonien. (B,D,C,A) en est un autre.

La **longueur d'un chemin** est égale au nombre d'arcs qui constituent ce chemin. La longueur d'un chemin ayant n sommets est $n - 1$.

Exemple

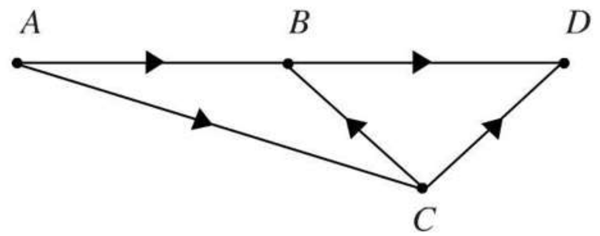


Figure 6.6

Pour aller de A à D, il y a un chemin de longueur 2 : le chemin (A,B,D).
Il y a aussi un chemin hamiltonien, de longueur 3 : le chemin (A,C,B,D).

6.2.2 Nombre de chemins de longueur donnée

Propriété 6.1

Soit M la matrice d'adjacence d'un graphe orienté à n sommets $a_1, a_2, ..., a_n$.

Soit p un entier et $M^p = (m_{i,j})$ la puissance d'exposant p de la matrice M.

Alors $m_{i,j}$ est le nombre de chemins de longueur p allant du sommet a_i au sommet a_j.

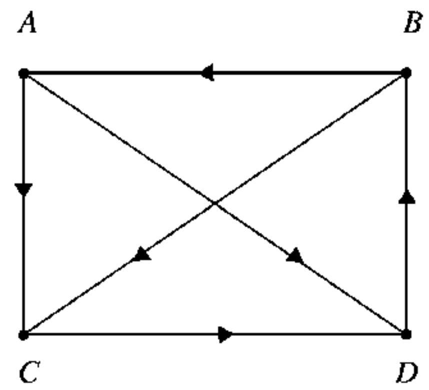


Figure 6.7

La matrice d'adjacence du graphe présenté fig. 6.7 est $M = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$.

Prenons $p = 2$. Les calculs matriciels nous donnent $M^2 = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$.

Considérons la deuxième ligne de la matrice M^2. Elle concerne le sommet B du graphe. Les deux premiers coefficients sont 0, cela veut dire qu'il n'y a pas de chemin de longueur 2 qui relie B à A, ou B à B. Le troisième coefficient est 1, donc il y a un chemin de longueur 2 qui relie B à C : le chemin (B,A,C).

Enfin, le dernier coefficient est 2, attestant l'existence de deux chemins de longueur 2 reliant B à D : les chemins (B,A,D) et (B,C,D).

Prenons maintenant $p = 3$: $M^3 = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 2 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 2 \end{pmatrix}$.

Cette fois, la matrice M^3 permet de connaître le nombre de chemins de longueur 3 qui relient un sommet à un autre.

Par exemple, puisque le coefficient $m_{3,1}$ de la matrice M^3 vaut 1, cela signifie qu'il y a un chemin de longueur 3 qui relie le sommet C au sommet A : c'est le chemin (C,D,B,A).

6.2.3 Opérations sur les matrices booléennes

Les matrices d'adjacences ne comportant que des 0 et des 1, on peut appliquer sur leurs coefficients l'addition et la multiplication booléennes.

Exemple

Soit $M = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$ et $M' = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

L'addition booléenne des deux matrices se note $M \oplus M'$ et vaut $M \oplus M' = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$.

En pratique, on peut calculer $M + M' = \begin{pmatrix} 0 & 2 & 0 \\ 2 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$ et rendre cette matrice booléenne en gardant les 0 et les 1, et en remplaçant par 1 tous les coefficients supérieurs à 1.

On peut aussi faire la multiplication booléenne des matrices, qui se note $M \otimes M'$.

En pratique, il est plus simple de calculer $MM' = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}$ et de rendre la matrice booléenne.

Ainsi, $M \otimes M' = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}$.

On peut également définir des puissances booléennes, notées $M^{[p]}$.

$M^2 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ donc $M^{[2]} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$

Soit M et M' deux matrices booléennes :

- La **somme booléenne** de M et M' est notée $M \oplus M'$: elle est obtenue en faisant les sommes booléennes des coefficients des deux matrices.
- On peut définir de même le **produit booléen** noté $M \otimes M'$.
- La **puissance booléenne** d'exposant p de la matrice M se note $M^{[p]}$ et vaut $M \otimes M \otimes ... \otimes M$, produit de p matrices égales à M.

Propriété 6.2

Soit M la matrice d'adjacence d'un graphe ayant n sommets $a_1, a_2, ..., a_n$ et soit $M^{[p]} = (m_{i,j})$ la matrice booléenne d'exposant p.

Si $m_{i,j} \neq 0$ alors il existe au moins un chemin de longueur p reliant le sommet a_i au sommet a_j.

Exemple

Avec $M = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$, on a $M^{[4]} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}$ car $M^4 = \begin{pmatrix} 2 & 3 & 0 \\ 3 & 5 & 0 \\ 1 & 2 & 0 \end{pmatrix}$

La matrice booléenne permet d'affirmer qu'il y a au moins un chemin de longueur 4 entre *A* et *A*, ainsi qu'entre *A* et *B*. Il en est de même entre *B* et *A*, *B* et *B*, *C* et *A*, *C* et *B*.

6.2.4 Fermeture transitive d'un graphe

Faire la **fermeture transitive** d'un graphe consiste à rajouter tous les arcs (a_i, a_j) dès qu'il existe un chemin allant du sommet a_i au sommet a_j.

Exemple

Sur le graphe présenté fig. 6.8, il existe un chemin allant de *A* à *C* en passant par *B* donc on ajoute l'arc (*A*,*C*).

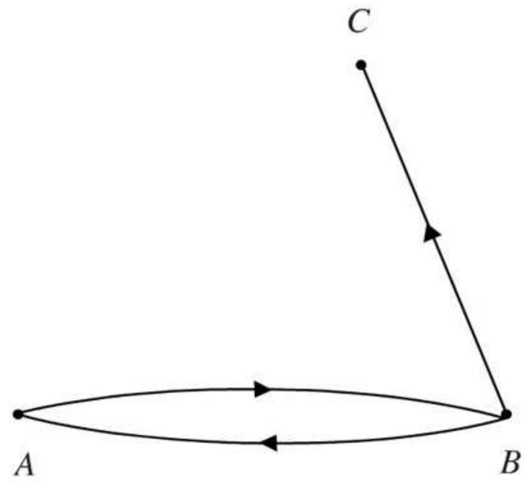


Figure 6.8

De même, on constate qu'il faut ajouter les boucles (*A*,*A*) et (*B*,*B*). On obtient le graphe complété fig. 6.9, sur lequel il ne reste aucun chemin sans l'arc « direct » correspondant.

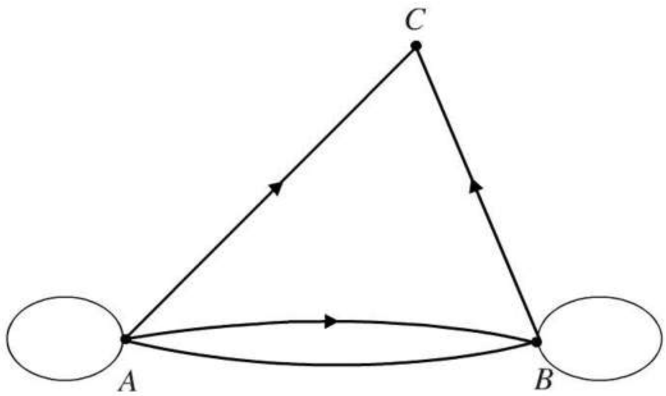


Figure 6.9

Ce nouveau graphe est la fermeture transitive du précédent.

Il peut être difficile de faire la fermeture transitive d'un graphe sans oublier d'arcs. Mais en utilisant la matrice d'adjacence d'un graphe, et au prix de quelques calculs matriciels, on peut déterminer la matrice d'adjacence de la fermeture transitive. Il est alors plus facile de ne pas oublier d'arcs.

Nous admettrons la propriété suivante.

Propriété 6.3

Soit M la matrice d'adjacence d'un graphe à n sommets et soit $\hat{M}$ la matrice d'adjacence de la fermeture transitive du graphe. Alors $\hat{M} = M \oplus M^{[2]} \oplus ... \oplus M^{[n]}$.

Exemple

Reprenons le graphe précédent dont la matrice d'adjacence est $M = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$.

Le graphe possède $n = 3$ sommets donc d'après la propriété précédente, la matrice d'adjacence de la fermeture transitive est $\hat{M} = M \oplus M^{[2]} \oplus M^{[3]}$.

Les calculs donnent les résultats suivants :

$$M^2 = M^{[2]} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

et $M^3 = M^{[3]} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$

$$\begin{aligned} \hat{M} &= M \oplus M^{[2]} \oplus M^{[3]} \\ &= \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \oplus \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \oplus \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} \end{aligned}$$

Connaissant maintenant la matrice d'adjacence de la fermeture transitive, il est facile de compléter le graphe.

6.3 Niveau des sommets d'un graphe sans circuit

Les graphes sans circuit ni boucle peuvent être ordonnés par niveaux.

La définition suivante va préciser la notion de niveau d'un sommet d'un graphe orienté défini sur un ensemble S de sommets.

6.3.1 Dessin d'un graphe par niveaux

On appelle **sommet de niveau 0** tout sommet qui n'a pas de prédécesseur. Si on note S_0 l'ensemble des sommets de niveau 0, alors on appelle **sommet de niveau 1** tout sommet qui n'a pas de prédécesseur dans $S - S_0$ (ensemble S privé des éléments de S_0). On définit ensuite les **sommets de niveau 2** et ainsi de suite…

Exemple

Sur l'ensemble $S = \{A;B;C;D;E\}$, on définit un graphe par sa matrice d'adjacence M.

$$M = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Cherchons les sommets de niveau 0.

Tableau 6.3

Sommets	*A*	*B*	*C*	*D*	*E*
Prédécesseurs	*B*	---	*A*, *E*	*B*	*A*, *D*

B est le seul sommet qui n'a pas de prédécesseur, il est donc le seul sommet de niveau 0.

$$S_0 = \{B\}$$

On va maintenant chercher les sommets de niveau 1 : ce sont les sommets qui n'ont pas de prédécesseurs dans $S - S_0 = \{A;C;D;E\}$.

Tableau 6.4

Sommets	*A*	*C*	*D*	*E*
Prédécesseurs	---	*A*, *E*	---	*A*, *D*

A et D n'ont pas de prédécesseurs dans $S - S_0$, ils sont donc de niveau 1.

$$S_1 = \{A,D\}$$

Les sommets de niveau 2 sont ceux qui n'ont pas de prédécesseurs dans $(S - S_0) - S_1 = \{C,E\}$.

E n'a plus de prédécesseur alors que C a le sommet E comme prédécesseur. Donc E est de niveau 2, et enfin C est de niveau 3. Les niveaux des sommets sont résumés dans le tableau 6.5.

Tableau 6.5

Niveaux	0	1	2	3
Sommets	*B*	*A*, *D*	*E*	*C*

On peut faire le graphe en alignant verticalement les sommets de même niveau (fig. 6.10).

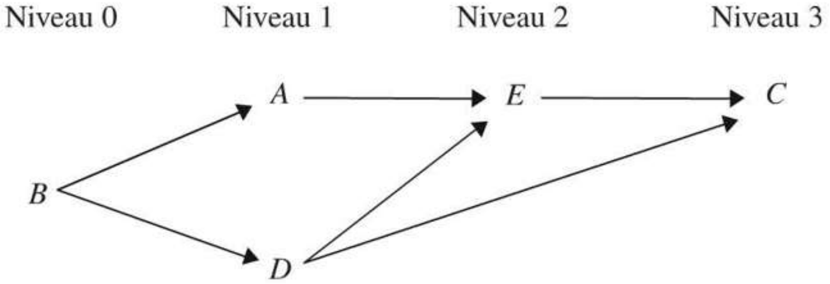


Figure 6.10

6.3.2 Arborescence

Une arborescence est un graphe orienté qui possède un sommet de niveau 0, appelé **racine**, à partir duquel on peut atteindre tout autre sommet par un **chemin unique.**

Exemple

Le graphe de la fig. 6.11 est une arborescence dont la racine est *A*.

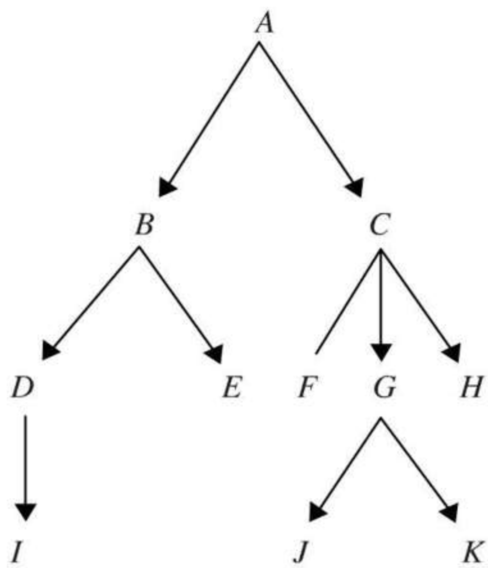


Figure 6.11

6.3.3 Graphe pondéré (ou valué)

Sur certains graphes orientés, il peut être nécessaire d'attribuer une valeur à chacun des arcs : on obtient alors un **graphe pondéré** ou **valué.**

La **valeur d'un chemin** est la somme des valeurs des arcs qui constituent le chemin. Il est alors possible de chercher le chemin de valeur minimale ou maximale reliant un sommet à un autre.

Exemple

Sur le graphe pondéré de la fig. 6.12, pour relier B à C, il y a trois chemins possibles :
- Le chemin (B,A,E,C) de valeur $4 + 3 + 5 = 12$.
- Le chemin (B,D,E,C) de valeur $7 + 4 + 5 = 16$.
- Le chemin (B,D,C) de valeur $7 + 8 = 15$.

Le chemin minimal est (B,A,E,C), le chemin maximal est (B,D,E,C).

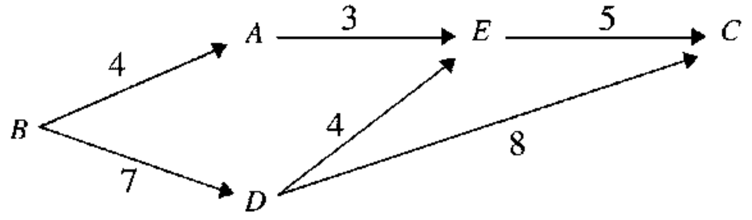


Figure 6.12

Un algorithme de recherche de chemin minimal sera proposé en exercice corrigé 6.16.

6.4 MÉTHODE MPM D'ORDONNANCEMENT D'UN GRAPHE

La réalisation d'un projet passe par l'exécution de différentes tâches, de durées souvent différentes. Si certaines tâches peuvent être réalisées simultanément, d'autres nécessitent que certaines tâches aient été réalisées antérieurement.

Faire l'**ordonnancement d'un projet** consiste à organiser ce projet en respectant les contraintes d'antériorité des tâches tout en minimisant la durée totale de réalisation.

La **méthode MPM** (Méthode des potentiels metra) permet l'ordonnancement de projets, c'est la méthode que nous exposerons dans ce cours. Nous aurions pu choisir la méthode PERT, mais elle est plus complexe à mettre en œuvre.

Exemple

Pour illustrer la méthode MPM, nous allons réaliser l'ordonnancement d'un projet fictif, comprenant des tâches que nous noterons A, B, C, D, E, F, G. Les contraintes d'antériorité et les durées de ces tâches sont consignées dans le tableau 6.6.

Tableau 6.6

Tâche	Durée (en jours)	Tâches antérieures
A	3	aucune
B	2	aucune
C	4	A
D	5	A, B
E	3	C
F	4	C
G	3	E

La première chose à faire est de définir le niveau de chaque tâche. Il est simple de voir que *A* et *B* sont de niveau 0, *C* et *D* de niveau 1, *E* et *F* de niveau 2 et *G* de niveau 3.

Tableau 6.7

Niveaux	0	1	2	3
Sommets	A, B	C, D	E, F	G

Le graphe ordonné par niveaux est celui de la figure 6.13.

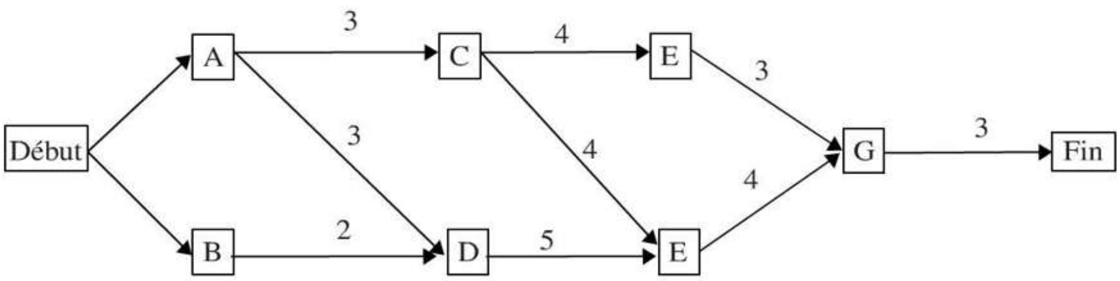


Figure 6.13

On peut remarquer que l'on a rajouté deux tâches fictives : Début et Fin ; et que l'on a pondéré le graphe en ajoutant les durées de chaque tâche.

Il est plutôt simple de calculer la durée minimale du projet (si vous ne la trouvez pas, nous la calculerons un peu plus loin). Mais on peut se poser quelques questions :

- À quel moment peut-on commencer une tâche ?
- Est-ce qu'on peut retarder le moment de démarrer certaines tâches sans que cela ait d'impact sur la durée minimale du projet ?

Pour répondre à ces questions, nous allons définir, pour chaque tâche, les notions de **date au plus tôt** et de **date au plus tard**.

Ces notions utilisant les durées des tâches, nous noterons $d(x_j)$ la durée d'une tâche x_j.

6.4.1 Date au plus tôt – Date au plus tard

La **date au plus tôt** d'une tâche est la date minimale à laquelle on peut commencer la tâche, car toutes les tâches antérieures sont terminées.

Nous noterons $t(x_j)$ la date au plus tôt d'une tâche x_j. $t(x_j)$ est le plus grand des nombres $t(x_i) + d(x_i)$ où x_i est une des tâches qui précèdent immédiatement la tâche x_j.

Exemple

Calculons les dates au plus tôt des tâches du projet :
- $t(\text{A}) = t(\text{B}) = 0$
- $t(\text{C}) = t(\text{A}) + d(\text{A}) = 0 + 3 = 3$
- D a deux prédécesseurs : A et B. On calcule $t(\text{A}) + d(\text{A}) = 0 + 3 = 3$ et $t(\text{B}) + d(\text{B}) = 0 + 2 = 2$. $t(\text{D})$ est le plus grand de ces deux nombres donc $t(\text{D}) = 3$.
- $t(\text{E}) = t(\text{C}) + d(\text{C}) = 3 + 4 = 7$
- F a deux prédécesseurs : C et D. On calcule $t(\text{C}) + d(\text{C}) = 7$ et $t(\text{D}) + d(\text{D}) = 3 + 5 = 8$. Le plus grand de ces deux nombres est 8 donc $t(\text{F}) = 8$.
- Pour G, on calcule $t(\text{E}) + d(\text{E}) = 7 + 3 = 10$ et $t(\text{F}) + d(\text{F}) = 8 + 4 = 12$ donc $t(\text{G}) = 12$.
- Et pour finir, $t(\text{Fin}) = t(\text{G}) + d(\text{G}) = 12 + 3 = 15$.

On a ainsi calculé la date au plus tôt de la fin du projet qui est de 15 jours.

La **date au plus tard** d'une tâche est la date maximale à laquelle on peut commencer la tâche sans que cela ne repousse la date de fin du projet.

Nous noterons $T(x_j)$ la date au plus tard d'une tâche x_j. $T(x_j)$ est le plus petit des nombres $T(x_k) - d(x_j)$ où x_k est une des tâches qui suit immédiatement la tâche x_j.

Exemple (suite)

Calculons les dates au plus tard des tâches du projet : on doit commencer par la fin puisqu'à chaque fois, on doit considérer les successeurs.
- On a bien sûr $T(\text{Fin}) = t(\text{Fin}) = 15$
- $T(\text{G}) = T(\text{Fin}) - d(\text{G}) = 15 - 3 = 12$
- $T(\text{F}) = T(\text{G}) - d(\text{F}) = 12 - 4 = 8$
- $T(\text{E}) = T(\text{G}) - d(\text{E}) = 12 - 3 = 9$
- $T(\text{D}) = T(\text{F}) - d(\text{D}) = 8 - 5 = 3$
- C a deux successeurs : E et F. On calcule $T(\text{E}) - d(\text{C}) = 9 - 4 = 5$ et $T(\text{F}) - d(\text{C}) = 8 - 4 = 4$. Le plus petit de ces deux nombres est 4, donc $T(\text{C}) = 4$.
- $T(\text{B}) = T(\text{D}) - d(\text{B}) = 3 - 2 = 1$
- A a deux successeurs : C et D. On calcule $T(\text{C}) - d(\text{A}) = 4 - 3 = 1$ et $T(\text{D}) - d(\text{A}) = 3 - 3 = 0$. Donc $T(\text{A}) = 0$.

Pour chaque tâche, on va rajouter les dates au plus tôt et au plus tard en présentant chaque tâche par un rectangle :

Tableau 6.8

$t(x_j)$	$T(x_j)$
x_j	

Par exemple, pour la tâche E :

7	9
E	

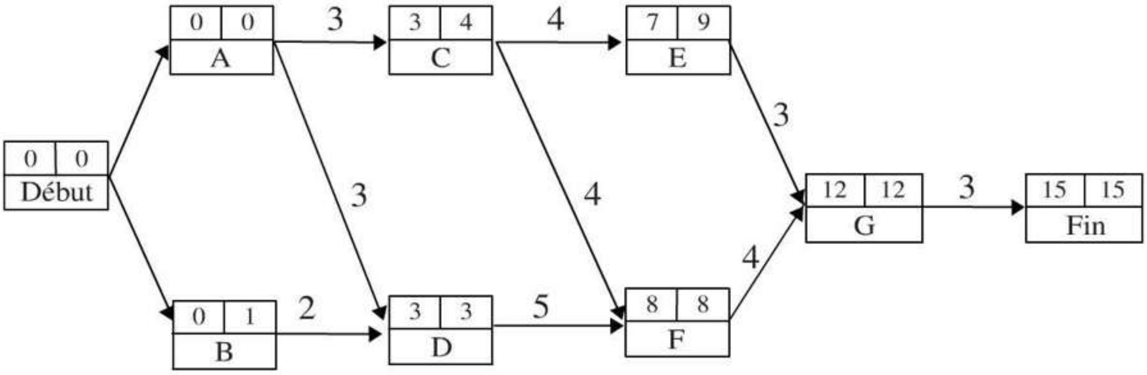


Figure 6.14

Examinons la tâche C : on a $t(C) = 3$ et $T(C) = 4$. Au plus tôt, on peut commencer à 3 jours et au plus tard à 4 jours. Pour cette tâche, on dispose donc d'une « liberté » d'un jour. Ce n'est pas ainsi pour d'autres tâches (comme D par exemple) pour lesquelles les dates au plus tôt et au plus tard sont les mêmes. Ces tâches sont qualifiées de **tâches critiques**. Et l'enchaînement A-D-F-G-Fin, qui ne contient que des tâches critiques, est qualifié de **chemin critique**.

Une **tâche critique** est une tâche dont les dates au plus tôt et au plus tard sont égales. Donc x_j est critique si et seulement si $t(x_j) = T(x_j)$.

Un **chemin critique** est un chemin reliant le début à la fin et qui n'est constitué que de tâches critiques.

6.4.2 Marges d'une tâche

La **marge totale** d'une tâche c'est le retard maximum que l'on peut accepter sur la date de début de la tâche sans que cela ne retarde la date de fin du projet.

La marge totale d'une tâche x_j se note $MT(x_j)$ et elle vaut $MT(x_j) = T(x_j) - t(x_j)$.

Exemple (suite)

La tâche C n'est pas critique. On peut constater qu'on peut retarder le début de C d'un jour sans retarder la date de fin du projet (et c'est le maximum).

En effet, si C commence à la date 4, cela n'empêcherait pas F de commencer à la date 8, E pourrait commencer à la date 8 et n'empêcherait pas G de commencer à la date 12.

Ce retard maximum s'appelle la **marge totale** de la tâche.

La marge totale d'une tâche critique est nulle.

La **marge libre** d'une tâche c'est le retard maximum que l'on peut accepter sur la date de début de la tâche sans que cela ne retarde la date de début au plus tôt de chacune des tâches qui suit immédiatement.

La marge libre d'une tâche x_j se note $ML(x_j)$. $ML(x_j)$ est le plus petit des nombres $t(x_k) - t(x_j) - d(x_j)$, où x_k est une tâche qui suit x_j.

Exemple (suite)

Dans le graphe du projet, on a $MT(B) = 1$; $MT(C) = 1$; $MT(E) = 2$.
Retarder la date de début d'une tâche peut avoir des conséquences sur la date de fin d'un projet. Mais si on s'intéresse simplement aux tâches qui suivent immédiatement, on peut regarder l'impact d'un retard sur les dates au plus tôt de ces tâches.
Si on examine la tâche C, un retard d'un jour ne retarde la date au plus tôt de F, mais a un impact sur la date au plus tôt de E. Donc, si on ne veut pas retarder les dates au plus tôt des successeurs immédiats de C, on ne peut pas retarder C. Il s'agit ici de la **marge libre** de C (qui est donc nulle).
Sur le projet, B n'a qu'un successeur : la tâche D. Donc la marge libre de B est $ML(B) = t(D) - t(B) - d(B) = 3 - 0 - 2 = 1$
C a deux successeurs : E et F.
On calcule $t(E) - t(C) - d(C) = 7 - 3 - 4 = 0$ et $t(F) - t(C) - d(C) = 8 - 3 - 4 = 1$.
Le plus petit de ces deux nombres est 0 donc la marge libre de C est $ML(C) = 0$.

Une autre marge que l'on peut définir est la **marge certaine**. C'est le retard maximum que l'on peut accepter sur la date de début de la tâche, **quand celle-ci commence à sa date au plus tard**, sans que cela ne retarde la date de début au plus tôt de chacune des tâches qui suit immédiatement.

La marge certaine d'une tâche x_j est notée $MC(x_j)$. $MC(x_j)$ est le plus petit des nombres $t(x_k) - T(x_j) - d(x_j)$, où x_k est une tâche qui suit x_j sauf si ce nombre est négatif, auquel cas $MC(x_j) = 0$.

Exemple (suite)

Calculons par exemple, la marge certaine de C. Le principe des calculs est le même que pour le calcul de la marge libre, sauf qu'on remplace la date au plus tôt de C par la date au plus tard.
C a deux successeurs : E et F.
On calcule $t(E) - T(C) - d(C) = 7 - 4 - 4 = -1$ et $t(F) - T(C) - d(C) = 8 - 4 - 4 = 0$.
Le plus petit des deux nombres est -1, mais puisqu'il est négatif, la marge certaine de C est $MC(C) = 0$.
Les valeurs des différentes marges de chaque tâche du projet sont consignées dans le tableau 6.9.

Tableau 6.9

Tâche	A	B	C	D	E	F	G
Marge totale	0	1	1	0	2	0	0
Marge libre	0	1	0	0	2	0	0
Marge certaine	0	0	0	0	0	0	0

TD – Déplacements dans un jeu vidéo

Un développeur de jeu vidéo doit travailler sur les déplacements possibles d'un joueur à l'intérieur d'un donjon, composé de six salles notées A, B, C, D, E et F. L'entrée et la sortie du donjon ne peuvent se faire que par la salle F.

À l'intérieur du donjon, les déplacements autorisés sont schématisés par des flèches sur le plan de la fig. 6.15. Par exemple, la flèche entre la salle A et la salle B montre qu'on peut aller de B vers A, mais pas de A vers B.

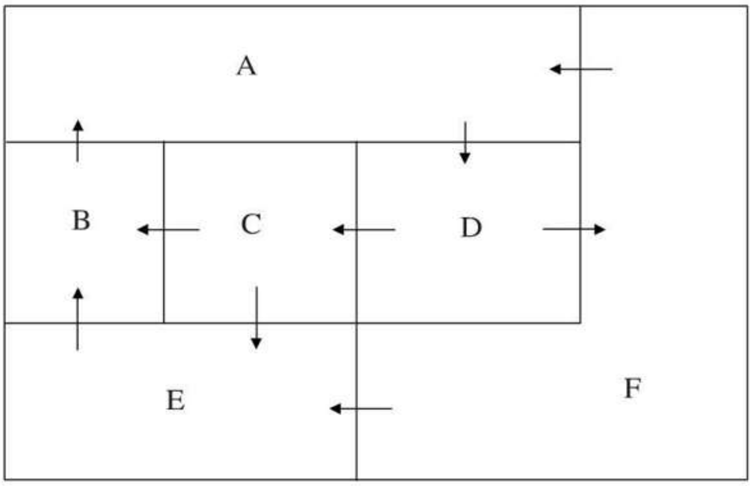


Figure 6.15

Partie A

a) Dessiner un graphe modélisant les salles et les déplacements.

b) Faire un tableau donnant les successeurs et les prédécesseurs de chaque sommet.

c) Le graphe peut-il être ordonné par niveaux ?

d) Écrire la matrice d'adjacence du graphe.

Partie B

a) Calculer M^3. Expliquer la signification des nombres de la dernière ligne de cette matrice.

b) Le joueur se trouvant dans la salle F, déterminer les salles dans lesquelles il peut se retrouver en quatre déplacements. Écrire tous les déplacements correspondants.

c) Est-il possible, à partir de F et en cinq déplacements, de visiter chaque salle ?

d) Le joueur pourra-t-il de partir de F et y revenir en exactement six déplacements ? Si oui, ces trajets permettent-ils de visiter chaque salle ?

Partie C

Chaque entrée dans une salle coûte au joueur un certain nombre de points. Les entrées dans les salles A, B, C, D, E et F coûtent respectivement 25, 20, 40, 30, 25 et 20 points. Le développeur souhaite que le joueur ne puisse pas faire plus de huit déplacements à l'intérieur du donjon, à partir de son entrée en F et son retour en F.

a) Dessiner le graphe valué modélisant la consommation en points de chaque déplacement.

b) Combien de trajets de huit déplacements le joueur peut-il faire à l'intérieur du donjon ?

c) En effectuant ces huit déplacements, combien de points le joueur consommera-t-il au maximum ?

Solution du TD

Partie A

a) Voir le graphe de la fig. 6.16.

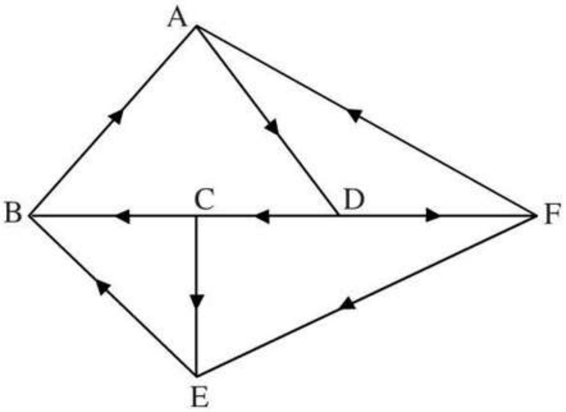


Figure 6.16

b)

Tableau 6.10

Sommets	A	B	C	D	E	F
Successeurs	D	A	B,E	C,F	B	A,E
Prédécesseurs	B,F	C,E	D	A	C,F	D

c) Chaque sommet a au moins un prédécesseur, donc on ne peut pas ordonner le graphe par niveaux.

d)

$$M = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Partie B

a)

$$M^3 = \begin{pmatrix} 1 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

La dernière ligne donne le nombre de chemins de longueur 3 qui relient F à chacun des autres sommets. Il y a donc :

- Un seul chemin qui va de F à A, à C ou à F.
- Aucun chemin de F à B, à C ou à E.

b) La dernière ligne de la matrice M^4 est : 1 1 0 1 2 0. Cette ligne donne le nombre de chemins de longueur 4 reliant F aux autres sommets.

En quatre déplacements, on peut se retrouver en A, en B, en D ou en E.

Pour aller de F en A, le trajet est FADFA.

De F à B, le trajet est FADCB.

De F à D, on emprunte FEBAD. De F à E, les deux possibilités sont FADCE et FADFE.

c) La dernière ligne de la matrice M^5 est : 1 2 1 1 0 1.

Le seul chemin de longueur 5 entre F et A est FADCBA, il ne passe pas par E.

Deux chemins de longueur 5 existent entre F et B : FADFEB et FADCEB. Ce deuxième chemin passe par toutes les salles : c'est un chemin hamiltonien.

Le seul chemin de longueur 5 qui va de F à C est FEBADC et il passe par toutes les salles.

FADFAD est le seul chemin reliant F à D, il ne passe pas par B, C et E.

Enfin, FEBADF relie F à lui-même sans passer par C.

Finalement, il y a deux chemins de longueur 5 qui visitent toutes les salles.

d) Le calcul de M^6 montre qu'il n'y a qu'un seul circuit de longueur 6 qui relie F à lui-même. Ce circuit est FADFADF, il ne passe pas par toutes les salles.

Partie C

a)

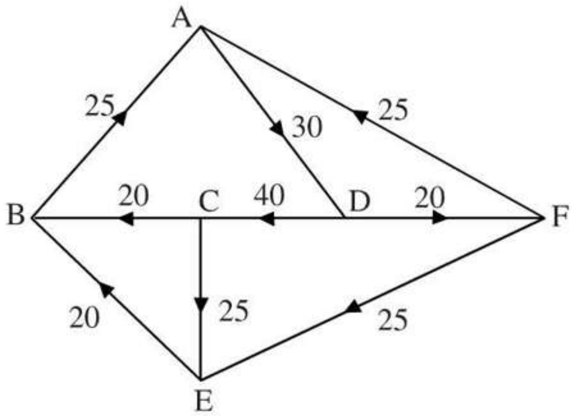


Figure 6.17

b) Le calcul de M^8 permet de dire qu'il y a trois circuits de longueur 8 de F à F. Ces trois circuits sont :

- FADCEBADF qui consomme $25 + 30 + 40 + 25 + 20 + 25 + 30 + 20 = 215$ points.
- FEBADFADF et FADFEBADF qui consomment 195 points.

Exercices corrigés

6.1 Pour chacun des graphes (fig. 6.18), faire le tableau des successeurs et des prédécesseurs.

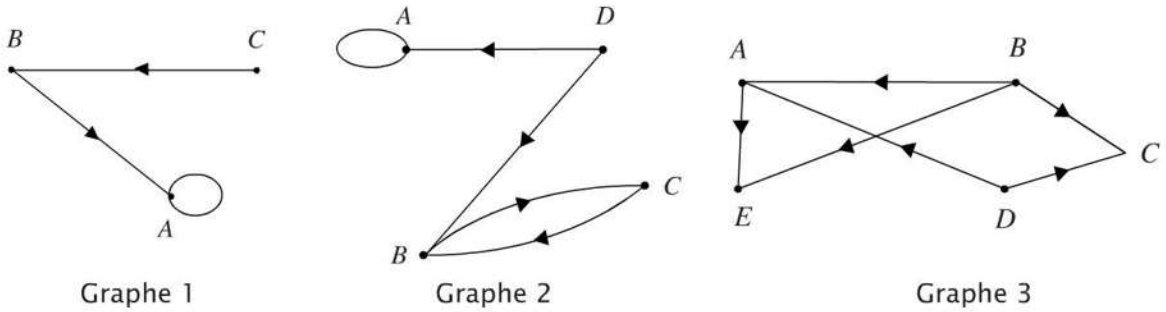


Figure 6.18

6.2 Pour chacun des graphes de l'exercice 6.1, écrire la matrice d'adjacence (les sommets seront pris dans l'ordre alphabétique).

6.3 Compléter chaque tableau en ajoutant les prédécesseurs, puis construire le graphe associé.

Tableau 6.11

Sommet	*A*	*B*	*C*	*D*
Successeurs	*B, C*	*B*	*B, D*	*C*
Prédécesseurs				

Tableau 6.12

Sommet	*A*	*B*	*C*	*D*
Successeurs	*C*	*A, C*	*D*	*B*
Prédécesseurs				

6.4 Écrire la matrice d'adjacence de chacun des graphes de l'exercice 6.3.

6.5 Compléter chaque tableau en ajoutant les successeurs.

Tableau 6.13

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs					
Prédécesseurs	*B, E*	*A*	*B,C,D,E*	*A*	*B*

Tableau 6.14

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs					
Prédécesseurs	*B*	*C, D, E*	*A, E*	*C*	*D*

6.6 M_1 et M_2 sont les matrices d'adjacence de deux graphes définis respectivement sur les ensembles $S_1 = \{A; B; C; D\}$ et $S_2 = \{A; B; C; D; E\}$. Faire le tableau des successeurs et des prédécesseurs de chaque graphe.

$$M_1 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \qquad M_2 = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

6.7 a) Écrire la matrice d'adjacence M du graphe de la fig. 6.19.

b) Calculer M^2 Expliquer la signification des quatre nombres de la deuxième ligne de la matrice M^2 ?

c) Combien y a-t-il de chemins de longueur 2 dans le graphe ? Les citer.

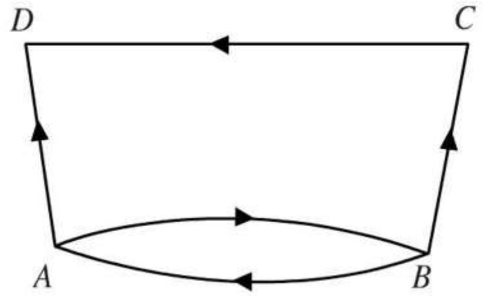


Figure 6.19

6.8 a) Écrire la matrice d'adjacence M du graphe de la fig. 6.20.

b) Calculer M^3 Expliquer pourquoi on peut affirmer qu'il existe deux chemins de longueur 3 reliant B à A. Citer ces deux chemins.

c) Le graphe possède-t-il des circuits de longueur 3 ?

d) Calculer M^4. Entre quels sommets n'existe-t-il pas de chemin de longueur 4 ?

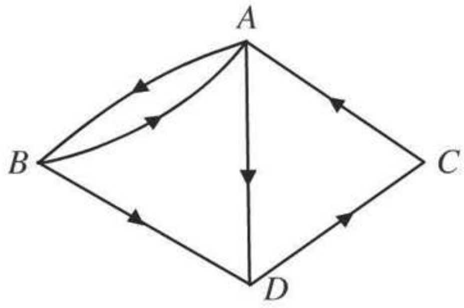


Figure 6.20

6.9 a) Écrire la matrice d'adjacence M du graphe de la fig. 6.21.

b) Calculer M^4.

c) Combien y a-t-il de chemins de longueur 4 qui partent de A ? Citer ces chemins. Parmi eux, y a-t-il des chemins hamiltoniens ?

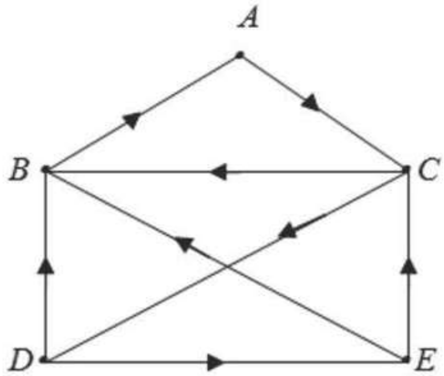


Figure 6.21

6.10 a) Écrire la matrice d'adjacence M du graphe de la fig. 6.22.

b) Quels arcs doit-on rajouter pour faire la fermeture transitive du graphe ?

c) Calculer les matrices booléennes $M^{[2]}$ et $M^{[3]}$. En déduire la matrice $\hat{M}$ de la fermeture transitive.

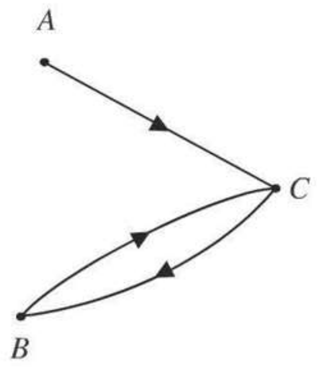


Figure 6.22

6.11 a) Écrire la matrice d'adjacence M du graphe de la fig. 6.23.

b) Quels arcs doit-on rajouter pour faire la fermeture transitive du graphe ?

c) Calculer la matrice $\hat{M}$ de la fermeture transitive.

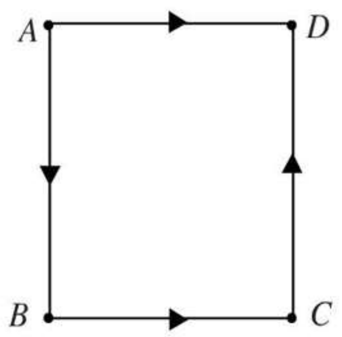


Figure 6.23

6.12 Calculer la matrice $\hat{M}$ de la fermeture transitive du graphe de la fig. 6.24. Quels arcs doit-on ajouter pour la réaliser ?

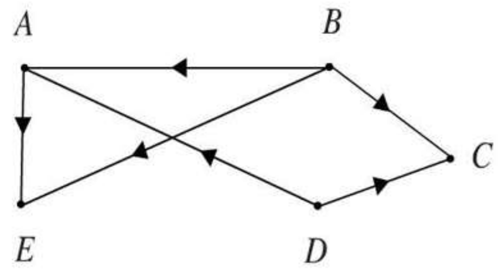


Figure 6.24

6.13 Un graphe est défini par le tableau 6.15. Déterminer le niveau de chaque sommet et représenter le graphe en tenant compte des niveaux.

Tableau 6.15

Sommets	A	B	C	D	E	F	G
Successeurs	C, E	---	F, G	---	B, F	D	D

6.14 Un graphe est défini par le tableau 6.16. Déterminer le niveau de chaque sommet et représenter le graphe en tenant compte des niveaux.

Tableau 6.16

Sommets	A	B	C	D	E	F	G	H
Successeurs	E, G	---	A, H	F	---	G	---	B, D

6.15 Un graphe est défini par le tableau 6.17. Déterminer les prédécesseurs de chaque sommet. Pourquoi peut-on affirmer que le graphe est une arborescence ? Dessiner cette arborescence.

Tableau 6.17

Sommets	A	B	C	D	E	F	G	H	I	J	K	L
Successeurs	B,L	D,K	---	---	A,J	C,G,I	---	---	---	F	---	H

6.16 Sept villes notées *A, B, C, D, E, F* et *G* constituent un réseau. Sur le graphe valué suivant (fig. 6.25) figurent les liaisons et leurs durées en heures.

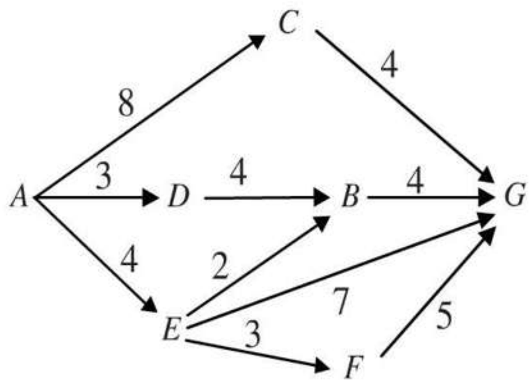


Figure 6.25

a) On va rechercher le chemin minimal pour aller de *A* à *G*. Pour cela, on va se servir de la propriété « *tout chemin optimal est composé de chemins eux-mêmes optimaux* ». À chaque sommet, on va attribuer une marque en appliquant l'algorithme suivant :

- Les sommets de niveau 0 reçoivent la marque 0.
- Les sommets de niveau 1 reçoivent une marque égale à la valeur minimale des chemins venant du niveau 0.
- Les sommets des autres niveaux reçoivent une marque égale à la valeur minimale de tous les chemins provenant des niveaux précédents.

Appliquer cet algorithme sur le graphe et en déduire la durée minimale du trajet pour aller de *A* à *G*.

b) On donne maintenant les distances entre les villes, en kilomètres : $AC = 500$, $AD = 200$, $AE = 210$, $BG = 250$, $CG = 200$, $DB = 250$, $EB = 180$, $EF = 200$, $EG = 450$, $FG = 210$. Appliquer l'algorithme sur le graphe et en déduire la longueur minimale du trajet pour aller de A à G.

6.17 Sur le graphe de la fig. 6.26, déterminer le chemin de longueur minimale reliant *A* à *I*, puis le chemin de longueur maximale reliant *A* à *I*.

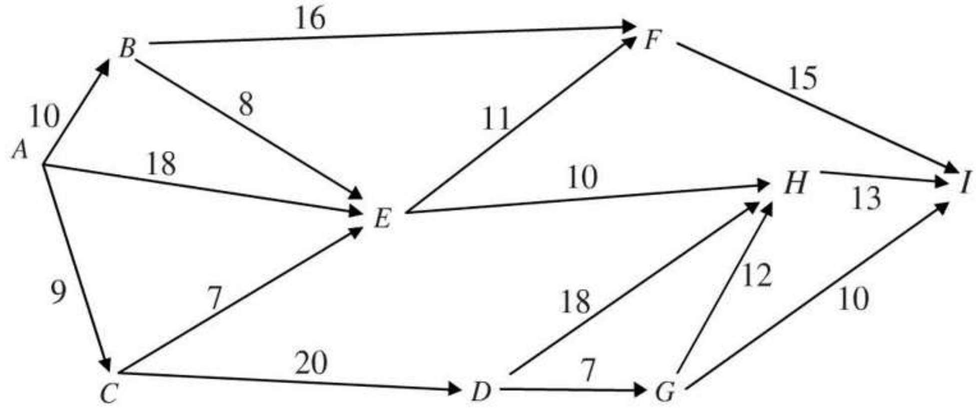


Figure 6.26

6.18 La mise en service d'un nouvel équipement routier demande la réalisation d'un certain nombre de tâches. Le tableau 6.18 représente ces différentes tâches avec leurs relations d'antériorité.

Tableau 6.18

Tâches	*A*	*B*	*C*	*D*	*E*	*F*	*G*
Durées (en jours)	6	3	6	2	4	3	1
Tâches antérieures	aucune	aucune	aucune	*B*	*B*	*A*, *D*	*C*, *E*, *F*

a) Déterminer le niveau de chacune des tâches.

b) Construire le graphe d'ordonnancement du projet et calculer les dates au plus tôt et au plus tard de chaque tâche.

c) Déterminer le chemin critique. Quelle est la durée minimale de réalisation du projet ?

d) Calculer la marge totale de la tâche *E*. Quelle est sa signification ?

e) Calculer la marge libre de la tâche *C*. Quelle est sa signification ?

6.19 La réalisation d'un projet nécessite plusieurs tâches successives dont les durées en jours sont données dans le tableau suivant, ainsi que les tâches devant être réalisées antérieurement.

Tableau 6.19

Tâches	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
Durées	4	2	2	1	2	5	3	3	3	4
Tâches antérieures	aucune	aucune	*A*	*A*	*A*, *B*	*C*	*D*, *E*	*E*, *G*	*H*	*F*,*I*

a) Déterminer le niveau de chacune des tâches.

b) Construire le graphe d'ordonnancement du projet et calculer les dates au plus tôt et au plus tard de chaque tâche.

c) Déterminer le chemin critique. Quelle est la durée minimale de réalisation du projet ?

d) En réalité, la tâche C a nécessité une durée de 5 jours. Est-ce que cela a eu une incidence sur la durée de réalisation du projet ?

6.20 Calculer la marge totale et la marge libre de chacune des tâches de l'exercice 6.19.

Solutions

6.1

Tableau 6.20

Sommet	*A*	*B*	*C*
Successeurs	*A*	*A*	*B*
Prédécesseurs	*A, B*	*C*	-----

Tableau 6.21

Sommet	*A*	*B*	*C*	*D*
Successeurs	*A*	*C*	*B*	A, B
Prédécesseurs	*A, D*	*C, D*	*B*	---

Tableau 6.22

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs	*E*	*A, C, E*	---	*A, C*	---
Prédécesseurs	*B, D*	----	*B, D*	----	*A, B*

6.2

$$M_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \qquad M_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \qquad M_3 = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

6.3

Tableau 6.23

Sommet	*A*	*B*	*C*	*D*
Successeurs	*B, C*	*B*	*B, D*	*C*
Prédécesseurs	---	*A,B,C*	*A, D*	*C*

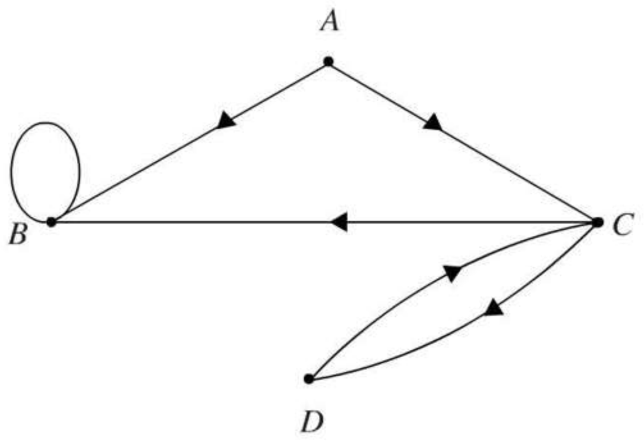


Figure 6.27

Tableau 6.24

Sommet	*A*	*B*	*C*	*D*
Successeurs	*C*	*A, C*	*D*	*B*
Prédécesseurs	*B*	*D*	*A, B*	*C*

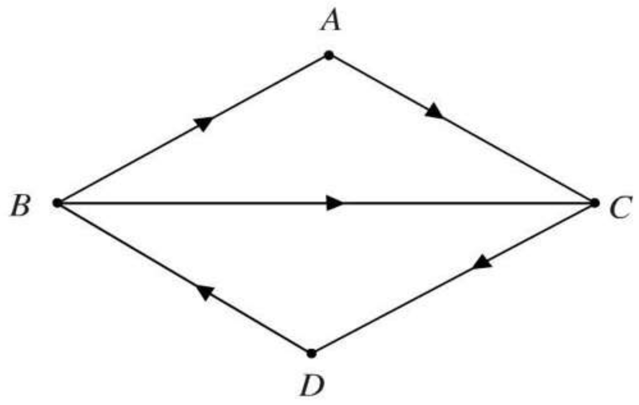


Figure 6.28

6.4 $M_1 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$ $\qquad M_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$

6.5

Tableau 6.25

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs	*B, D*	*A, C, E*	*C*	*C*	*A, C*
Prédécesseurs	*B, E*	*A*	*B,C,D,E*	*A*	*B*

Tableau 6.26

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs	*C*	*A*	*B, D*	*B, E*	*B, C*
Prédécesseurs	*B*	*C, D, E*	*A, E*	*C*	*D*

6.6

Tableau 6.27

Sommet	*A*	*B*	*C*	*D*
Successeurs	*D*	*C, D*	*A, B*	*B,C,D*
Prédécesseurs	*C*	*C, D*	*B, D*	*A,B,D*

Tableau 6.28

Sommet	A	B	C	D	E
Successeurs	C, D	A, E	B, D	B, E	A, C
Prédécesseurs	B, E	C, D	A, E	A, C	B, D

6.7 a) $M = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$

b) $M^2 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$

On sait que les coefficients de M^2 représentent les nombres de chemins de longueur 2 qui relient un sommet à un autre. Donc il n'y a pas de tels chemins reliant B à A, ou à C. Un chemin relie B à lui-même et deux chemins relient B à D.

c) Dans la matrice, la somme des coefficients non nuls est 5 : il y a donc cinq chemins de longueur 2 dans le graphe. Ces cinq chemins sont : (A,B,A), (A,B,C), (B,A,B), (B,A,D) et (B,C,D).

6.8 a) $M = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$ $\quad M^3 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$ $\quad M^4 = \begin{pmatrix} 2 & 1 & 1 & 2 \\ 1 & 2 & 1 & 2 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}$

b) Les coefficients de la matrice M^3 représentent le nombre de chemins de longueur 3 reliant chaque sommet aux autres. À l'intersection de la deuxième ligne et de la première colonne, se trouve le nombre 2 qui est le nombre de chemins de longueur 3 reliant B à A. Ces deux chemins sont (B,A,B,A) et (B,D,C,A).

c) Un circuit étant un chemin dont le sommet de départ coïncide avec le sommet d'arrivée, la matrice M^3 montre qu'il y a trois circuits : au départ de A, de C et de D.

d) Il n'y a aucun chemin de longueur 4 entre D et B.

6.9 a) $M = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$

b) $M^4 = \begin{pmatrix} 1 & 1 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 2 & 1 & 2 & 0 \\ 1 & 2 & 1 & 1 & 1 \\ 1 & 2 & 2 & 1 & 0 \end{pmatrix}$

c) Il y a $1 + 1 + 2 + 0 + 0 = 4$ chemins de longueur 4 qui partent de A.

Un chemin relie A à A : (A,C,D,B,A).

Un chemin relie A à B : (A,C,D,E,B), il est hamiltonien.

Deux chemins relient A à C : (A,C,B,A,C) et (A,C,D,E,C).

6.10 a) $M = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$

b) Il faut ajouter les arcs (B,B), (C,C) et (A,B).

c) $M^{[2]} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ $\quad M^{[3]} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$ $\quad \hat{M} = M + M^{[2]} + M^{[3]} = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$

6.11 a) $M = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$

b) Il faut ajouter les arcs (A,C) et (B,D).

c) $\hat{M} = M + M^{[2]} + M^{[3]} + M^{[4]} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$

6.12 $M = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$ $\qquad \widehat{M} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$

Le seul arc à ajouter est (D,E).

6.13 Il faut d'abord chercher les prédécesseurs de chaque sommet :

Tableau 6.29

Sommets	A	B	C	D	E	F	G
Successeurs	C, E	---	F, G	---	B, F	D	D
Prédécesseurs	---	E	A	F, G	A	C, E	C

$$S_0 = \{A\},\quad S_1 = \{C, E\},\quad S_2 = \{B; F; G\},\quad S_3 = \{D\}$$

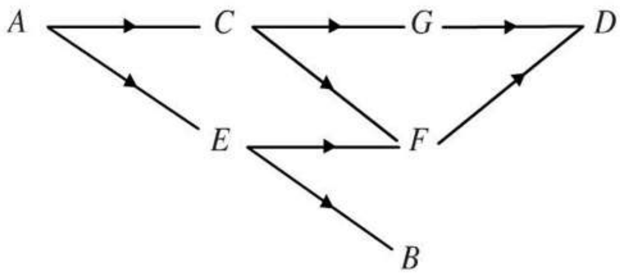


Figure 6.29

6.14

Tableau 6.30

Sommets	A	B	C	D	E	F	G	H
Successeurs	E, G	---	A, H	F	---	G	---	B, D
Prédécesseurs	C	H	---	H	A	D	A, F	C

$$S_0 = \{C\},\quad S_1 = \{A; H\},\quad S_2 = \{B; D; E\},\quad S_3 = \{F\},\quad S_4 = \{G\}$$

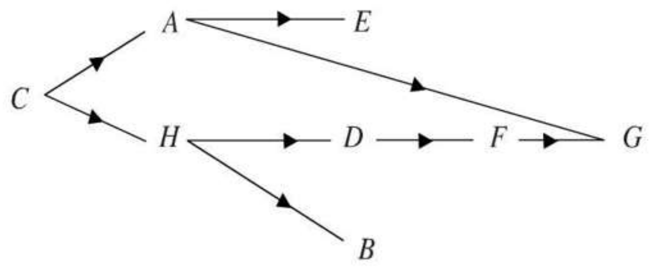


Figure 6.30

6.15 Un sommet n'a pas de prédécesseur et chaque sommet n'a qu'un seul prédécesseur. Ainsi chaque sommet est relié à E par un chemin unique. Le graphe est donc une arborescence dont la racine est E.

Tableau 6.31

Sommets	A	B	C	D	E	F	G	H	I	J	K	L
Successeurs	B,L	D,K	---	---	A,J	C,G,I	---	---	---	F	---	H
Prédécesseurs	E	A	F	B	---	J	F	L	F	E	B	A

$$S_0 = \{E\},\quad S_1 = \{A;\, J\},\quad S_2 = \{B;\, F;\, L\},\quad S_3 = \{C;\, D;\, G;\, H;\, I;\, K\}$$

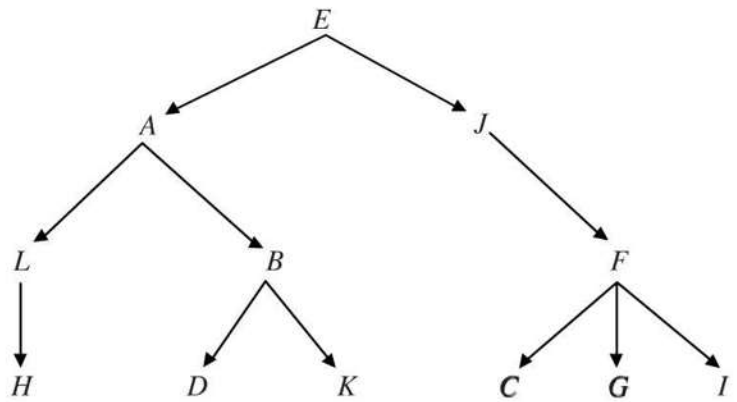


Figure 6.31

6.16 a) A reçoit la marque 0. C, D et E sont de niveau 1, chacun d'eux n'a qu'un prédécesseur, donc on leur attribue respectivement les marques 8, 3 et 4.

Deux chemins arrivent sur B, de durées $3 + 4 = 7$ et $4 + 2 = 6$ donc on attribue la marque 6 au sommet B. On attribue la marque 7 à F.

Quatre chemins arrivent sur G : (marque de C) + 8 = 12 ; (marque de B) + 4 = 10 ; (marque de E) + 7 = 11 ; (marque de F) + 5 = 12. La marque de G est donc 10 et c'est la durée minimale du trajet pour relier A à G.

Le chemin de durée minimale est (A,E,B,G).

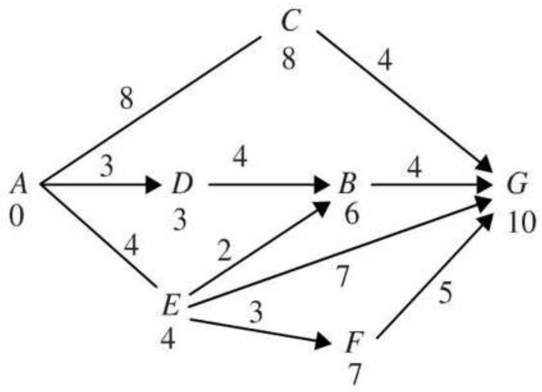


Figure 6.32

b) Le trajet le plus court est (A,E,F,G) de longueur $210 + 200 + 210 = 620$ km.

6.17 Le chemin de longueur minimale est (A,C,E,H,I) de longueur $9 + 7 + 10 + 13 = 39$. Celui de longueur maximale est (A,C,D,G,H,I) de longueur $9 + 20 + 7 + 12 + 13 = 61$.

6.18 a) $S_0 = \{A; B; C\}$, $S_1 = \{D; E\}$, $S_2 = \{F\}$, $S_3 = \{G\}$

b)

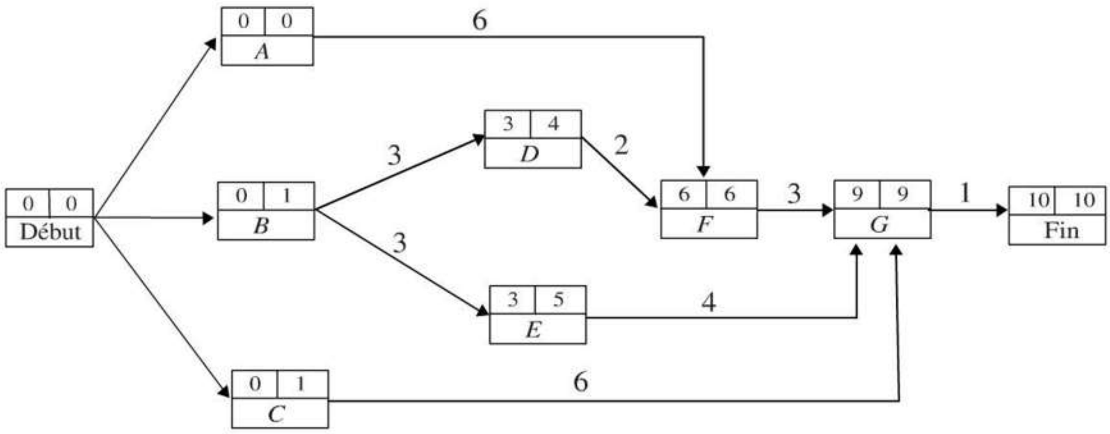


Figure 6.33

c) Le chemin critique est (A,F,G). La durée minimale de réalisation du projet est 10 jours.

d) La marge totale de E est $MT(E) = T(E) - t(E) = 5 - 3 = 2$ (jours). Cela veut dire que l'on peut accepter au maximum deux jours de retard sur le départ de la tâche E sans que cela ne retarde la date de fin du projet.

e) La marge libre de la tâche C est $ML(C) = t(G) - t(C) - d(C) = 9 - 0 - 6 = 3$ (jours). C'est le retard maximum que l'on peut accepter sur la date de début de la tâche C sans que cela ne retarde la date de début au plus tôt de chacune de la tâche G (la seule tâche qui suit immédiatement C).

6.19 a)

Tableau 6.32

Tâches	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
Niveaux	0	0	1	1	1	2	2	3	4	5

b)

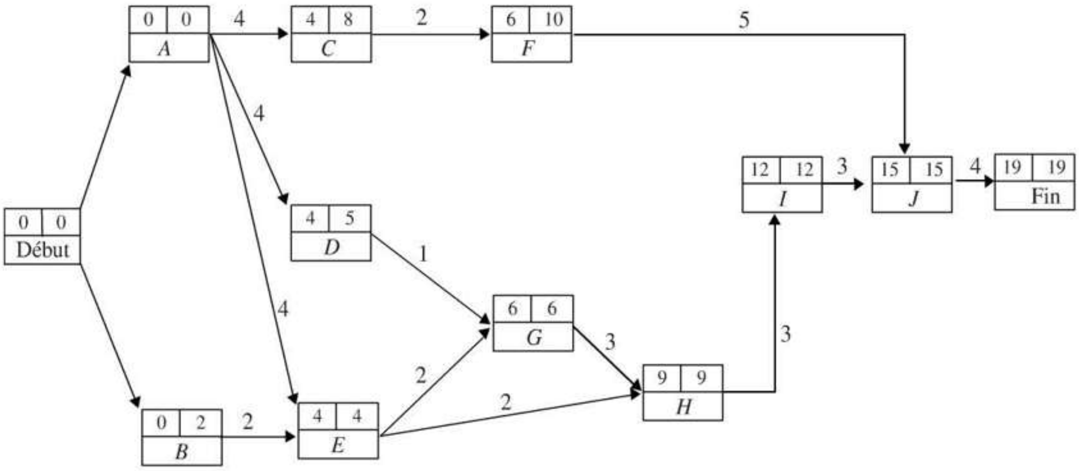


Figure 6.34

c) Le chemin critique est (A,E,G,H,I,J). La durée minimale de réalisation du projet est 19 jours.

d) Avec une durée de 5 jours pour la tâche C, la date au plus tôt de début de la tâche F est $t(F) = 9$. Comme $t(F) \leq T(F)$, cela n'a pas d'impact sur la date de fin du projet.

6.20 On rappelle que :

- la marge totale d'une tâche x_j est $MT(x_j) = T(x_j) - t(x_j)$;
- la marge libre $ML(x_j)$ d'une tâche x_j est le plus petit des nombres $t(x_k) - t(x_j) - d(x_j)$, où x_k est une tâche qui suit x_j.

Tableau 6.33

Tâche	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*	*J*
Marge totale	0	2	4	1	0	4	0	0	0	0
Marge libre	0	2	0	1	0	4	0	0	0	0

L'EXAMEN DE MATHÉMATIQUES

7

L'épreuve de mathématiques s'appuie sur un sujet comportant deux ou trois exercices, couvrant une partie significative de l'ensemble du programme de mathématiques. Celle-ci ne vise pas à évaluer les connaissances en algorithmique appliquée. Les exercices posés présentent des situations concrètes en relation avec des activités professionnelles visées par le diplôme.

La calculatrice est autorisée et son usage est très utile dans divers domaines, notamment pour le calcul matriciel.

L'évaluation de cette épreuve prend, sous forme écrite, l'une ou l'autre des formes suivantes :

- **Forme ponctuelle** : un examen d'une durée de deux heures en fin de formation.
- **Contrôle en cours de formation** (CCF) : plusieurs devoirs surveillés se déroulant pendant le deuxième semestre de la seconde année, au sein de l'établissement de formation.

SUJET MÉTROPOLE 2014

Exercice 1

Un lycée a été doté de postes informatiques et de logiciels. Le proviseur envisage de transformer une salle de cours en salle informatique. Pour cela, le responsable du projet définit les taches à réaliser avec leur durée. Le tableau 7.1 regroupe l'ensemble de ces données.

Le but de cet exercice est d'ordonner la réalisation de ces tâches de façon à ce que la salle soit disponible le plus rapidement possible. On considère le graphe orienté correspondant aux conditions d'antériorité données par le tableau 7.1.

1. Déterminer le niveau de chacun des sommets du graphe.

2. Donner le tableau des successeurs.

3. a) Construire le graphe d'ordonnancement du projet (selon la méthode P.E.R.T. ou M.P.M.). Déterminer pour chaque tâche les dates au plus tôt et au plus tard.
b) En déduire le chemin critique et la durée minimale de réalisation du projet.

Tableau 7.1

Tâche à réaliser	Repère	Durée en jours	Tâches précédentes
Vider la salle de cours et démonter le matériel	A	2	-
Nettoyer et repeindre la salle	B	4	A
Installer les tables et fixer un tableau	C	1	B
Commander et réceptionner le matériel et câblage	D	10	-
Déballer et contrôler le matériel livré	E	1	D
Câbler la salle	F	3	B,E
Installer et brancher les postes informatiques	G	1	C,F
Installer les logiciels, configurer les postes et tester leur fonctionnement	H	7	G

4. En fait, la réalisation de la tâche B a nécessité 10 jours au lieu de 4 car il a fallu enduire un mur et le laisser sécher avant de le peindre. Ce changement a-t-il une incidence sur la durée du projet ? Expliquer pourquoi.

Exercice 2

La loi de Moore, énoncée en 1975 par Gordon Moore, cofondateur de la société Intel, prévoit que le nombre de transistors des microprocesseurs proposés à la vente au grand public double tous les deux ans. Les microprocesseurs fabriqués en 1975 comportaient 9 000 transistors. Pour modéliser cette loi de Moore, on considère la suite (u_n) définie par $u_0 = 9\,000$ et $u_{n+1} = 2u_n$ pour tout entier naturel. Un terme u_n de cette suite correspond au nombre de transistors prévus par la loi de Moore pour un microprocesseur fabriqué lors de l'année $1975 + 2n$.

1. Calculer u_1 et u_2 puis interpréter ces nombres.

2. Quelle est la nature de la suite (u_n) ?

3. Donner l'expression de u_n en fonction de n.

4. Déterminer le nombre de transistors prévus par la loi de Moore pour un microprocesseur fabriqué en 2001.

5. Selon ce modèle, à partir de quelle année les microprocesseurs intégreront-ils plus de 100 milliards de transistors ?

Exercice 3

Partie A

1. a) Décomposer le nombre 2014 en produit de facteurs premiers.
b) En déduire la liste des diviseurs positifs de 2014.

2. Calculer le PGCD des nombres 2014 et 212. On note d ce PGCD. Déterminer l'entier p tel que $2\,014 = p \times d$.

Partie B

Un jury de concours doit établir l'ordre de passage des 2014 candidats qui doivent passer une épreuve orale. Le président du jury envisage la procédure automatique décrite ci-après.

Tout d'abord, il classe les 2 014 candidats par ordre alphabétique et attribue à chacun, en suivant cet ordre, un numéro allant de 1 à 2 014. Ainsi, pour définir un ordre de passage à l'oral des candidats il suffit de dresser la liste des numéros des candidats qui seront appelés l'un après l'autre à passer l'épreuve orale.

Pour établir cette liste, le président du jury choisit un entier n compris entre 1 et 400, puis procède de la manière suivante :

- le premier numéro inscrit sur la liste est le nombre n ;
- le deuxième numéro inscrit sur la liste est le nombre $2n$;
- le troisième numéro inscrit est le nombre $3n$;
- de façon générale, pour obtenir chaque numéro inscrit à partir du deuxième, on ajoute n au numéro précédent et :
  - si la somme s obtenue est inférieure ou égale à 2 014, le numéro inscrit est égal à cette somme s ;
  - sinon, le numéro inscrit est égal à $s - 2\,014$.

Par exemple, en choisissant $n = 257$, les premiers numéros inscrits sur la liste sont, dans l'ordre :

$$257 - 514 - 771 - 1\,028 - 1\,285 - 1\,542 - 1\,799 - 42 - 299 - 556 - \ldots$$

En effet : le premier numéro inscrit est $n = 257$; du $2^{\text{ème}}$ numéro (égal à 514) au $7^{\text{ème}}$ numéro (égal à 1 799), on a ajouté 257 au numéro précédent puisque la somme ne dépassait par 2 014 ; le $8^{\text{ème}}$ numéro inscrit est le numéro 42 car $1\,799 + 257 = 2\,056$ et, comme 2 056 dépasse 2 014, le numéro à inscrire est $2\,056 - 2\,014 = 42$.

Ainsi le candidat 257 passera en premier l'oral ; il sera suivi du candidat 514 et ainsi de suite. Le président du jury se demande si cette procédure permet de convoquer tous les candidats, c'est-à-dire si la liste obtenue, en 2 014 étapes, contient tous les nombres de 1 à 2 014.

1. Dans cette question, le président du jury choisit $n = 212$. Les neuf premiers numéros inscrits sont donc :

$$212 - 424 - 636 - 848 - 1060 - 1272 - 1484 - 1696 - 1908$$

a) Donner la liste des 15 numéros suivants. La valeur de $n = 212$ permet-elle de convoquer tous les candidats ?

b) Avec cette valeur de n, combien de numéros différents la liste comporte-t-elle ?

2. Dans cette question, le président du jury choisit $n = 38$. Déterminer combien de numéros différents comporte la liste. Justifier la réponse. On pourra remarquer que 38 est un diviseur de 2 014.

Partie C

D'après la partie B, il apparaît que, pour certaines valeurs de n, la procédure utilisée ne permet pas de convoquer tous les candidats, c'est-à-dire de constituer une liste comportant tous les nombres de 1 à 2014. On admet le résultat suivant : « Le nombre n choisi permet de former une liste complète comportant tous les numéros de 1 à 2014 dans le cas où le PGCD de 2014 et de n est égal à 1, et dans ce cas seulement ». Ainsi, les nombres permettant de convoquer tous les candidats sont les entiers n compris entre 1 et 400 qui sont premiers avec 2014.

1. Si $n = 15$, la procédure utilisée permet-elle de convoquer tous les candidats ?
2. Dans cette question, on cherche à déterminer le nombre d'entiers n, parmi ceux compris entre 1 et 400, qui permettent par la procédure utilisée de convoquer tous les candidats.

   a) Donner le nombre de multiples de 2 non nuls, inférieurs ou égaux à 400.

   b) Donner la liste des multiples impairs de 19, inférieurs ou égaux à 400.

   c) Donner la liste des multiples impairs de 53, inférieurs ou égaux à 400.

   d) En déduire le nombre d'entiers n qui ne permettront pas de convoquer tous les candidats, puis le nombre d'entiers n qui le permettent.

Partie 2

Algorithmique appliquée

PREMIERS PAS 8

PLAN
8.1 Qu'est-ce qu'un algorithme ?
8.2 Le logiciel Python

OBJECTIFS
- Découvrir la notion d'algorithme
- Comprendre l'intérêt d'un algorithme
- Découverte et installation du logiciel Python

8.1 Qu'est-ce qu'un algorithme ?

Un **algorithme** est une suite finie d'actions élémentaires assemblées de manière logique. Le but d'un algorithme est de réaliser une action complexe en la décomposant en actions simples, c'est-à-dire pouvant être effectuées sans réflexion.

Exemple

Considérons les actions simples suivantes :
Action 1 : Poser $a = 1$.

Action 2 : Effectuer le calcul $\frac{a}{2} + \frac{1}{a}$

Action 3 : Noter *a* le résultat.
Effectuons maintenant ces actions simples dans l'ordre suivant : **Action 1** puis **Action 2** puis **Action 3** puis **Action 2** puis **Action 3**.

Cela donne : $a=1$; $\frac{1}{2}+\frac{1}{1}=1{,}5$; $a=1{,}5$; $\frac{1{,}5}{2}+\frac{1}{1{,}5}=1{,}41666...$; $a=1{,}416666...$

La valeur de *a* obtenue est une approximation de $\sqrt{2}$.
Pour obtenir une meilleure précision, on poursuit en renouvelant autant qu'on le souhaite les Actions 2 et 3, comme par exemple : **Action 1** puis Faire quatre fois de suite **Action 2** puis **Action 3**.

On obtient alors au final $a=1{,}414213562...$, un résultat très proche de $\sqrt{2}$. Ce procédé est un algorithme connu appelé aussi *algorithme de Héron*.

Les premiers algorithmes connus remontent à l'époque babylonienne. Un algorithme peut être exécuté « à la main », c'est-à-dire en effectuant soi-même une à une les actions élémentaires, mais il existe depuis les années 1970 un instrument de calcul permettant de réaliser bien plus rapidement l'ensemble de toutes ces actions : l'ordinateur. En effet, les **langages informatiques** (Python, C, C++, C#, Visual Basic, Scilab…) permettent d'exécuter les algorithmes. Il faut toutefois apprendre à les transcrire. Par exemple en Python : « print » pour « Afficher », « if... : … » pour « Si … alors … »…

Lorsqu'on traduit un algorithme dans un langage de programmation, on dit que l'on **implémente** cet algorithme, la transcription est alors appelée **programme informatique**.

Il existe différentes façons de présenter un algorithme. Notre but étant l'implémentation dans un langage de programmation, nous choisirons d'écrire les algorithmes en **pseudo-code**. Il s'agit de la présentation et de la syntaxe la plus proche des langages de programmation.

Dans ce manuel, tous les programmes sont écrits en langage Python. Nous avons choisi ce langage pour la simplicité de sa syntaxe et sa lisibilité qui permettent un meilleur apprentissage des concepts algorithmiques.

8.2 Le logiciel Python

Python est un logiciel libre qui fonctionne sur la plupart des plates-formes informatiques (Windows, Linux, Mac OS…). Nous utilisons ici la version 3.4, dernière version stable au moment où ce livre est écrit. Celle-ci est téléchargeable gratuitement à l'adresse suivante : *https://www.python.org/downloads/*

Après téléchargement puis installation, on accède à la **console Python** (fig. 8.1) en lançant « IDLE (Python GUI) ».

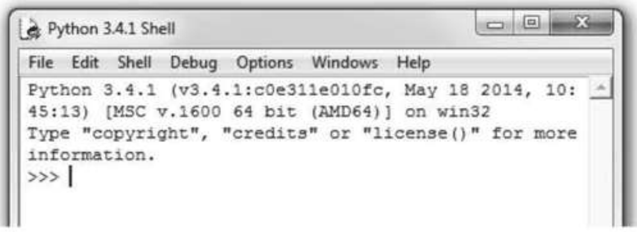


Figure 8.1

Cette console se comporte comme une calculatrice, on peut y saisir directement des instructions qui sont aussitôt exécutées. Lorsque leur nombre devient plus conséquent, on utilise l'**éditeur Python** (fig. 8.2) accessible en cliquant sur « File » puis « New File ».

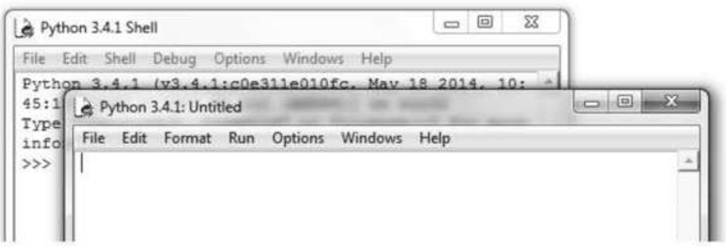


Figure 8.2

CONCEPTS FONDAMENTAUX

9

PLAN

9.1 Données : types et opérations
9.2 Stockage des données
9.3 Lecture et écritures des données
9.4 Instructions conditionnelles
9.5 Instructions itératives
9.6 Fonctions et procédures
9.7 Récursivité

OBJECTIFS

➤ Découvrir les grands principes algorithmiques
➤ Expérimenter les instructions algorithmiques de base
➤ Écrire, interpréter, compléter et corriger des algorithmes, trouver des solutions

9.1 DONNÉES : TYPES ET OPÉRATIONS

En algorithmique, on manipule différents types de données :

- les **booléens** : VRAI, FAUX (notés True et False en Python) ;
- les **entiers** : ..., –3, –2, –1, 0, 1, 2, 3, 4, ... ;
- les **réels** : par exemple 5.37, 15.0 et –14.001 ;
- les **chaînes de caractères** : par exemple 'Bonjour' et '41n5K6' ;
- les **listes** (ou **tableaux**) : par exemple ['az',2,19,VRAI], [2,4,5] et [[5,3],[1,8]].

9.1.1 Type et valeur d'une donnée

Pour connaître le type d'une donnée x, on saisit **`type(x)`** sur la console Python qui retourne respectivement **`'bool'`**, **`'int'`**, **`'float'`**, **`'str'`** ou **`'list'`** si x est de type booléen, entier, réel, chaîne de caractères ou liste.

```
>>> type (7.34)
<class 'float'>
```

Le nombre `7.34` est de type réel.

Activité de découverte 9.1

Utiliser la console Python pour déterminer le type de chacune des données suivantes : 43 ; 43.0 ; '43' ; [43] ; '' ; [] ; [1,2,3] ; False ; [[1],[2,3]].

SOLUTION. 43 est de type entier ; 43.0 est de type réel ; '43' est de type chaîne de caractères ; [43] est de type liste (liste ne contenant qu'un seul élément) ; '' est de type chaîne de caractères (c'est la chaîne de caractère vide) ; [ ] est de type liste (c'est la liste vide) ; [1,2,3] est de type liste ; False est de type booléen ; [[1],[2,3]] est de type liste (liste dont les éléments sont des listes).

On détermine la valeur d'une donnée en la saisissant sur la console Python :

```
>>> 12+5.3
17.3
```

9.1.2 Opération sur les nombres

Activité de découverte 9.2

Sur la console Python effectuer les opérations suivantes :

*2*3 ; 2**3 ; 10**4 ; 25/4 ; 25//4 ; 52%4*

*Que représentent les opérations a*b, a**b, a/b, a//b et a%b en langage Python ?*

SOLUTION. 2*3 vaut 6 ; 2**3 vaut 8 ; 10**4 vaut 10 000 ; 25/4 vaut 6.25 ; 25//4 vaut 6 ; 25%4 vaut 1.
a*b représente $a \times b$; a**b représente a^b ; a/b représente $a \div b$ (division décimale de a par b) ; a//b représente le quotient de la division euclidienne de a par b ; a%b représente le reste de la division euclidienne de a par b.

Tableau 9.1 – **Opérations sur les nombres.**

Opérations	En pseudo-code	En Python
$a + b$	a+b	a + b
$a - b$	a-b	a - b
$a \times b$	a*b	a * b
a^b	a^b	a ** b
Division décimale de a par b	a/b	a / b
Quotient dans la division euclidienne de a par b	a//b	a // b
Reste dans la division euclidienne de a par b	a%b	a %b

9.1.2 Comparer des nombres

Le résultat d'une comparaison est un booléen : VRAI ou FAUX. Par exemple la valeur de « 5 < 8 » est VRAI, celle de « 8 < 5 » est FAUX.

Activité de découverte 9.3

Utiliser la console Python pour déterminer la valeur de chaque booléen :

4==4 ; 3==7 ; 2!=3 ; 6!=6 ; 2<5 ; 2<=5 ; 2>=5

Si a et b sont des nombres, que signifie a==b, a!=b, a<=b et a>=b en Python ?

SOLUTION. 4==4 vaut VRAI, 3==7 vaut FAUX ; 2!=3 vaut VRAI ; 6!=6 vaut FAUX ; 2<5 vaut VRAI ; 2<=5 vaut VRAI ; 2>=5 vaut FAUX.
a==b, a!=b, a<=b et a>=b signifient respectivement « a est bien égal à b », $a \neq b$, $a \leq b$ et $a \geq b$.

Tableau 9.2 – **Opérateurs de comparaison sur les nombres.**

Pseudo-code	Python
a==b (traduction : a **est bien égal** à b)	`a==b`
$a \neq b$	`a!=b`
a<b	`a<b`

Pseudo-code	Python
$a \leq b$	`a<=b`
a>b	`a>b`
$a \geq b$	`a>=b`

9.1.3 Opérations sur les booléens

Activité de découverte 9.4

Utiliser la console Python pour déterminer la valeur de chaque booléen :

not(2==3) ; 2<3 and 3<2 ; 2<3 or 3<2 ; (2<3)<=(1<2) ; (5<3)==(5==7)

Si A et B sont des booléens, que signifient not(A), A and B, A or B, A<=B et A==B en Python ?

SOLUTION. not(2==3) vaut VRAI ; (2<3 and 3<2) vaut FAUX ; (2<3 or 3<2) vaut VRAI ; (2<3)<=(1<2) vaut VRAI ; (5<3)==(5==7) vaut VRAI.
Si A et B sont des booléens, not(A), A and B, A or B, A<=B et A==B signifient non(A), « A et B », « A ou B », $A \Rightarrow B$ et $A \Leftrightarrow B$ respectivement.

Tableau 9.3 – Opérations sur les booléens.

Pseudo-code	Python
non A	not(A)
A et B	A and B
A ou B	A or B

Pseudo-code	Python
A⇒B	A<=B
A⇔B	A==B

9.1.4 Opérations sur les chaînes de caractères

Activité de découverte 9.5

Utiliser la console Python pour déterminer la valeur de chaque chaîne de caractères : '11'+'32' ; 'bonjour'+'112' ; 'bonjour'[0] ; 'bonjour'[1] ; len('bonjour') ; len('11227') ; 'bonjour'[2:4] ; 'bonjour'[0:4] ; 3'bonjour'+'23' ; 4*('11'+'5')+'AA'.*

SOLUTION. '11'+'32' vaut '1132' ; 'bonjour'+'112' vaut 'bonjour112' ; 'bonjour'[0] vaut 'b' ; 'bonjour'[1] vaut 'o' ; len('bonjour') vaut 7 ; len('11227') vaut 5 ; 'bonjour'[2:4] vaut 'nj' ; 'bonjour'[0:4] vaut 'bonj' ; 3*'bonjour'+'23' vaut 'bonjourbonjourbonjour23' ; 4*('11'+'5')+'AA' vaut '115115115115AA'.

Tableau 9.4 – Opérations sur les chaînes de caractères.

Opérations	Pseudo-code	Python
Concaténation des chaînes ch1 et ch2	ch1+ch2	ch1+ch2
Nombre de caractères de la chaîne ch	longueur(ch)	len(ch)
Premier caractère de la chaîne ch	ch[0]	ch[0]
$i+1^{\text{ème}}$ caractère de la chaîne ch	ch[i]	ch[i]
Partie de la chaîne ch, du $i+1^{\text{ème}}$ au $j^{\text{ème}}$ caractère	ch[i:j]	ch[i:j]

9.1.5 Opérations sur les listes

Activité de découverte 9.6

Utiliser la console Python pour déterminer la valeur de chaque liste :

[1,3,5,7][0] ; [1,3,5,7][1] ; [1,3,5,7,9][1:4] ; [1,[3,5],7,9][1] ; [1,[3,5],7,9][1][0] ; [1,[3,5],7,9][1][1] ; [1,[3,5],7,9][2].

Solution. [1,3,5,7][0] vaut 1 ; [1,3,5,7][1] vaut 3 ; [1,3,5,7,9][1:4] vaut [3,5,7] ; [1,[3,5],7,9][1] vaut [3,5] ; [1,[3,5],7,9][1][0] vaut 3 ; [1,[3,5],7,9][1][1] vaut 5 ; [1,[3,5],7,9][2] vaut 7.

En Python, pour ajouter un élément x à une liste L, on utilise la fonction L.append(x).

Tableau 9.5 – Opérations sur les listes.

Opérations	Pseudo-code	Python
Nombre d'éléments de la liste L	longueur(L)	`len(L)`
1er élément de la liste L	L[0]	`L[0]`
$i+1$ ème élément de la liste L	L[i]	`L[i]`
Partie de la liste L, du $i+1$ ème au j ème élément	L[i:j]	`L[i:j]`
Ajouter l'élément e à la liste L	**Ajouter** e à L	`L.append(e)`

9.1.6 Convertir des données

Activité de découverte 9.7

Utiliser la console Python pour déterminer la valeur de chaque donnée : str(15) ; str(56.8) ; str(True) ; int('53') ; float('68.2')

Solution. str(15) vaut '15' ; str(56.8) vaut '56.8' ; str(True) vaut 'True' ; int('53') vaut 53 ; float('68.2') vaut 68.2.

Tableau 9.6 – Conversion de données.

Fonction	Effet
str(*x*)	Renvoie la donnée *x* dans une chaîne de caractères.
int(*ch*)	Renvoie, lorsque c'est possible, l'entier associé à la chaîne *ch*.
float(*ch*)	Renvoie, lorsque c'est possible, le réel associé à la chaîne *ch*.

9.2 STOCKAGE DES DONNÉES

Pour pouvoir réutiliser une donnée dans un programme informatique, il faut la stocker provisoirement dans un espace mémoire de l'ordinateur appelé **variable**. Cet espace peut être considéré comme une boîte portant un nom et dans laquelle on range une donnée. Lorsqu'on stocke une donnée dans une variable, on dit que la variable est **affectée** à cette donnée.

Exemple

En Python, on stocke la valeur 4.2 dans une variable *a1* en écrivant « `a1=4.2` ». Sur la console Python, réalisons cette affectation, appelons ensuite la valeur de `a1` puis celle de `10*a1+7`. Cela donne :

```
>>> a1=4.2
>>> a1
4.2
>>> 10*a1+7
49
```

En pseudo-code, on écrit « a1 **prend la valeur** 4.2 » ou plus simplement « a1 ← 4.2 ».

Tableau 9.7 – Affectation des variables.

	Instruction en pseudo-code	Instruction en Python
	nom_variable ← donnée	`nom_variable=donnée`
Exemples d'affectation	a ← 8	`a=8`
	nom ← 'Deschamp'	`nom='Deschamp'`
	ok ← Vrai	`ok=True`
	liste1 ← ['12',4,'azerty']	`liste1=['12',4,'azerty']`

En Python, comme dans tout langage de programmation, le nom d'une variable doit respecter un certain nombre de règles :

- son premier caractère doit être une lettre (majuscule ou minuscule) ou « _ », les caractères suivants peuvent être des lettres, des chiffres ou « _ » ;
- il ne doit pas être un mot réservé du langage (comme « print » ou « while »).
- Par exemple, une variable peut se nommer « aZe_Y44 » ou « _hr_t9 », mais pas « 4ahTR » ou « #qi4 ».

De plus, pour faciliter la lecture d'un algorithme, il est essentiel de donner à chaque variable un nom en lien avec sa valeur. Par exemple, il est préférable qu'une variable ayant pour valeur 'Paul' s'appelle *prenom* ou *prénom* plutôt que *d4j_5* ou *z*.

Nous présenterons dans un premier temps, les algorithmes comme suit, en déclarant le nom et le type de chaque variable, puis en plaçant la suite des instructions à effectuer entre des marqueurs de début et de fin.

```
Variables : nom-variable1(type), nom-variable2 (type), etc.
Début
  <suite d'instructions>
Fin
```

Lorsqu'on exécute un algorithme, les instructions sont effectuées les unes après les autres. Considérons l'algorithme suivant :

```
Variables : n, m, nb (entiers)
Début
  n ← 8
  m ← 3
  nb ← n^2-m*10
Fin
```

Lorsqu'on exécute cet algorithme, la variable *n* prend la valeur 8, puis la variable *m* prend la valeur 3 et enfin la variable *nb* prend la valeur de *n^2–m*10* donc $8^2 - 3 \times 10$, c'est-à-dire 34. Implémentons cet algorithme sur la console Python puis appelons la valeur de *nb* pour vérifier :

```
>>> n=8
>>> m=3
>>> nb=n**2-m*10
>>> nb
34
```

En langage Python, on peut faire des « affectations multiples », c'est-à-dire affecter plusieurs variables en une seule instruction. Plus précisément, les affectations `n=8` et `m=3` de l'exemple précédent peuvent s'effectuer en écrivant `n,m=8,3` (ou `m,n=3,8`).

Activité de découverte 9.8

Que vaut la variable z après exécution de l'algorithme suivant ?

```
Variables : z (entier)
Début
  z ← 6
  z ← 2*z
  z ← z+3
Fin
```

SOLUTION. Après exécution, *z* vaut 15. *z* prend d'abord la valeur 6, puis $2 \times 6 = 12$ et enfin $12 + 3 = 15$. On peut vérifier ce résultat avec la console Python en validant la suite d'instructions « z=6 », « z=2*z », « z=z+3 » puis « z ».

Comme nous venons de le voir, une variable peut participer à son affectation.

Activité de découverte 9.9

Sur la console Python, affecter une variable p à la valeur 31, demander son type, puis réaffecter cette variable p à la valeur 'azerty' et redemander son type.

Que dire du typage des variables en Python ?

SOLUTION.

```
>>> p=31
>>> type(p)
<class 'int'>
>>> p='azerty'
>>> type(p)
<class 'str'>
```

Le typage des variables n'est pas statique en Python, c'est-à-dire que l'on peut, au cours d'un programme, changer la valeur d'une variable par une autre valeur de type différent. On parle de **typage dynamique**.

9.3 LECTURE ET ÉCRITURE DES DONNÉES

Activité de découverte 9.10

Ouvrir l'éditeur Python (sur la console, sélectionner le menu File puis New File) puis recopier le programme suivant :

```
k=30
k=k+8
print("le nombre k vaut",k)
```

Enregistrer (File puis Save ou Ctrl+S) puis exécuter ce programme (Run puis Run Module ou F5).
Expliquer le résultat puis donner le rôle de la fonction « print » en Python.

SOLUTION. Le programme affiche « *Le nombre k vaut 38* ». Explications : k prend la valeur 30 puis k prend la valeur de $k + 8$ c'est-à-dire $30 + 8 = 38$.
L'instruction « `print` (données) » a pour rôle d'afficher des données.

Tableau 9.8 – Affichage de données (écriture).

Instruction en pseudo-code	Traduction Python	Résultat après exécution
Afficher données	`print` (données)	Affiche les données

Considérons l'algorithme suivant :

Variable : x (réel)
Début
  x ← 8.4
  Afficher "Le quart du nombre ",x," est",x/4
Fin

Implémentons en Python (dans l'éditeur) :

```
x=8.4
print("Le quart du nombre",x,"est",x/4)
```

Après exécution, on obtient, « *Le quart du nombre 8.4 est 2.1* ».

Pour inviter l'utilisateur à rentrer des données, on se sert de l'instruction « Saisir » nommée « `input` » en langage Python.

Tableau 9.9 – Saisie de données (lecture).

Type de la variable	Pseudo-code	Python
Chaîne de caractères	Saisir nom_var	`nom_var=input()`
Entier	Saisir nom_var	`nom_var=int(input())`
Réel	Saisir nom_var	`nom_var=float(input())`

L'algorithme suivant a pour rôle de demander un mot et un nombre entier *n* à l'utilisateur, puis d'afficher la somme de *n* et du nombre de lettres du mot saisi :

Variables : n (entier), mot (chaîne)
Début
  Afficher "Saisir un mot : "
  Saisir mot
  Afficher "Saisir un nombre entier : "
  Saisir n
  Afficher "Nombre de lettres du mot ",mot," + ",n," = ",n+longueur(mot)
Fin

Implémentons cet algorithme en Python (dans l'éditeur) :

```
mot=input("Saisir un mot : ")
n=int(input("Saisir un nombre entier : "))
print("Nombre de lettres du mot",mot,"+",n,"=",n+len(mot))
```

Après exécution, si l'on saisit par exemple « `azerty` » et « `3` », on obtient l'affichage « *Nombre de lettres du mot azerty + 3 = 9* ».

Nous écrirons maintenant chaque algorithme non trivial avec un en-tête muni de son nom, de son rôle, des données saisies et des données affichées :

```
--------------------------------------------------------------------------------
Nom de l'algorithme
Rôle : description rapide du rôle de l'algorithme
Entrée(s) : donnée(s) saisie(s)
Sortie(s) : donnée(s) affichée(s)
--------------------------------------------------------------------------------
Variables : nom-variable1(type), nom-variable2 (type), etc.
Début
| <partie principale>
Fin
```

9.4 Instructions conditionnelles

En algorithmique, on peut vouloir effectuer ou non une instruction selon qu'une condition est vraie ou fausse. On parle d'**instruction conditionnelle**. Le tableau 9.10 en présente la structure en pseudo-code et en Python.

Tableau 9.10 – **Structure des instructions conditionnelles.**

Pseudo-code	Python

Si <condition> **Alors**
|<instructions>
SinonSi <condition> **Alors**
|<instructions>
Sinon
|<instructions>
FinSi
	`if <condition> :`
	`    <instructions>`
	`elif <condition> :`
	`    <instructions>`
	`else :`
	`    <instructions>`

L'algorithme suivant demande à l'utilisateur un nombre entier puis affiche sa parité.

```
--------------------------------------------------------------------------------
Algorithme parité
Rôle : donne la parité d'un nombre rentré par l'utilisateur
Entrée : saisie d'un nombre entier
Sortie : affiche la parité du nombre
--------------------------------------------------------------------------------
Variable : n (entier)
Début
|  Afficher "Saisir un nombre entier : "
|  Saisir n
|  Si n%2==0 Alors
|  |  Afficher "Ce nombre est pair."
|  Sinon
|  |  Afficher "Ce nombre est impair."
|  FinSi
Fin
```

Implémentation en Python

```
n=int(input("Saisir un nombre entier : "))
if n%2==0:
    print("Ce nombre est pair.")
else:
    print("Ce nombre est impair.")
```

Une instruction conditionnelle peut se limiter à une structure du type « Si ... Alors ... » ou « Si ... Alors ... Sinon ... ».

Le mot-clé « Alors » du pseudo-code se traduit par « : » en Python.

En Python, il n'y a pas de marqueur de fin (comme FinSi), c'est l'indentation (le décalage) qui permet de délimiter les blocs d'instructions.

On peut aussi imbriquer les instructions conditionnelles comme ci-après :

```
Si <condition1> Alors
    Si <condition2> Alors
        <instructions>
    FinSi
FinSi
```

9.5 Instructions itératives

Pour exécuter plusieurs fois de suite un bloc d'instructions, on utilise une boucle. Si le nombre d'exécutions est connu à l'avance, on va privilégier une boucle « Pour », sinon une boucle « TantQue ».

9.5.1 La boucle Pour

Activité de découverte 9.11

*Exécuter le programme Python suivant. En déduire la définition de range(0,10) et le rôle de la structure « **for** nom_var **in** L: <instructions> » pour une liste L.*

```
for k in range(0,10):
    print(k)
```

Solution. Après exécution on obtient l'affichage des entiers compris entre 0 et 9. `range(0,10)` est la liste des entiers de 0 à 9. Lors de l'exécution de « `for nom_var in L: <instructions>` », la variable `nom_var` prend, une à une, toutes les valeurs des éléments de la liste *L*, et pour chacune d'elles, le bloc d'instruction `<instructions>` est exécuté.

Tableau 9.11 – Structure de la Boucle Pour.

En pseudo-code	En Python	Résultat
Pour *var* **dans** *liste* **Faire** | <instructions> **FinPour**	`for var in liste :` `    <instructions>`	La variable *var* prend toutes les valeurs des éléments de la liste `liste` et, pour chacune d'elles, exécution du bloc `<instructions>`.
Pour *var* **de** n à p **Faire** | <instructions> **FinPour**	`for var in range(n, p+1) :` `    <instructions>`	La variable *var* prend toutes les valeurs entières de l'entier `n` à l'entier `p`, et pour chacune d'elles, exécution du bloc `<instructions>`.

La variable *var* est appelée **compteur** de la boucle **« Pour »**.

En Python, `range(n, p)` désigne la liste $[n,n+1,...,p-1]$, donc par exemple `range(1, n)` désigne la liste $[1,2,...,n-1]$. De même, `range(0, n)` désigne la liste $[0,1,...,n-1]$, qui peut aussi se noter plus simplement `range(n)`.

Lorsque l'on écrit un algorithme ou un programme non trivial, il est bon d'écrire des **commentaires** à côté de certaines instructions. On utilise pour cela « # » en Python. On fait de même en pseudo-code (on peut aussi utiliser « // »). Les commentaires n'ont aucun effet lors de l'exécution. Ils constituent simplement une aide dans la lecture ou la relecture d'un programme ou d'un algorithme.

L'algorithme suivant calcule puis affiche la somme S = 10 + 11 + 12 + ... + 200 :

Algorithme somme
Rôle : calcul de la somme 10+11+...+200
Entrée : -------------
Sortie : affichage de la somme

Variables : S,k (entiers)
Début
| S ← 0 # initialisation
| **Pour** k de 10 à 200 **Faire**
| | S ← S + k
| **FinPour**
| **Afficher** S
Fin

Implémentation en Python :

```
S=0  # initialisation
for k in range(10, 201):
    # attention !  range(10, 201) et non range(10, 200)
    S=S+k
print(S)
```

Résultat après exécution : 20 055.

9.5.2 La boucle TantQue

Activité de découverte 9.12

Exécuter le programme Python suivant et en déduire le rôle de la structure « while <condition> : <instructions> ».

```
n=0
while n<10:
    n=n+3
    print(n)
```

SOLUTION. Après exécution on obtient l'affichage des nombres 3, 6, 9 puis 12. Lors de l'exécution, la variable n prend d'abord la valeur 0. Ensuite, tant que la condition $n < 10$ est vérifiée, on augmente n de 3 puis on affiche n. Voici le détail, étape par étape : $0 < 10$ donc $n = 0 + 3 = 3$ et on affiche 3 ; $3 < 10$ donc $n = 3 + 3 = 6$ et on affiche 6 ; $6 < 10$ donc $n = 6 + 3 = 9$ et on affiche 9 ; $9 < 10$ donc $n = 9 + 3 = 12$ et on affiche 12 ; on n'a pas $12 < 10$ donc on n'effectue plus le bloc d'instructions « $n = n + 3$; print(n) ». La structure « while <condition> : <instructions> » effectue le bloc d'instructions `<instructions>` tant que la condition `<condition>` est vraie.

Tableau 9.12 – **Structure de la boucle TantQue.**

En pseudo-code	En Python	Résultat
TantQue <condition> **Faire** │<instructions> **FinTantQue**	`while <condition> :` `    <instructions>`	Tant que la condition `<condition>` est vraie, on effectue le bloc d'instructions `<instructions>`.

L'algorithme suivant détermine et affiche le plus petit entier n tel que $1{,}01^n > 200$:

```
Variable : n (entier)
Début
  n ← 1        # initialisation à 1 par exemple
  TantQue 1.01^n<=200 Faire
    n ← n + 1
  FinTantQue
  Afficher n
Fin
```

Implémentation en Python :

```
n=1
while 1.01**n<=200:
    n=n+1
print(n)
```

Résultat après exécution : 533

9.6 Fonctions et procédures

9.6.1 Les fonctions usuelles

Les fonctions usuelles (exp, ln, sin, etc.) sont disponibles en Python en chargeant le module `math` à l'aide de l'instruction `import math`. L'appel de la fonction s'effectue alors en écrivant `math.nom_fonction`.

Exemple

On a affiché ci-dessous une valeur approchée de $\sqrt{3}$:

```
>>> import math
>>> print(math.sqrt(3))
1.7320508075688772
```

9.6.1 Création de fonctions et procédures

On peut créer en algorithmique toute sorte de fonctions, comme la fonction f définie par $f(x) = 3x^2 - 5x + 2$ ou des fonctions beaucoup plus originales comme la fonction g qui à tout nombre entier n renvoie la chaîne de caractères 'nombre pair' ou 'nombre impair' selon la valeur de n.

Activité de découverte 9.13

1) Exécuter le programme Python suivant :

```
def f(x):
    return(x*x)
def g(x):
    print(x*x)
```

2) Dans la console Python :
a) Saisir et valider f(5) puis f(6), g(5) et g(6). Que représentent f et g ?
b) Saisir « a=f(3) » puis appeler a (taper « a » puis valider).
c) De même, saisir « b=g(3) » puis appeler b. Que constate-t-on ?

Solution. 2) a) $f(5)$ et $g(5)$ renvoient 25. $f(6)$ et $g(6)$ renvoient 36. f et g représentent la fonction carrée, c'est-à-dire qu'elles sont définies par $f(x) = g(x) = x^2$.
b) On obtient $a = f(3) = 9$.
c) Lorsqu'on appelle b, rien ne se passe.
Explications : la fonction f, telle qu'elle est définie est une « vraie » fonction au sens algorithmique, car elle **retourne** à tout réel x une valeur $f(x)$, qui est ensuite stockable dans une variable. En algorithme et en programmation, on dit que f est une **fonction.**
A contrario, g n'est pas, en ce sens, une fonction car elle ne retourne pas une valeur pouvant affecter une variable. Elle ne fait qu'effectuer des instructions (ici : affichage du résultat d'un calcul). On dit que g est une **procédure**. La variable x est appelée **paramètre d'entrée** de la fonction f et de la procédure g.

Activité de découverte 9.14

Exécuter le programme Python suivant puis, dans la console, saisir et valider mystere(3) puis mystere(4) et mystere(5).
Justifier que **mystere(n)** *est une fonction et donner son rôle.*

```
def mystere(n):
    if n%2==0:
        return('oui')
    else:
        return('non')
```

SOLUTION. On obtient 'non' pour *mystere(3)*, 'oui' pour *mystere(4)* et 'non' pour *mystere(5)*. *mystere*(*n*) est une fonction car une valeur ('oui' ou 'non') est bien retournée pour toute valeur de *n* : *mystere*(*n*) renvoie 'oui' si le paramètre d'entrée *n* est un entier pair et 'non' sinon.

On peut créer toutes sortes de fonctions, comme par exemple :

- la fonction **listediv(n)** qui retourne la liste des diviseurs d'un entier *n* ;
- la fonction **somme(L)** qui retourne la somme des éléments d'une liste *L* de nombres ;
- la fonction **originale(n,ch)** qui à un nombre entier *n* et à une chaîne de caractères *ch* retourne la liste [*n+longueur*(*ch*), *n*ch+'azerty'*]. *n* et *ch* sont les paramètres d'entrée et le résultat est de type *liste* ;
- etc.

Tableau 9.13 – Structures d'une fonction et d'une procédure.

En pseudo-code	En Python

Fonction nom_fct(paramètres et types) :
Début
<instructions>
Retourner (donnée de sortie)
FinFonction
	`def nom_fct(paramètres):`
	`    <instructions>`
	`    return(donnée de sortie)`

Procédure nom_proc(paramètres et types) **Début** <instructions> **FinProcédure**	`def nom_proc(paramètres):` `    <instructions>`

La fonction suivante retourne à tout entier naturel *n*, la liste de ses chiffres :

```
Fonction liste_chiffres (n : entier) : liste
Variables locales : k (entier), L (liste), ch (chaîne)
Début
    ch ← str(n)          # chaîne de caractères associée à n
    L ← []               # initialisation (liste vide)
    Pour k de 0 à longeur(ch)-1 Faire
        Ajouter int(ch[k]) à L    # int(ch[k]) renvoie le chiffre associé à ch[k]
    FinPour
    Retourner (L)
FinFonction
```

Implémentation en Python :

```
def liste_chiffres(n):
    ch,L=str(n),[]
    for k in range(len(ch)):
        L.append(int(ch[k]))
    return(L)
```

Activité de découverte 9.15

1) Exécuter le programme Python.

```
a=6
def f(x):
     a=20
     b=2*x
     return(b+1)
```

2) Dans la console Python, appeler la valeur de f(4) puis celle de a, et enfin celle de b. Tenter d'expliquer le résultat.

SOLUTION. On obtient 9 pour $f(4)$ $(4 \times 2 + 1 = 9)$ et 6 pour la variable a. Pour b, on obtient le message d'erreur « *name 'b' is not defined* ».
Explications : lorsqu'une variable est définie dans une fonction, elle est créée lors de l'appel de la fonction, puis détruite lors de la sortie de celle-ci. La variable b n'a donc une existence que lors de l'appel de la fonction f. Quant à la variable a, l'instruction « a=20 » se comporte comme la déclaration d'une nouvelle variable, détruite après la sortie de f. Lorsqu'on appelle la variable a, on obtient donc 6.

Une variable définie dans une fonction ou une procédure est appelée **variable locale**, elle est détruite après sortie de la fonction ou de la procédure. Toute autre variable est appelée **variable globale**. En pseudo-code, on déclare toute variable, qu'elle soit locale ou globale.

Terminons maintenant la structure générale de l'écriture d'un algorithme.

Structure générale de l'écriture d'un algorithme

Nom de l'algorithme
Rôle : description rapide du rôle de l'algorithme
Entrée(s) : donnée(s) saisie(s)
Sortie(s) : donnée(s) affichée(s)
Variables globales : nom_variable1(type), nom_variable2 (type), etc.

. . . . Fonctions et procédures . . . .
Fonction nom_fonction(liste des paramètres et types) : type du résultat
Rôle : description rapide du rôle de la fonction
Variables locales : nom_var_locale1 (type), nom_var_locale2 (type), etc.
Début
| <instructions>
FinFonction

De même pour toutes les autres fonctions.

```
Procédure nom_procédure(liste des paramètres et types)
Rôle : description rapide du rôle de la procédure
Variables locales : nom_var_locale1 (type), nom_var_locale2 (type), etc.
Début
| <instructions>
FinProcédure
```

De même pour toutes les autres procédures.

```
........................ Partie principale ............................
Début
| <partie principale>
Fin
```

Exemple

Le plus petit chiffre de 3 528 est 2. L'algorithme suivant détermine combien d'entiers naturels de 4 chiffres ont leur plus petit chiffre égal à 2. Celui-ci fait appel à la fonction **pluspetitchiffre(n)** qui renvoie à tout entier $n \geq 0$ le plus petit chiffre de n.

```
----------------------------------------------------------------------
Algorithme combien_nbde4chiffres_pluspetitchiffre2
Rôle : détermine combien de nb de 4 chiffres ont pour plus petit chiffre 2
Entrée : --------------
Sortie : Nombre de nb de 4 chiffres de plus petit chiffre 2
----------------------------------------------------------------------

Variables globales : nb, compt (entiers)

........................... Fonction ..................................

Fonction pluspetitchiffre(n: entier) : entier
Rôle : renvoie le plus petit chiffre du nombre n
Variables locales : k, min (entiers), ch (chaîne)
Début ......................
| chiffremin ← 9          # initialisation : plus petit chiffre = 9
| ch ← str(n)            # chaîne de caractères associée à n
| Pour k de 0 à longueur(ch)-1 Faire
| | Si int(ch[k])< chiffremin Alors
| | | chiffremin ← int(ch[k])
| | FinSi
| FinPour
| Retourner chiffremin
FinFonction

........................ Partie principale. ...........................

Début
| compt ← 0              # initialisation du compteur
| Pour nb de 1000 à 9999 Faire
| | Si pluspetitchiffre(nb)==2 Alors
| | | compt ← compt + 1
| | FinSi
| FinPour
| Afficher compt
Fin
```

Implémentation en Python :

```
########################### Fonction ################################
def pluspetitchiffre(n):
    chiffremin=9
    ch=str(n)
    for k in range(len(ch)):
        if int(ch[k])<chiffremin:
            chiffremin=int(ch[k])
    return(chiffremin)
##################### Partie principale ###########################
compt=0
for nb in range(1000,10000):
    if pluspetitchiffre(nb)==2:
        compt=compt+1
print(compt)
```

Résultat après exécution : 1695.

Lorsqu'on passe par l'instruction « Retourner » (ou `return`), on sort définitivement de la fonction. Considérons par exemple la fonction *essai*(*n*) suivante. Le compteur de la boucle ne prend que les valeurs 1 puis 2 avant de sortir de la fonction et de retourner la valeur VRAI.

```
Fonction essai (n : entier) : booléen
Variable locale : k (entier)
Début
  Pour k de 1 à 100 Faire
    Si k%2==0 Alors
      Retourner VRAI
    FinSi
  FinPour
FinFonction
```

9.7 Récursivité

La fonction **puissance(x,n)** suivante renvoie à tout réel x, et tout entier naturel n, le nombre x^n. Elle utilise une boucle, on dit qu'il s'agit d'une **fonction itérative** :

```
Fonction puissance(x: réel, n: entier) : réel
Rôle : calcul itératif de x^n
Variables locales : p, k (entiers)
Début
  p ← 1
  Si n>0 Alors
    Pour k de 1 à n Faire
      p ← p*x
    FinPour
  FinSi
  Retourner(p)
FinFonction
```

On peut remarquer que, pour tout réel x et tout entier $n \geq 1$, on a $x^n = xx^{n-1}$. Écrivons une seconde fonction puissanceR(x,n) renvoyant le nombre x^n, utilisant la relation puissanceR(x,n)=x*puissanceR(x,n–1) :

```
Fonction puissanceR(x: réel, n: entier) : réel
Rôle : calcul récursif de xⁿ
Début
  Si n==0 Alors
  | Retourner(1)
  Sinon
  | Retourner(x*puissance(x,n-1))
  FinSi
FinFonction
```

Une telle fonction fait appel à elle-même, on dit qu'il s'agit d'une **fonction récursive**.

Une **fonction récursive** est une fonction s'appelant elle-même. Plus généralement, un **algorithme récursif** est un algorithme s'appelant lui-même.

Lorsqu'on souhaite écrire une fonction $u(n)$ retournant le terme u_n d'une suite (u_n), si cette suite est définie par récurrence, il est alors plus commode de passer par une fonction résursive. Cependant, son exécution peut dans certains cas, s'avérer être beaucoup plus lente que si cette fonction avait été construite de manière itérative.

Exemples

La suite de Fibonacci est la suite (u_n) définie par $u_0 = u_1 = 1$ et pour tout entier $n \geq 2$ par $u_n = u_{n-1} + u_{n-2}$. La fonction récursive **fib(n)** suivante, écrite en Python, renvoie u_n à tout entier naturel n :

```
def fib(n):
    if n<=1:
        return(1)
    else:
        return(fib(n-1)+fib(n-2))
```

L'appel de fib(5) renvoie 8 quasi instantanément, alors que l'on obtient la valeur de fib(35) en 22 secondes environ, et un temps encore plus long lorsque n augmente. On résout ce problème avec la fonction itérative **fibI(n)** suivante :

```
def fibI(n):
    a,b=1,1
    if n>1:
        for k in range(1,n):
            b=a+b
            a=b-a
    return(b)
```

Exercices corrigés

9.1 Que vaut la variable *c* après exécution de l'algorithme suivant ?

```
Variables : a, b, c (chaînes)
Début
   a ← 'az'
   b ← 'er'
   c ← a+b+'ty'
Fin
```

9.2 Que vaut la variable *a1* après exécution de l'algorithme suivant ?

```
Variables : a1 (réel), a2, a3 (entiers)
Début
   a1 ← 0.1
   a2 ← 5
   a3 ← 100
   a1 ← a1+a2/a3
Fin
```

9.3 Que valent les variables *x* et *n* après exécution de l'algorithme suivant ?

```
Variables : x (liste), n (entier)
Début
   x ← [1,2]
   Ajouter 3 à x
   n ← x[0]+x[1]+x[2]
Fin
```

9.4 Expliquer le rôle de l'algorithme suivant puis écrire son en-tête :

```
Variables : a,b,c (réels)
Début
   Afficher "Saisir un nombre a :"
   Saisir a
   Afficher "Saisir un nombre b :"
   Saisir b
   c ← a
   a ← b
   b ← c
   Afficher "a=",a," ; b=",b
Fin
```

9.5 Écrire un algorithme demandant à l'utilisateur un nombre entier de trois chiffres, puis affichant la somme des chiffres de ce nombre. Implémenter cet algorithme en Python.

9.6 Compléter l'algorithme suivant :

```
Variable : L (liste)
Début
    Saisir L            # Liste d'au moins 3 éléments
    Si (L[0]≤L[1] et L[1]≤L[2]) Alors
        Afficher "Les trois premiers éléments ......................."
    FinSi
Fin
```

9.7 Donner le rôle de l'algorithme suivant, le simplifier, puis implémenter en Python l'algorithme obtenu.

```
Variable : n (entier)
Début
    Afficher "Rentrer un nombre entier : "
    Saisir n
    Si n%3==0 Alors
        Si n>100 Alors
            Afficher"C'est bon !"
        FinSi
    FinSi
Fin
```

9.8 Écrire un algorithme (avec un en-tête) demandant à l'utilisateur de saisir un nombre puis affichant « *Ce nombre est strictement positif.* », « *Ce nombre est nul.* » ou « *Ce nombre est strictement négatif.* » selon le cas.

9.9 Écrire un algorithme calculant puis affichant $\sum_{k=1}^{50} k^2$ c'est-à-dire la somme $1^2 + 2^2 + ... + 50^2$. Implémenter en Python, puis exécuter le programme.

9.10 Quel est le rôle de l'algorithme suivant ? Implémenter en Python puis exécuter le programme.

```
Variables : p,x (entiers), L (liste)
Début
    p ← 1
    L ← [2,4,6,8,10]
    Pour x dans L Faire
        p ← p*x
    FinPour
    Afficher p
Fin
```

9.11 Écrire un algorithme qui génère et affiche la liste des diviseurs d'un nombre entier naturel rentré par l'utilisateur. Implémenter et exécuter cet algorithme en Python avec le nombre 646.

9.12 Quel est le rôle de l'algorithme suivant ?

```
Variables : k, compt (entiers), texte (chaîne)
Début
    Afficher "Saisit un texte : "
    Saisir texte
    k ← 0
    compt ← 0
    TantQue k<=longueur(texte) - 1 Faire
        Si texte[k]=='e' Alors
            compt ← compt + 1
        FinSi
        k ← k+1
    FinTantQue
    Afficher compt
Fin
```

9.13 Écrire un algorithme demandant à l'utilisateur un pseudo d'au moins quatre caractères, jusqu'à ce qu'il soit correct, puis affichant « *Bonjour <pseudo>!* ».

9.14 1) Écrire un algorithme de jeu de hasard dont la règle est la suivante : on tire au hasard un nombre entier de trois chiffres que l'utilisateur doit deviner. Tant qu'il ne l'a pas trouvé, on lui indique le nombre de chiffres corrects. Une fois trouvé, on affiche son nombre d'essais.

2) Implémenter en Python cet algorithme en important au préalable le module random (écrire en début de programme l'instruction `import random`). Le choix d'un nombre entier de trois chiffres au hasard peut alors s'obtenir avec :

```
int(899.9*random.random()+100)
```

9.15 1) Écrire une fonction booléenne position(mot,texte,n) qui à deux chaînes de caractères *mot* et *texte*, et à un entier naturel *n*, renvoie VRAI si la chaîne *mot* apparaît dans la chaîne *texte* à partir du *n*-ième caractère, et FAUX sinon. Par exemple, on devra avoir : `position('aze,'clavier azerty',9)=VRAI` et `position('aze,'clavier azerty',8)=FAUX`.

2) Écrire une fonction compte(mot,texte) faisant appel à la fonction *position(mot,texte,n)*, qui compte le nombre d'apparitions de la chaîne *mot* dans la chaîne *texte*. Par exemple, on devra avoir : `compte('ne','une fine plume')=2`.

9.16 Écrire en Python une procédure comparaison(a,b) qui affiche le résultat de la comparaison de deux réels *a* et *b*. Par exemple, il devra résulter de l'appel de *comparaison(8,19)* l'affichage « 8<19 » et de l'appel de *comparaison(12,12)* l'affichage « 12=12 ».

9.17 On considère le programme Python suivant :

```
def u(n):
    return(2**n)
def g(x):
    N=0
    while u(N)<=x:
        N=N+1
    return(N)
```

1) Que représentent les nombres $u(0)$, $u(1)$, $u(2)$, $u(3)$, etc.

2) L'appel de $g(100)$ dans la console Python renvoie 7. Interpréter ce résultat.

3) Pourquoi $g(x)$ est-elle définie quel que soit $x > 0$?

9.18 Écrire une fonction récursive puissance_rap(x,n) qui à tout réel x et à tout entier naturel n, retourne x^n, en utilisant la propriété : pour tout entier naturel p, on a $x^{2p} = (x^2)^p$ et $x^{2p+1} = x(x^2)^p$ (on parle de puissance rapide).

9.19 Soit (u_n) la suite définie par $u_0 = 2$ et $u_{n+1} = 2u_n - 1$.

1) Écrire une fonction récursive u(n) qui à l'entier $n \geq 0$ renvoie u_n. Implémenter en Python, puis vérifier que $u_{10} = 1\,025$.

2) Même question avec une fonction itérative.

9.20 Pour tout entier naturel n, on appelle « factorielle n », le nombre que l'on note $n!$ et défini par : $0! = 1$; $1! = 1$; $2! = 1 \times 2 = 2$; … ; $n! = 1 \times 2 \times 3 \times ... \times n$. Par exemple $4! = 1 \times 2 \times 3 \times 4 = 24$.

1) En remarquant que, pour tout $n \geq 1$, $n\,! = n \times (n-1)!$, écrire une fonction récursive factR(n) qui à tout entier $n \geq 0$ renvoie $n!$.

2) Écrire une fonction itérative factI(n) qui à tout entier $n \geq 0$ renvoie $n!$.

3) Que renvoie à tout entier naturel n, la fonction **inconnue(n)** suivante ?

```
Fonction inconnue(n: entier) : entier
Variable locale : p (entier)
Début
  p ← 1
  TantQue n>1 Faire
    p ← n*p
    n ← n-1
  FinTantQue
  Retourner(p)
FinFonction
```

Solutions

9.1 Après exécution, c vaut `'az'+'er'+'ty'='azerty'`. On peut le vérifier sur la console Python en tapant `a,b='az','er'` puis `c=a+b+'ty'` et en appelant enfin la valeur de c.

9.2 Après exécution, a1 vaut 0,1 + 5/100 = 0,15. (Vérifier sur la console Python.)

9.3 x vaut [1,2,3] et n vaut 6. En effet, après exécution, x prend la valeur [1,2] puis [1,2,3]. Ensuite n prend la valeur 1 + 2 + 3 = 6. On peut le vérifier sur la console Python en saisissant `x = [1,2]` puis `x.append(3)` puis `x = x[0] + x[1] + x[2]` et en appelant enfin les valeurs de x et n.

9.4 L'algorithme intervertit les valeurs des variables a et b puis les affiche. En effet, la valeur de a est d'abord stockée dans la variable c, la valeur de b est ensuite stockée dans a (qui a donc pris la valeur initiale de b) puis c (qui contient la valeur initiale de a) est stockée dans b (qui a donc pris la valeur initiale de a), d'où l'interversion. Voici l'en-tête complété de l'algorithme :

Algorithme interversion
Rôle : intervertit deux valeurs
Entrées : a,b (réels)
Sortie : affichage de a et b intervertis

Remarquons que Python peut effectuer directement cette opération en écrivant `a,b=b,a`.

9.5 1) Algorithme :

Algorithme somme_chiffres
Rôle : donne la somme des chiffres d'un nombre de trois chiffres
Entrée : saisie d'un nombre entier de trois chiffres
Sortie : affichage de la somme des chiffres du nombre

Variables : n (chaîne), somme (entier)
Début
  Afficher "Rentrer un nombre entier de trois chiffres : "
  Saisir n
  somme ← int(n[0])+int(n[1])+int(n[2])
  Afficher"La somme des chiffres du nombre ",n," est ",somme,"."
Fin

Remarquons que n[0], n[1] et n[2] sont respectivement les 1^er^, 2^e^ et 3^e^ caractères de la chaîne de caractères n rentrée par l'utilisateur, donc int(n[0]), int(n[0]) et int(n[0]) sont respectivement les 1^er^, 2^e^ et 3^e^ chiffres du nombre.

Implémentation en Python

```
n=input("Rentrer un nombre entier compris entre 100 et 999 :")
somme=int(n[0])+int(n[1])+int(n[2])
print("La somme des chiffres du nombre",n,"est",somme,".")
```

9.6 "Les trois premiers éléments de cette liste sont rangés par ordre croissant".

9.7 L'algorithme affiche le message « *C'est bon !* », que si l'entier n saisi par l'utilisateur est à la fois divisible par 3 et supérieur à 100. Algorithme simplifié :

Début
  Afficher "Rentrer un nombre entier : "
  Saisir n
  Si n%3==0 et n>100 **Alors**
    Afficher("C'est bon !")
  FinSi
Fin

Implémentation en Python

```
n = int(input("Rentrer un nombre entier : "))
if n%3==0 and n>100:
    print("C'est bon !")
```

9.8

Algorithme signe_nombre
Rôle : détermination du signe d'un nombre saisi
Entrée : saisie d'un nombre entier
Sortie : affichage du signe du nombre

```
Variable : x (réel)
Début
|  Afficher "Saisir un nombre : "
|  Saisir x
|  Si x<0 Alors
|  |  Afficher "Ce nombre est strictement négatif."
|  SinonSi x==0 Alors
|  |  Afficher "Ce nombre est nul."
|  Sinon
|  |  Afficher "Ce nombre est strictement positif."
|  FinSi
Fin
```

Implémentation en Python

```
x=float(input("Saisir un nombre : "))
if x<0:
    print("Ce nombre est strictement négatif.")
elif x==0:
    print("Ce nombre est nul.")
else:
    print("Ce nombre est strictement positif.")
```


9.9

Algorithme somme_carrés
Rôle : calcule et affiche la somme 1^2+2^2+...+50^2
Entrée : --------------
Sortie : affichage de la somme

```
Variables : s,k (entiers)
Début
|  s ← 0              # initialisation
|  Pour k de 1 à 50 Faire
|  |  s ← s+k^2
|  FinPour
|  Afficher s
Fin
```

Implémentation en Python

```
s=0
for k in range(1,51):
    s=s+k**2
print(s)
```

Résultat après exécution : 42 925.

9.10 Rôle : calcul puis affichage de $\prod_{k=1}^{5} 2k$, c'est-à-dire de $2 \times 4 \times 6 \times 8 \times 10$.

Implémentation en Python

```
p=1
L=[2,4,6,8,10]
for x in L:
    p=p*x
print(p)
```

Résultat après exécution : 3 840.

9.11

```
--------------------------------------------------------------------------------
Algorithme liste_diviseurs
Rôle : génère la liste des diviseurs d'un nombre
Entrée : saisie d'un nombre entier
Sortie : affichage de la liste des diviseurs
--------------------------------------------------------------------------------
Variables : n,k (entiers), L (liste)
Début
  L ← []                          # initialisation : liste vide
  Afficher "Saisir un nombre entier : "
  Saisir n
  Pour k de 1 à n Faire
    Si n%k==0 Alors
      Ajouter k à L
    FinSi
  FinPour
  Afficher L
Fin
```

Implémentation en Python

```
L=[]  # initialisation : liste vide
n=int(input("Saisir un nombre entier :"))
for k in range(1,n+1):
    if n%k==0:
        L.append(k)
print(L)
```

Résultat après exécution avec $n = 646$: [1, 2, 17, 19, 34, 38, 323, 646].

9.12 Rôle : détermine le nombre de caractères « e » du texte saisi par l'utilisateur.

9.13

Algorithme pseudo
Rôle : obtenir un pseudo correct puis afficher bonjour <pseudo>
Entrée : pseudo de l'utilisateur
Sortie : message « Bonjour <pseudo> »

```
Variable : pseudo (chaîne)
Début
   Afficher "Saisir un pseudo d'au moins 4 caractères : "
   Saisir pseudo
   TantQue longueur(pseudo)<4 Faire
      Afficher "Pseudo incorrect, veuillez le saisir de nouveau : "
      Saisir pseudo
   FinTantQue
   Afficher "Bonjour ", pseudo, " !"
Fin
```

Implémentation en Python

```
pseudo = input("Saisir un pseudo d'au moins 4 caractères : ")
while len(pseudo)<4 :
    pseudo = input("Pseudo incorrect, veuillez le saisir de
nouveau : ")
print("Bonjour",pseudo,"!")
```

9.14

Algorithme jeu_hasard

```
Variables : k, nombre, essai, nb_bonnes_rep (entiers), rep nb(chaînes)
Début
   nombre ← entier au hasard entre 100 et 999
   nb ← str(nombre)        # chaîne associée à nombre
   essai ← 1              # initialisation : compteur des essais
   Afficher "Saisir un entier de trois chiffres : "
   Saisir rep
   TantQue rep!=nb Faire
      nb_bonnes_rep ← 0
      essai ← essai + 1
      Pour k de 0 à 2 Faire
         Si nb[k]==rep[k] Alors
            nb_bonnes_rep ← nb_bonnes_rep + 1
         FinSi
      FinPour
      Afficher nb_bonnes_rep," chiffre(s) correct(s). Recommencez : "
      Saisir rep
   FinTantQue
   Afficher "Gagné en ",essai," essai(s) !"
Fin
```

Implémentation en Python

```
import random
nombre=int(899.9*random.random()+100)
nb=str(nombre)
essai=0
rep=input("devine nb : ")
while rep!=nb:
    nb_bonnes_rep=0
    essai=essai+1
    for k in range(3):
        if nb[k]==rep[k]:
            nb_bonnes_rep=nb_bonnes_rep+1
    rep=input(str(nb_bonnes_rep)+' chiffre(s) correct(s),
recommence : ')
print('gagné en '+str(essai)+' essai(s) !')
```

9.15 1) Fonction position(mot,texte : chaînes ,n: entier) : booléen.

Rôle : renvoie VRAI si *mot* apparaît dans *texte* à la position *n*, FAUX sinon.
Variable locale : i (entier)
Début
  Si n−1+longueur(mot)>longueur(texte) **Alors**
    Retourner(FAUX)   # mot trop long pour la position *n*
  FinSi
  Pour i de 0 à longueur(mot)-1 **Faire**
    Si texte[n-1+i]!=mot[i] **Alors**
      Retourner(FAUX)
    FinSi
  FinPour
  Retourner(VRAI)
FinFonction

Implémentation en Python

```
def position(mot,texte,n):
    if n-1+len(mot)>len(texte):
        return(False)  # mot trop long pour la position n
    for i in range(len(mot)):
        if texte[n-1+i]!=mot[i]:
            return(False)
    return(True)
```

2) Fonction **compte(mot,texte: chaînes)** : entier.

Rôle : compte le nombre d'apparitions de la chaîne *mot* dans la chaîne *texte*.
Variables locales : n,compteur (entiers)
Début
  compteur ← 0   # initialisation du compteur
  Pour n de 1 à longueur(texte)) **Faire**
    Si position(mot,texte,n) **Alors**
      compteur ← compteur + 1
    FinSi
  FinPour
  Retourner(compteur)
FinFonction

Implémentation en Python

```
def compte(mot,texte):
    compteur=0
    for n in range(1,len(texte)+1):
        if position(mot,texte,n):
            compteur=compteur+1
    return(compteur)
```

9.16 Procédure comparaison(a,b) :

```
def comparaison(a,b):
    if a<b:
        print(a,"<",b)
    elif a==b:
        print(a,"=",b)
    else:
        print(a,">",b)
```

9.17 1) Les nombres $u(0)$, $u(1)$, $u(2)$, $u(3)$, etc. représentent les termes de la suite (u_n) de terme général 2^n, c'est-à-dire $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, etc.

2) 7 est le plus petit entier naturel n tel que tel que $2^n > 100$.

3) $g(x)$ renvoie, pour tout réel $x > 0$, le plus petit entier naturel n tel que $2^n > x$. Un tel entier existe toujours car $\lim\limits_{n\to+\infty} 2^n = +\infty$ (puisque $2 > 1$), ce qui prouve que $g(x)$ est bien définie quel que soit $x > 0$.

9.18

```
Fonction puissance_rap(x: réel, n: entier) : réel
Rôle : calcul récursif de x^n
Début
   Si n==0 Alors
      Retourner(1)
   FinSi
   Si n%2==0 Alors
      Retourner(puissance_rap(x*x,n//2)
   Sinon
      Retourner(x*puissance_rap(x*x,n//2)
   FinSi
FinFonction
```

9.19 1)

```
Fonction u(n: entier) : entier
Rôle : calcul récursif de u_n
Début
   Si n==0 Alors
      Retourner(2)
   Sinon
      Retourner(2*u(n-1)-1)
   FinSi
FinFonction
```

Implémentation Python

```
def u(n):
    if n==0:
        return(2)
    else:
        return(2*u(n-1)-1)
```

2)

Fonction u(n: entier) : entier
Rôle : calcul itératif de u_n
Variables locales : k, res (entiers)
Début
  res ← 2
  Si n==0 **Alors**
    Retourner (res)
  FinSi
  Pour k de 1 à n **Faire**
    res ← 2*res-1
  FinPour
  Retourner (res)
FinFonction

Implémentation Python

```
def u(n):
    res=2
    if n==0:
        return(res)
    for k in range(1,n+1):
        res=2*res-1
    return(res)
```

9.20 1)

Fonction factR(n: entier) : entier
Rôle : calcul récursif de n!
Début
  Si n==0 **Alors**
    Retourner(1)
  Sinon
    Retourner(n*factR(n-1))
  FinSi
FinFonction

2)

```
Fonction factI(n: entier) : entier
Rôle : calcul itératif de n!
Variables locales : k,p (entiers)
Début
  p ← 1
  Si n==0 Alors
    Retourner (p)
  FinSi
  Pour k de 1 à n Faire
    p ← p*k
  FinPour
  Retourner(p)
FinFonction
```

3) La fonction **inconnue(n)** renvoie aussi $n!$ à tout entier naturel n.

TRAVAUX PRATIQUES 10

PLAN

10.1 Nombres parfaits
10.2 Évolution d'un salaire
10.3 Nombres premiers palindromes
10.4 Calcul formel
10.5 Calcul matriciel
10.6 Opérations sur les ensembles
10.7 Méthodes de tri
10.8 Cryptographie

OBJECTIFS

- Analyser un énoncé.
- Formaliser une démarche de résolution de problème.
- Écrire et interpréter des algorithmes.
- Pratiquer, expérimenter, tester, corriger.
- Implémenter et exécuter des algorithmes.

10.1 NOMBRES PARFAITS

Énoncé

a) Le problème

Un nombre entier $N > 1$ est dit **parfait** s'il est égal à la somme de tous ses diviseurs autres que lui-même. Par exemple :

- Le nombre 6 est parfait car ses diviseurs sont 1, 2, 3 et 6 et on a $6 = 1 + 2 + 3$.
- Le nombre 10 n'est pas parfait car ses diviseurs sont 1, 2, 5 et 10 et on n'a pas $10 = 1 + 2 + 5$.
- Le nombre 28 est parfait car ses diviseurs sont 1, 2, 4, 7, 14 et 28 et on a $28 = 1 + 2 + 4 + 7 + 14$.

Le but de ce TP est de déterminer le seul nombre parfait de trois chiffres et de calculer la somme des inverses de ses diviseurs.

b) Les questions

1) Quel est le rôle de la fonction **li(n)** suivante ?

```
Fonction li(n: entier) : liste
Rôle : .............................................
Variables locales : L (liste), k (entier)
Début
| L ← []
| Pour k de 1 à n Faire
| | Si n%k==0 Alors
| | | Ajouter k à L
| | FinSi
| FinPour
| Retourner(L)
FinFonction
```

2) Écrire une fonction booléenne **estparfait(n)** qui renvoie VRAI si le nombre entier n est parfait et FAUX sinon.

Par exemple, on devra avoir estparfait(6) = VRAI et estparfait(7) = FAUX.

3) Écrire une fonction **sominvdiv(n)** qui renvoie la somme des inverses des diviseurs d'un nombre entier n non nul.

Par exemple, puisque les diviseurs de 10 sont 1, 2, 5 et 10 on devra avoir sominvdiv(10)= 1/1 + 1/2 + 1/5 + 1/10 = 1 + 0,5 + 0,2 + 0,1 = 1,8.

4) Écrire un algorithme déterminant et affichant le seul nombre parfait de trois chiffres puis la somme des inverses des diviseurs de ce nombre. Cet algorithme devra faire appel aux fonctions précédentes. Implémenter sous Python cet algorithme puis exécuter le programme.

Corrigé

a) Algorithme

```
------------------------------------------------------------------------
Nom de l'algorithme Nombres_parfaits
Rôle : Détermination du nombre parfait de 3 chiffres et de la somme des
inverses de ses diviseurs
Entrée : --------------
Sorties : Affichage du nombre parfait de 3 chiffres et de la somme cherchée
------------------------------------------------------------------------
Variable globale : nb (entier)
.......................... Fonctions ..............................
Fonction li(n:entier) : liste
Rôle : détermine la liste des diviseurs de n
Variables locales : L (liste), k (entier)
Début
| L ← []
```

```
    Pour k de 1 à n Faire
        Si n%k==0 Alors
            Ajouter k à L
        FinSi
    FinPour
    Retourner(L)
FinFonction

Fonction estparfait(n: entier) : booléen
Rôle : détermine si le nombre n est parfait ou non
Variables locales : s, k (entiers)
Début
    s ← 1
    Pour k de 2 à n-1 Faire
        Si n%k==0 Alors
            s ← s+k
        FinSi
    FinPour
    Si n==s Alors
        Retourner(VRAI)
    FinSi
    Retourner(FAUX)
FinFonction

Fonction sominvdiv(n: entier) : réel
Rôle : calcule la somme des inverses des diviseurs du nombre n
Variables locales : som (réel), p (entier)
Début
    som ← 0
    Pour p dans li(n) Faire
        som ← som + 1/p
    FinPour
    Retourner(som)
FinFonction

. . . . . . . . . . . . . . . . . . . . . . . . Partie principale . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Début
    nb ← 100
    TantQue non(estparfait(nb)) Faire
        nb ← nb+1
    FinTantQue
    Afficher "Nombre parfait de 3 chiffres : ",nb
    Afficher "Somme des inverses de ses diviseurs : ",sominvdiv(nb)
Fin
```

b) Implémentation Python

```
######################### Fonction #############################
def li(n):
    L=[]
    for k in range(1,n+1):
        if n%k==0:
            L.append(k)
    return(L)
```

```
def estparfait(n):
    s=1
    for k in range(2,n):
        if n%k==0:
            s=s+k
        if n==s:
            return(True)
    return(False)

def sominvdiv(n):
    som=0
    for p in li(n):
        som=som+1/p
    return(som)

#################### Partie principale ########################
nb=100
while not(estparfait(nb)):
    nb=nb+1
print("Nombre parfait de 3 chiffres : ",nb)
print("Somme des inverses de ses diviseurs : ",sominvdiv(nb))
```

c) Résultats obtenus après exécution

Nombre parfait de trois chiffres : 496

Somme des inverses de ses diviseurs : 2.0

10.2 Évolution d'un salaire

Énoncé

a) Le problème

Au premier janvier 2014, Paul vient de se faire embaucher pour un salaire net mensuel de 2 000 € dans une entreprise qui a mis en place la règle d'évolution des salaires suivante : chaque mois, les salaires mensuels sont augmentés de 0,1 % puis de 1 €. Donc le salaire net mensuel de Paul au mois de février 2014 sera de 1,001 × 2 000 + 1 = 2 003 €, puis de 1,001 × 2 003 + 1 = 2 006,003 € au mois de mars, etc.

Le but du sujet est de déterminer le mois et l'année à partir desquels le salaire net mensuel de Paul dépassera 3 000 €.

b) Les questions

1) Quel est le rôle de la fonction **s(n)** ? Quelle est sa nature ?

Fonction s(n: entier) : réel   # nature de la fonction ....
Rôle : ....

```
Début
  Si n==1 Alors
    Retourner(2000)
  Sinon
    Retourner (s(n-1)*1,001+1)
  FinSi
FinFonction
```

2) Écrire une fonction **mois(n)** qui renvoie le *n*-ième mois de Paul dans l'entreprise.

Par exemple, on devra avoir mois(1)='janvier', mois(2)='février' et mois(14)='février'.

Cette fonction devra utiliser la liste suivante :

```
L=['janvier','février','mars','avril','mai','juin','juillet',
'août','septembre','octobre','novembre','décembre']
```

3) Écrire une fonction **annee(n)** retournant l'année correspondant au *n*-ième mois de Paul dans l'entreprise. Par exemple, on devra avoir annee(3)=2014, annee(12)=2014 et annee(13)=2015.

4) Écrire un algorithme donnant le mois et l'année à partir desquels le salaire net mensuel de Paul dépassera 3 000 €. Cet algorithme devra faire appel aux fonctions **s(n)**, **mois(n)** et **annee(n)**. Implémenter sous Python cet algorithme puis exécuter le programme.

Corrigé

a) Algorithme

```
----------------------------------------------------------------------------

Nom de l'algorithme Salaire_Paul
Rôle : Détermination mois et année à partir desquels : salaire de Paul > 3 000 €
Entrée : ---------------
Sorties : Affichage du mois et de l'année
----------------------------------------------------------------------------

Variable globale : N (entier)


........................ Fonctions ............................

Fonction s(n: entier) : réel     # Nature de la fonction : récursive
Rôle : calcule le salaire net mensuel au n-ième mois
Début
  Si n==1 Alors
    Retourner(2000)
  Sinon
    Retourner (s(n-1)*1.001+1)
  FinSi
FinFonction
```

Fonction mois(n: entier) : chaîne
Rôle : détermine le n-ième mois
Variable locale : L (liste)
Début
L ← ['janvier','février','mars','avril','mai','juin','juillet','août','septembre', 'octobre','novembre','décembre']
n ← (n-1)%12
Retourner(L[n])
FinFonction

Fonction annee(n: entier) : entier
Rôle : détermine l'année correspondant au *n*-ième mois
Variable locale : a (entier)
Début
a ← 2014+(n-1)//12
Retourner(a)
FinFonction

. . . . Partie principale . . . .
Début
N ← 1 # premier mois
TantQue s(N)<=3000 **Faire**
N ← N+1
FinTantQue
Afficher "Le salaire de Paul dépassera 3 000 € au mois de",mois(N),annee(N),"."
Fin

b) Implémentation Python

```
######################### Fonctions #############################
def s(n) :
    if n==1:
        return(2000)
    else :
        return(s(n-1)*1.001+1)

def mois(n):
    L=['janvier','février','mars','avril','mai','juin','juillet',
        'août','septembre','octobre','novembre','décembre']
    n=(n-1)%12
    return(L[n])

def annee(n):
        a=2014+(n-1)//12
        return(a)


#################### Partie principale ########################


N=1
while s(N)<=3000:
     N=N+1
print("Le salaire de Paul dépassera 3000 € au mois
de",mois(N),annee(N),".")
```

c) Résultats obtenus après exécution

Le salaire de Paul dépassera 3 000 € au mois de janvier 2038.

10.3 NOMBRES PREMIERS PALINDROMES

Énoncé

a) Le problème

Un entier naturel N est dit **premier** s'il possède exactement deux diviseurs : 1 et lui-même. Les nombres premiers sont 2, 3, 5, 7, 11, 13, 17, 19, 23, etc.

Un nombre entier est appelé **palindrome** s'il peut se lire indifféremment de gauche à droite ou de droite à gauche. Par exemple, 25852 et 423595324 sont des palindromes.

18181 est un nombre premier palindrome de cinq chiffres dont la somme des chiffres est 19. On se propose de déterminer le plus petit nombre premier palindrome de cinq chiffres dont la somme des chiffres est 19.

b) Les questions

1) Écrire une fonction booléenne **estpremier(n)** qui renvoie VRAI si n est un nombre premier et FAUX sinon.

Par exemple, on devra avoir **estpremier**(17) = VRAI et estpremier(18) = FAUX.

2) Écrire une fonction **somchiffre(n)** qui renvoie la somme des chiffres d'un nombre entier n.

Par exemple, on devra avoir **somchiffre**(2135) = 11.

3) Quel est le rôle de la fonction **change(n)** suivante ?

```
Fonction change(n: entier) : entier
Rôle : ..................................................................
Variables locales : ch,nb (chaînes), res (entier)
Début
 | nb ← str(n)
 | ch ← ''
 | Pour k de 0 à longueur(nb)-1 Faire
 |  | ch ← nb[k] + ch
 | FinPour
 | res ← int(ch)
 | Retourner(res)
FinFonction
```

4) Écrire un algorithme affichant le plus petit nombre premier palindrome de cinq chiffres dont la somme des chiffres est 19. Cet algorithme devra faire appel aux fonctions **estpremier**(n), **somchiffre**(n) et **change**(n).

Implémenter sous Python cet algorithme puis exécuter le programme.

Corrigé

a) Algorithme

```
---------------------------------------------------------------------------------
Nom de l'algorithme Nombre_premier_palindrome
Rôle : Détermination du plus petit nombre premier palindrome de cinq chiffres
Entrée : ---------------
Sortie : Affichage du plus petit nombre premier palindrome de cinq chiffres
---------------------------------------------------------------------------------
Variable globale :  N (entier)

. . . . . . . . . . . . . . . . . . . . . . . . Fonctions . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Fonction estpremier(n: entier) : booléen
Rôle : détermine si n est premier ou non
Variable locale : k (entier)
Début
   Si n<2 Alors
   |  Retourner(FAUX)              # 0 et 1 ne sont pas premiers
   Sinon
   |  Pour k de 2 à n-1 Faire
   |  |  Si n%k==0 Alors
   |  |  |  Retourner(FAUX)
   |  |  FinSi
   |  FinPour
   FinSi
   Retourner(VRAI)        # si aucun diviseur entre 2 et n-1, on renvoie VRAI
FinFonction

Fonction somchiffre(n: entier) : entier
Rôle : calcule la somme des chiffres de n
Variables locales : som (réel), nb (chaîne)
Début
   som ← 0
   nb ← str(n)                        # chaîne de caractères associée à n
   Pour k de 0 à longueur(nb)-1 Faire
   |  som ← som + int(nb[k])
   FinPour
   Retourner(som)
FinFonction

Fonction  change(n: entier) : entier
Rôle : retourne l'entier obtenu en inversant l'ordre des chiffres de n
Variables locales : ch,nb (chaînes), res
Début
   nb ← str(n)                          # chaîne associée à n
   chaine ← ''                               # chaîne vide
   Pour k de 0 à longueur(n)-1 Faire
   |  ch ← nb[k] + ch    # concaténation
   FinPour
   res ← int(ch)                    # nombre associé à la chaîne ch
   Retourner(res)
FinFonction
```

```
........................ Partie principale ............................

Début
  N ← 10000                  # plus petit entier de 5 chiffres
  TantQue non(estpremier(N) et change(N)==N et somchiffre(N)==19) Faire
    N ← N+1
  FinTantQue
  Afficher('Le plus petit nombre premier palindrome de
              cinq chiffres dont la somme des chiffres est 19, est ",N)
Fin
```

b) Implémentation Python

```
######################### Fonctions #############################

def estpremier(n):
      if n<2:
            return(False)
      else:
            for k in range(2,n):
                  if n%k==0:
                        return(False)
      return(True)

def somchiffre(n):
    som=0
    nb=str(n)
    for k in range(len(nb)):
       som=som+int(nb[k])
    return(som)

def change(n):
     nb=str(n)
     ch=''         # chaine vide
     for k in range(len(nb)):
        ch=nb[k]+ch
     res=int(ch)
     return(res)



##################### Partie principale #########################

N=10000
while not(estpremier(N) and change(N)==N and
somchiffre(N)==19):
      N=N+1
print("Le plus petit nombre premier palindrome de cinq chiffres
dont la somme des chiffres est 19, est",N)
```

c) Résultats obtenus après exécution

Le plus petit nombre premier palindrome de cinq chiffres dont la somme des chiffres est 19, est 16 561.

10.4 Calcul formel

Énoncé

a) Le problème

Le but de ce TP est de créer un algorithme de calcul formel sur les sommes de fractions. Plus précisément, il s'agit de demander à l'utilisateur une expression du type *a/b+c/d* (où *a*, *b*, *c* et *d* sont des nombres entiers naturels avec $b \neq 0$ et $d \neq 0$) et d'afficher le résultat sous la forme d'une fraction simplifiée.

b) Les questions

1) Écrire une fonction somme(ch) qui à une chaîne *ch* du type 'a/b+c/d' (où *a*, *b*, *c* et *d* sont des nombres entiers naturels avec $b \neq 0$ et $d \neq 0$) retourne la liste [a*d+b*c,b*d] correspondant au numérateur et au dénominateur de la somme des fractions a/b et c/d.

Par exemple, on devra avoir somme('1/2+1/3')=[5,6].

2) Quel est le rôle de la fonction récursive **p(a,b)** suivante ?

```
Fonction p(a,b: entiers) : entier
Rôle : ................................................................
Début
  Si b==0 Alors
    Retourner (a)
  Sinon
    Retourner (p(b,a%b))
  FinSi
FinFonction
```

3) Écrire une fonction fractionsimp(a,b) qui à des nombres entiers naturels *a* et *b* (avec $b \neq 0$) retourne la chaîne de caractères 'u/v' et où u/v est l'écriture simplifiée de la fraction a/b. Par exemple, on devra avoir fractionsimp(2,6) = '1/3'.

4) Écrire un algorithme demandant à l'utilisateur une somme du type a/b+c/d puis calculant et affichant le résultat sous la forme d'une fraction simplifiée. Implémenter sous Python puis exécuter le programme.

Corrigé

a) Algorithme

```
-------------------------------------------------------------------------
Nom de l'algorithme calcul_formel_somme_fractions
Rôle : calculer la somme de deux fractions avec résultat simplifié
Entrée : chaîne du type 'a/b+c/d'
Sortie : Affichage de la fraction sous forme simplifiée
-------------------------------------------------------------------------
Variables globales : expression, simp (chaînes), som (liste)
```

. . . . Fonctions . . . .

Fonction somme(ch: chaîne) : liste
Rôle : calcule la somme sous la forme d'une liste [numérateur,dénominateur]
Variables locales : k, a, b, c, d (entiers), L (liste)
Début
  L ← []
  Pour k de 0 à longueur(ch)-1 **Faire**
    Si ch[k]=='/' ou ch[k]=='+' **Alors**
      Ajouter k à L
    FinSi
  FinPour
  a ← int(ch[0:L[0]])
  b ← int(ch[L[0]+1:L[1]])
  c ← int(ch[L[1]+1:L[2]])
  d ← int(ch[L[2]+1:longueur(ch)])
  Retourner(a*d+b*c,b*d)
FinFonction

Fonction p(a,b: entiers): entier
Rôle : calcule le pgcd de a et b par l'algorithme d'Euclide
Début
  Si b==0 **Alors**
    Retourner (a)
  Sinon
    Retourner (p(b,a%b))
  FinSi
FinFonction

Fonction fractionsimp(a,b: entiers) : chaîne
Rôle : simplifie la fraction a/b
Variables locales : u,v (entiers)
Début
  u ← a//p(a,b)
  v ← b//p(a,b)
  Retourner(str(u)+'/'+str(v))
FinFonction

. . . . Partie principale . . . .

Début
  Afficher "Ecrire une somme de fractions de la forme a/b+c/d : "
  Saisir expression
  som ← somme(expression)
  simp ← fractionsimp(som[0],som[1])
  Afficher (expression+'='+simp)
Fin

b) Implémentation Python

```
######################### Fonctions ##############################

def somme(ch):
    L=[]
    for k in range(len(ch)):
      if ch[k]=='/' or ch[k]=='+':
            L.append(k)
    a=int(ch[0:L[0]])
```

```
    b=int(ch[L[0]+1:L[1]])
    c=int(ch[L[1]+1:L[2]])
    d=int(ch[L[2]+1:len(ch)])
    return(a*d+b*c,b*d)

def p(a,b):
    if b==0:
        return(a)
    else:
        return(p(b,a%b))

def fractionsimp(a,b):
    u,v=a//p(a,b),b//p(a,b)
    return(str(u)+'/'+str(v))

#################### Partie principale #########################

expression=input("Ecrire une somme de fractions du type
a/b+c/d : ")
som=somme(expression)
simp=fractionsimp(som[0],som[1])
print(expression+'='+simp)
```

c) Résultats obtenus après exécution

Par exemple, pour 1/2 + 5/6, on obtient 1/2 + 5/6 = 4/3.

10.5 Calcul matriciel

Énoncé

a) Le problème

Le but de ce TP est d'écrire un algorithme permettant de calculer des expressions du type $x_0 \times I_3 + x_1 \times A + x_2 \times A^2 + ... + x_n \times A^n$ pour A une matrice A carrée d'ordre 3 et pour des coefficients réels x_0, x_1, … et x_n (les coefficients et les éléments de la matrice étant rentrés par l'utilisateur). Nous définirons une matrice :

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}$$

comme la liste dont les éléments sont les listes des éléments de chaque ligne :

$$\left[\left[a_{1,1},a_{1,2},a_{1,3}\right],\left[a_{2,1},a_{2,2},a_{2,3}\right],\left[a_{3,1},a_{3,2},a_{3,3}\right]\right]$$

Dans tout le TP, toute matrice sera carrée d'ordre 3.

b) Les questions

1) Quel est le rôle de la fonction so(A,B) suivante ?

```
Fonction so(A,B: listes) : liste
Rôle : ..........................................................
Variables locales : i,j (entiers) C (liste)
Début
  C ← [[0,0,0],[0,0,0],[0,0,0]]
  Pour i de 0 à 2 Faire
    Pour j de 0 à 2 Faire
      C[i][j] ← A[i][j]+B[i][j]
    FinPour
  FinPour
  Retourner(C)
FinFonction
```

2) Écrire une fonction **produitR(x,A)** qui à un réel x et à une matrice A carrée d'ordre 3, retourne la matrice xA.

3) Écrire une fonction produit(A,B) qui à A et B deux matrices carrées d'ordre 3, retourne la matrice A × B.

On rappelle que si $A = (a_{i,j})$, $B = (b_{i,j})$, $C = (c_{i,j})$ et $C = A \times B$ alors, pour tous $i,j \in \{1,2,3\}$ on a : $c_{i,j} = a_{i,1} \times b_{1,j} + a_{i,2}\ ´\ b_{2,j} + a_{i,3} \times b_{3,j}$.

4) Écrire une fonction récursive **puissance(A,n)** qui retourne A^n, à une matrice A, carrée d'ordre 3, et un entier naturel n.

5) Écrire un algorithme demandant à l'utilisateur un entier naturel n et les coefficients $x_0, \ldots, x_n$, puis les éléments d'une matrice A, carrée d'ordre 3, et affichant la matrice $x_0 \times I_3 + x_1 \times A + x_2 \times A^2 + \ldots + x_n \times A^n$. Cet algorithme devra faire appel aux fonctions so(A,B), produitR(A,B) et puissance(A,n). Implémenter sous Python puis exécuter le programme.

Corrigé

a) Algorithme

```
------------------------------------------------------------------------
Nom de l'algorithme expression_matricielle
Rôle : calculer x_0*I_3+x_1*A^1+...+x_n*A^n
Entrées : n, coefficients x_0, ..., x_n et éléments de A
Sortie : Affichage de la matrice x_0*I_3+x_1*A^1+...+x_n*A^n
------------------------------------------------------------------------

Variables globales : N,p,q (entiers), coef (réel), listecoefs, matA, S (listes)

. . . . . . . . . . . . . . . . . . . . . . . . Fonctions. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Fonction so(A,B: listes) : liste
Rôle : calcule la somme A+B, pour A,B matrices carrées d'ordre 3
```

```
Variables locales : i,j (entiers), C (liste)
Début
  C ← [[0,0,0],[0,0,0],[0,0,0]]
  Pour i de 0 à 2 Faire
    Pour j de 0 à 2 Faire
      C[i][j] ← A[i][j]+B[i][j]
    FinPour
  FinPour
  Retourner(C)
FinFonction

Fonction produitR(x: réel, A: liste) : liste
Rôle : calcule le produit xA
Variables locales : i,j (entiers), C (liste)
Début
  C ← [[0,0,0],[0,0,0],[0,0,0]]
  Pour i de 0 à 2 Faire
    Pour j de 0 à 2 Faire
      C[i][j] ← x*A[i][j]
    FinPour
  FinPour
  Retourner(C)
FinFonction

Fonction  produit(A,B: listes) : liste
Rôle : calcule le produit AB
Variables locales : i,j,k (entiers), C (liste)
Début
  C ← [[0,0,0],[0,0,0],[0,0,0]]
  Pour i de 0 à 2 Faire
    Pour j de 0 à 2 Faire
      Pour k de 0 à 2 Faire
        C[i][j] ← C[i][j]+A[i][k]*B[k][j]
      FinPour
    FinPour
  FinPour
  Retourner(C)
FinFonction

Fonction  puissance(A: liste, n: entier) : liste
Rôle : calcule A^n
Début
  Si n==0 :
    Retourner([[1,0,0],[0,1,0],[0,0,1]])        # matrice identité I_3
  Sinon
    Retourner(produit(puissance(A,n-1),A))       # A^n=A*A^(n-1)
  FinSi
FinFonction

. . . . . . . . . . . . . . . . . . . . . . . Partie principale . . . . . . . . . . . . . . . . . . . . . . . . .

Début
  Afficher "Calcul de x_0*I_3+x_1*A^1+x_2*A^2+...+x_n*A"
  Afficher "Rentrer n : "
  Saisir N
```

```
listecoefs ← []                   # initialisation des coefficients : liste vide
matA ← [[0,0,0],[0,0,0],[0,0,0]]    # initialisation de la matrice A

Pour p de 0 à N Faire
    Afficher "x_",p,"="
    Saisir coef              # Saisie des coefficients x_0, ..., x_n
    Ajouter coef à listecoefs
FinPour

Pour p de 0 à 2 Faire
    Pour q de 0 à 2 Faire
        Afficher "a_",p+1,",",q+1,"="
        Saisir matA[p][q]         # Saisie des éléments de A
    FinPour
FinPour

S ← [[0,0,0],[0,0,0],[0,0,0]]
Pour p de 0 à N Faire
    S ← so(S,produitR(listecoefs[p],puissance(A,p)))
FinPour

Afficher "Matrice obtenue :"
Afficher(S[0])       # première ligne de la matrice x_0*I_3+...+x_n*A
Afficher(S[1])       # deuxième ligne de la matrice x_0*I_3+...+x_n*A
Afficher(S[2])       # troisième ligne de la matrice x_0*I_3+...+x_n*A
Fin
```

b) Implémentation Python

```python
######################### Fonctions #############################
def so(A,B):
    C=[[0,0,0],[0,0,0],[0,0,0]]
    for i in range(0,3):
        for j in range(0,3):
            C[i][j]=A[i][j]+B[i][j]
    return(C)

def produitR(x,A):
    C=[[0,0,0],[0,0,0],[0,0,0]]
    for i in range(0,3):
        for j in range(0,3):
            C[i][j]=x*A[i][j]
    return(C)

def produit(A,B):
    C=[[0,0,0],[0,0,0],[0,0,0]]
    for i in range(0,3):
        for j in range(0,3):
            for k in range(0,3):
                C[i][j]=C[i][j]+A[i][k]*B[k][j]
    return(C)

def puissance(A,n):
    if n==0:
        return([[1,0,0],[0,1,0],[0,0,1]])
    else:
        return(produit(puissance(A,n-1),A))
```

```
##################### Partie principale ###########################

print("Calcul de x_0*I_3+x_1*A^1+x_2*A^2+...+x_n*A")
N=int(input("Rentrer n : "))
listecoefs=[]
matA=[[0,0,0],[0,0,0],[0,0,0]]
for p in range(N+1):
    coef=float(input('x_'+str(p)+'='))
    listecoefs.append(coef)
for p in range(3):
    for q in range(3):
        matA[p][q]=float(input('a_'+str(p+1)+','+str(q+1)+'='))
S=[[0,0,0],[0,0,0],[0,0,0]]
for p in range(N+1):
    S=so(S,produitR(listecoefs[p],puissance(matA,p)))
print("Matrice obtenue : ")
print(S[0])
print(S[1])
print(S[2])
```

c) Résultats obtenus après exécution

Par exemple, pour A=[[1,2,3],[2,3,4],[3,4,5]], $n = 2$, $x_0 = 1$, $x_1 = 2$ et $x_2 = 3$, on obtient :

- [45,64,84]
- [64,94,122]
- [84,122,161]

Donc, si $A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{pmatrix}$, alors $I_3 + 2A + 3A^2 = \begin{pmatrix} 45 & 64 & 84 \\ 64 & 94 & 122 \\ 84 & 122 & 161 \end{pmatrix}$.

10.6 Opérations sur les ensembles

Énoncé

a) Le problème

Le but de ce TP est de déterminer la réunion et l'intersection deux ensembles dont les éléments sont saisis par l'utilisateur. Un ensemble sera considéré comme une liste sans doublon, c'est-à-dire sans éléments se répétant plusieurs fois.

Par exemple [] sera l'ensemble vide, [A,3,[Azerty,'OK'],8.9] sera un sensemble, mais la liste [0,2,5,4,2,3] n'en sera pas un.

b) Les questions

1) Quel est le rôle de la fonction **ver(ens)** suivante ?

```
Fonction ver(ens: liste): booléen
Rôle : .....................................................................
Variables locales : i,j (entiers)
Début
  Pour i de 0 à longueur(ens)-2 Faire
    Pour j de i+1 à longueur(ens)-1 Faire
      Si ens[i]==ens[j] Alors         # deux éléments identiques
        Retourner(FAUX)
      FinSi
    FinPour
  FinPour
  Retourner(VRAI)
FinFonction
```

2) Écrire une fonction **appartient(x,ens)** qui retourne VRAI si l'élément *x* est dans la liste ens et FAUX sinon.
3) Écrire une fonction **inter(ens1,ens2)** qui retourne la liste des éléments de l'intersection des ensembles **ens1** et **ens2**.
4) Écrire une fonction **union(ens1,ens2)** qui retourne la liste des éléments de la réunion des ensembles **ens1** et **ens2**.

5) Écrire un algorithme demandant à l'utilisateur deux ensembles A et B, vérifiant qu'il s'agisse bien d'ensembles, puis affichant les éléments de $A \cup B$ et $A \cap B$. Implémenter sous Python cet algorithme puis exécuter le programme. Pour la saisie d'une liste, utiliser la fonction eval(input()).

Corrigé

a) Algorithme

```
-------------------------------------------------------------------------
Nom de l'algorithme opérations_ensembles
Rôle : déterminer l'intersection et la réunion de deux ensembles
Entrées : ensemble1, ensemble2 (listes)
Sorties : Affichage de l'intersection et de la réunion de ensemble1 et ensemble2
-------------------------------------------------------------------------
Variables globales : ensemble1, ensemble2 (listes)
. . . . . . . . . . . . . . . . . . . . . . . . Fonctions . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Fonction ver(ens: liste) : booléen
Rôle : vérifie si la liste ens est sans doublon
Variables locales : i,j (entiers)
Début
  Pour i de 0 à longueur(ens)-2 Faire
    Pour j de i+1 à longueur(ens)-1 Faire
      Si ens[i]==ens[j] Alors      # deux éléments identiques
        Retourner(FAUX)
      FinSi
    FinPour
  FinPour
  Retourner(VRAI)                  # si aucun doublon
FinFonction
```

```
Fonction appartient(x: tout type possible, ens: liste) : booléen
Variable locale : k (entier)
Début
  Pour k de 0 à longueur(ens)-1 Faire
    Si x==ens[k] Alors
      Retourner(VRAI)
    FinSi
  FinPour
  Retourner(FAUX)
FinFonction

Fonction inter(ens1,ens2: listes) : liste
Variables locales : k (entier), res (liste)
Début
  res ← []
  Pour k de 0 à longueur(ens1)-1 Faire
    Si appartient(ens1[k],ens2) Alors
      Ajouter ens1[k] à res
    FinSi
  FinPour
Retourner(res)
FinFonction

Fonction  union(ens1,ens2: listes) : liste
Variables locales :  k (entier), res (liste)
Début
  res ← ens2
  Pour k de 0 à longueur(ens1)-1 Faire
    Si non(appartient(ens1[k],ens2)) Alors
      Ajouter ens1[k] à res
    FinSi
  FinPour
  Retourner(res)
FinFonction

....................... Partie principale ..........................

Début
  Afficher "Saisir un premier ensemble : "
  Saisir ensemble1
  TantQue non(ver(ensemble1)) Faire
    Afficher "Attention ! Saisir un premier ensemble (sans doublon) : "
    Saisir ensemble1
  FinTantQue

  Afficher " Saisir un second ensemble : "
  Saisir ensemble2
  TantQue non(ver(ensemble2)) Faire
    Afficher "Attention ! Saisir un second ensemble (sans doublon) : "
    Saisir ensemble2
  FinTantQue

  Afficher "L'intersection de ces ensembles est : ",inter(ensemble1,ensemble2)
  Afficher "La réunion de ces ensembles est : ",union(ensemble1,ensemble2)
Fin
```

b) Implémentation Python

```
######################### Fonctions ##############################

def ver(ens):
      for i in range(len(ens)-1):
          for j in range(i+1,len(ens)):
              if ens[i]==ens[j]:
                    return(False)
      return(True)

def appartient(x,ens):
    for k in range(len(ens)):
        if x==ens[k]:
            return(True)
     return(False)

def inter(ens1,ens2):
    res=[]
    for k in range(len(ens1)):
      if appartient(ens1[k],ens2):
          res.append(ens1[k])
    return(res)

def union(ens1,ens2):
    res=ens2
    for k in range(len(ens1)):
      if not(appartient(ens1[k],ens2)):
         res.append(ens1[k])
    return(res)
##################### Partie principale #########################

ensemble1=eval(input("Saisir un premier ensemble : "))
while not(ver(ensemble1)):
    ensemble1=eval(input("Attention ! Saisir un premier ensemble
(sans doublon): "))

ensemble2=eval(input(" Saisir un second ensemble : "))
while not(ver(ensemble2)):
    ensemble2=eval(input("Attention ! Saisir un second ensemble
(sans doublon): "))

print("L'intersection de ces ensembles est :
",inter(ensemble1,ensemble2))
print("La réunion de ces ensembles est :
",union(ensemble1,ensemble2))
```

c) Résultats obtenus après exécution

Exemple :

- Saisir un premier ensemble : [1,'az',2,3,4,8].
- Saisir un second ensemble : [4,'ok',5,'az',6,5].
- Attention ! Saisir un second ensemble (sans doublon) : [4,'ok',5,'az',6].
- L'intersection de ces ensembles est : ['az',4].
- La réunion de ces ensembles est : [4,'ok',5,'az',6,1,2,3,8].

10.7 MÉTHODES DE TRI

Énoncé

a) Le problème

Le but de ce TP est de comparer le temps d'exécution de plusieurs méthodes de tri d'une liste de nombres par ordre croissant. Parmi ces méthodes, on peut distinguer :

- Le **tri par sélection** : on recherche le plus petit élément de la liste, que l'on place en première position, puis le plus petit de la liste formée par les autres éléments, que l'on place en seconde position, etc.
- Le **tri à bulles** : pour une liste L, on compare tous les éléments successifs $L[k]$, $L[k+1]$. Lorsque deux éléments successifs ne sont pas rangés par ordre croissant, on les intervertit. Après un parcours complet de la liste, on recommence, et ainsi de suite jusqu'à ce qu'au cours d'un parcours, aucune interversion n'ai lieu.
- Le **tri rapide** : pour une liste L ayant au moins deux éléments, on retire le premier élément x de cette liste puis on crée deux listes, celle des éléments inférieurs ou égaux à x (liste 1) et celle des éléments supérieurs à x (liste 2). On fait de même avec ces nouvelles listes et ainsi de suite (il s'agit d'un algorithme récursif). On obtient la liste triée en prenant les éléments triés de la liste 1 puis x puis les éléments de la liste 2.

b) Les questions

1) Écrire une fonction **tri_select(L)** retournant la liste triée par la méthode de tri par sélection.

2) Écrire une fonction **tri_bulles(L)** retournant la liste triée par la méthode de tri à bulles.

3) Écrire une fonction récursive **tri_rapide(L)** retournant la liste triée par la méthode de tri rapide. On se servira de la fonction Python **pivot=L.pop()** qui affecte à la variable **pivot** la valeur du dernier élément de la liste L et qui le supprime de cette liste.

4) Écrire un algorithme permettant de comparer la rapidité d'exécution de chacune des fonctions précédentes. Pour cela :

- importer le module Python ***time***, en insérant en début de partie principale, l'instruction ***import time***, puis utiliser la fonction ***time.time()*** renvoyant le temps en seconde écoulé depuis le 1er janvier 1970 à 0 h 00.
- importer le module Python ***random***, en insérant en début de partie principale, l'intruction ***import random***, puis générer une liste *Lh* de 100 nombres réels, en utilisant l'instruction ***Lh.append(random.random())*** qui insère dans la liste Lh un nombre réel choisi au hasard entre 0 et 1.

5) Classer ces méthodes de tri, de la plus rapide à la moins rapide.

Corrigé

a) Algorithme

```
-----------------------------------------------------------------------
Nom de l'algorithme méthodes de tri
Rôle : comparer les vitesses d'exécution des algorithmes de tri
Entrée : liste de 100 nombres réels au hasard
Sortie : affichage du classement des méthodes de tri
-----------------------------------------------------------------------

Variables globales : k (entier), debut, fin (réels), Lh (liste)

........................ Fonctions ............................

Fonction tri_select(L: liste) : liste
Variables locales : i,j (entiers)
Début
  Pour i de 0 à longueur(L)-2 Faire
    Pour j de i+1 à longueur(L)-1 Faire
      Si L[j]<L[i] Alors
        L[i],L[j] ← L[j],L[i]          #interversion de L[i] et L[j]
      FinSi
    FinPour
  FinPour
  Retourner(L)
FinFonction

Fonction tri_bulles(L: liste) : liste
Variables locales : i (entier), echange (booléen)
Début
  echange ← VRAI
  TantQue echange == VRAI Faire
    echange ← FAUX
    Pour i de 0 à longueur(L)-2 Faire
      Si L[i]>L[i+1] Alors
        L[i],L[i+1] ← L[i+1],L[i]      #intervertit L[i] et L[i+1]
        echange ← VRAI
      FinSi
    FinPour
  FinPour
  Retourner(L)
FinFonction

Fonction tri_rapide(L: liste) : liste
Variables locales : x, pivot (réels), petits, grands (listes)
Début
  Si longueur(L)<=1 Alors
    Retourner(L)
  FinSi
  pivot ← L.pop()
  petits ← []
  grands ← []
  Pour x dans L Faire
    Si x<=pivot Alors
```

```
    |   Ajouter x à petits
    Sinon
    |   Ajouter x à grands
    FinSi
  Retourner(tri_rapide(petits)+[pivot]+tri_rapide(grands))
FinFonction

........................ Partie principale ............................
Début
  Lh ← []
  Pour k de 0 à 99 Faire          # génère une liste de 100 réels au hasard
  |  Ajouter random.random() à Lh
  FinPour
  debut ← time.time()
  tri_select(Lh)
  fin ← time.time()
  Afficher "Temps d'exécution en secondes avec le tri par sélection : ",fin-debut
  Lh ← []
  Pour k de 0 à 99 Faire
  |  Ajouter random.random() à Lh
  FinPour
  debut ← time.time()
  tri_bulles(Lh)
  fin ← time.time()
  Afficher "Temps d'exécution en secondes avec le tri à bulles : ",fin-debut
  Lh ← []
  Pour k de 0 à 99 Faire
  |  Ajouter random.random() à Lh
  FinPour
  debut ← time.time()
  tri_rapide(Lh)
  fin ← time.time()
  Afficher "Temps d'exécution en secondes avec le tri rapide : ",fin-debut
Fin
```

b) Implémentation Python

```python
######################### Fonctions ############################

def tri_select(L):
    for i in range(0,len(L)-1):
        for j in range(i+1,len(L)):
            if L[j]<L[i]:
                L[i],L[j]=L[j],L[i]
    return(L)

def tri_bulles(L):
    echange = True
    while echange == True:
        echange = False
        for i in range(0,len(L)-1):
            if L[i]>L[i+1]:
                L[i],L[i+1]=L[i+1],L[i]
                echange = True
    return(L)
```

```
def tri_rapide(L):
    if len(L)<=1:
      return(L)
    pivot=L.pop()
    petits=[]
    grands=[]
    for x in L:
        if x<=pivot:
            petits.append(x)
        else:
            grands.append(x)
    return(tri_rapide(petits)+[pivot]+tri_rapide(grands))

#################### Partie principale ##########################

import time
import random

Lh=[]
for k in range(100):
    Lh.append(random.random())
debut=time.time()
tri_select(Lh)
fin=time.time()
print("Temps d'exécution en secondes avec le tri par
sélection : ",fin-debut)

Lh=[]
for k in range(100):
    Lh.append(random.random())
debut=time.time()
tri_bulles(Lh)
fin=time.time()
print("Temps d'exécution en secondes avec le tri à bulles :
",fin-debut)

Lh=[]
for k in range(100):
    Lh.append(random.random())
debut=time.time()
tri_rapide(Lh)
fin=time.time()
print("Temps d'exécution en secondes avec le tri rapide :
",fin-debut)
```

c) Résultats obtenus après exécution

Exemple d'exécution :

- Temps d'exécution en seconde avec le tri par sélection : 0.006999969482421875.
- Temps d'exécution en seconde avec le tri à bulles : 0.006000041961669922.
- Temps d'exécution en seconde avec le tri rapide : 0.0009999275207519531.

Une répétition de plusieurs exécutions du programme montre que le classement de ces méthodes de tri, de la plus rapide à la moins rapide, est : tri rapide, tri à bulles et tri par sélection.

10.8 CRYPTOGRAPHIE

Énoncé

a) Le problème

On considère un texte composé uniquement de caractères minuscules et d'espaces comme *'bonjour tout le monde'* par exemple. On peut crypter ce texte en remplaçant chaque lettre par la lettre obtenue par un décalage circulaire de +3 dans l'alphabet. Par exemple 'a' devient 'd', 'b' devient 'e', … ,'y' devient 'b' et 'z' devient 'c'. Donc le message *'bonjour tout le monde'* devient *'erqmrxu wrxw oh prqgh'*. On parle de codage par la méthode de César avec le décalage 3.

On définit de même le codage par la méthode de César avec le décalage *d* (où *d* est un entier compris entre 0 et 25).

Le but de ce TP est de décrypter le message **'tew wm jegmpi uyi gipe pi gsheki hi giwev'**, codé par la méthode de César, sans connaître le décalage appliqué.

Nous appelerons **message** toute chaîne de caractères ne contenant que des lettres minuscules et des espaces comme 'bonjour tout le monde' par exemple. Toutes les fonctions ci-dessous devront être implémentées en Python.

a) Les questions

1) Écrire une fonction **codage(message,d)** qui à un message **message** et à un entier *d* compris entre 0 et 25, renvoie le message codé par la méthode de César, avec le décalage *d*.
2) Écrire une fonction **freq_lettres(messagecode)** qui à un message **messagecode** (un message codé) retourne la liste des fréquences des lettres de l'alphabet. Par exemple, *freq_lettres(texte)[0]* devra être la fréquence du caractère 'a' dans le message *texte*, et *freq_lettres(texte)[3]*, celle de 'd'.
3) Écrire une fonction **calcul_auto_decal(messagecode)** qui à un message **messagecode** (un message codé), retourne le décalage probable, basé sur le fait que, statistiquement, le caractère le plus présent dans un texte est 'e'.
4) Écrire un algorithme permettant de décrypter le message codé **'tew wm jegmpi uyi gipe pi gsheki hi giwev'**. On pourra remarquer que l'on obtient le décryptage par utilisation de codage(textecode,-d), où *d* est le décalage utilisé pour coder.

Corrigé

a) Algorithme

```
----------------------------------------------------------------------------
Nom de l'algorithme crypage_César
Rôle : décrypter un message codé par la méthode de César
Entrée : ----------------
Sortie : affichage du message décrypté
----------------------------------------------------------------------------
```

```
Variables globales : alphabet, textcod (chaînes), decal (entier)
........................ Fonctions ............................

Fonction codage(message: chaîne ,d: entier) : chaîne
Variables locales : k,i (entiers), messagecode (chaine)
Début
 | messagecode ← ''          # Initialisation : chaîne vide
 | Pour k de 0 à longueur(message)-1 Faire
 | | Si message[k]==' ' Alors    # caractère "espace"
 | | | messagecode ← messagecode+' '
 | | FinSi
 | | Pour i de 0 à 25 Faire
 | | | Si message[k]==alphabet[i] Alors
 | | | | messagecode ← messagecode+alpha[(i+d)%26]
 | | | FinSi
 | | FinPour
 | FinPour
 | Retourner(messagecode)
FinFonction

Fonction freq_lettres(messagecode: chaîne) : liste
Variables locales : k,i (entiers), listefreq (liste)
Début
 | listefreq ← [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
 | Pour k de 0 à longueur(messagecode)-1 Faire
 | | Pour i de 0 à 25 Faire
 | | | Si messagecode[k]==alphabet[i] Alors
 | | | | listefreq[i] ← listefreq[i]+1
 | | | FinSi
 | | FinPour
 | FinPour
 | Retourner(listefreq)
FinFonction

Fonction calcul_auto_decal(messagecode: chaîne) : entier
Variables locales : i, indicemax (entiers), liste_freq (liste)
Début
 | indicemax ← 0
 | liste_freq ← freq_lettres(messagecode)
 | Pour i de 0 à 25 Faire
 | | Si liste_freq[i]>indicemax Alors
 | | | indicemax ← i
 | | FinSi
 | FinPour
 | Si indicemax>=4 Alors
 | | d ← indicemax-4
 | Sinon
 | | d ← indicemax+22
 | FinSi
 | Retourner(d)
FinFonction

........................ Partie principale ............................
```

Début
alphabet ← 'abcdefghijklmnopqrstuvwxyz'
textcod ← 'tew wm jegmpi uyi gipe pi gsheki hi giwev'
decal ← calcul_auto_decal(textcod)
Afficher(codage(textcod,-decal))
Fin

b) Implémentation Python

```
######################### Fonctions ###############################

def codage(message,d):
    messagecode=''
    for k in range(len(message)):
           if message[k]==' ':
               messagecode=messagecode+' '
           for i in range(26):
               if message[k]==alphabet[i]:
                   messagecode=messagecode+alphabet[(i+d)%26]
    return(messagecode)

def freq_lettres(messagecode):
    listefreq=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
    for k in range(len(messagecode)):
       for i in range(26):
          if messagecode[k]==alphabet[i]:
             listefreq[i]=listefreq[i]+1
   return(listefreq)

def calcul_auto_decal(messagecode):
    indicemax=0
    liste_freq=freq_lettres(messagecode)
    for i in range(26):
       if liste_freq[i]>indicemax:
          indicemax=i
    if indicemax>=4:
       d=indicemax-4
    else:
       d=indicemax+22
    return(d)


##################### Partie principale ##########################

alphabet='abcdefghijklmnopqrstuvwxyz'
textcod='tew wm jegmpi uyi gipe pi gsheki hi giwev'
decal=calcul_auto_decal(textcod)
print(codage(textcod,-decal))
```

C) Résultats obtenus après exécution

On obtient le message d'origine : « *pas si facile que cela le codage de cesar* ».

11 L'examen d'algorithmique

PLAN

Examen 1 – Remplissage d'une tirelire
Examen 2 – La suite de Syracuse

OBJECTIFS

➤ Pratiquer, s'entraîner à l'examen oral d'algorithmique.

L'algorithmique appliquée est évaluée en CCF (Contrôle en cours de formation) ou en forme ponctuelle. Il a pour but d'évaluer la capacité du candidat à analyser un énoncé, formaliser une démarche de résolution de problème, écrire, interpréter et éventuellement modifier ou compléter un ou plusieurs algorithmes.

Plus précisément, il s'agit d'évaluer :

- la maîtrise des connaissances d'algorithmique appliquée,
- l'efficacité et la pertinence de la solution proposée,
- la correction et la cohérence de l'utilisation du formalisme retenu,
- la qualité de la mise œuvre, notamment la lisibilité (indentation, commentaires, etc.),
- l'efficacité de l'implémentation,
- la pertinence de l'utilisation des composants logiciels disponibles,
- l'adéquation des tests de validation effectués,
- l'aptitude à proposer des éléments de correction pertinents.

L'examen se déroule en deux temps :

- **Un temps de préparation** qui se décompose en deux parties :
  - Une première partie de 30 minutes (sur table) : une situation problème et des consignes écrites sont données au candidat qui devra fournir une production manuscrite comportant un ou plusieurs algorithmes, susceptible d'être examinée par la commission.

- Une seconde partie de 30 minutes (sur machine) : sur un équipement dédié, mis à disposition par le centre d'examen, le candidat met en œuvre sur machine les algorithmes qu'il a écrits dans la première partie.
- Puis un **temps d'évaluation orale** qui se décompose aussi en deux parties :
  - Le candidat présente d'abord sa solution algorithmique et son implémentation, en une durée maximale de 10 minutes.
  - Ensuite il participe à un entretien d'explicitation conduit par la commission, en une durée maximale de 10 minutes.

Les compétences liées à l'unité « mathématiques » (U21) ne sont pas évaluées dans cette épreuve mais la réalisation demandée peut s'appuyer sur les techniques abordées dans cette unité, les méthodes requises sont alors fournies dans le sujet.

Mode d'emploi des examens proposés

Nous présentons ici deux sujets officiels donnés au CCF d'algorithmique appliquée lors de la session 2014. Nous fournissons aussi les fiches réponse données aux candidats. Ces sujets, entièrement corrigés, ne se prétendent pas exhaustifs, mais constituent tout de même une base de travail solide, en tant qu'exemple de ce qui peut être donné dans ce type d'évaluation.

Examen 1 – Remplissage d'une tirelire

Le problème initial

Je remplis une tirelire de la manière suivante :

- je commence par déposer 1 € dans la tirelire (un lundi),
- puis j'y dépose 2,01 € le jour suivant (un mardi),
- puis 3,02 € le jour suivant (un mercredi),
- puis 4,03 € le jour suivant (un jeudi),
- etc. (chaque jour j'y dépose 1,01 € de plus que la veille).

Par exemple : le huitième jour (un lundi), j'y dépose 8,07 € et le contenu total de la tirelire est : 1 + 2,01 + 3,02 + 4,03 + 5,04 + 6,05 + 7,06 + 8,07 = 36,28 €.

Lorsque le contenu total de la tirelire dépasse 1 500 €, je casse cette tirelire.

Le but du sujet est de déterminer le contenu de la tirelire lorsque je la casserai, ainsi que le jour de la semaine correspondant (lundi, mardi, mercredi, jeudi, vendredi, samedi ou dimanche).

Partie A (sur table) (30 minutes)

1) Compléter l'écriture de la fonction récursive **depot(n)** retournant la somme que j'ai déposée dans la tirelire le *n*-ième jour. Par exemple, on devra avoir depot(1) = 1 et depot(4) = 4.03.

```
Fonction depot(n: type .............)        # type du résultat : ....................
Début
  Si n==......... Alors
    Retourner(.................)
  Sinon
    Retourner (....................................)
  FinSi
FinFonction
```

2) Écrire une fonction **jour(n)** retournant le jour de la semaine correspondant au n-ième dépôt. Par exemple, on devra avoir jour(1)='lundi', jour(2)='mardi' et jour(12)='vendredi'. Cette fonction devra utiliser la liste suivante :

L=['lundi','mardi','mercredi','jeudi','vendredi','samedi','dimanche']

3) Quel est le rôle de l'algorithme suivant ?

```
Variables : k (entier), s, m (réels)
Début
  s ← 0
  Pour k de 1 à 7 Faire
    s ← s + depot(k)
  FinPour
  m ← s/7
  Afficher m
Fin
```

4) Écrire une fonction contenu(n) qui, à un entier $n \geq 1$, renvoie la somme totale en euro, contenue dans la tirelire le n-ième jour. Par exemple on devra avoir contenu(3) = 1 + 2.01 + 3.02 = 6.03.

5) Écrire un algorithme affichant le contenu de la tirelire lorsque je la casserai, ainsi que le jour de la semaine correspondant. Cet algorithme devra faire appel aux fonctions **depot(n)**, **jour(n)** et **contenu(n)**.

Partie B (sur machine) (30 minutes)

1) Implémenter l'algorithme de la partie A en corrigeant les éventuelles erreurs.
2) Exécuter le programme puis noter le jour et le contenu.

Fiche réponse

```
-----------------------------------------------------------------------------
Nom de l'algorithme ..........................................................
Rôle : .......................................................................
Entrée(s) : -------------
Sortie(s) : ..................................................................
-----------------------------------------------------------------------------
Variable(s) globale(s) et type(s) : ..........................................
. . . . . . . . . . . . . . . . . . . . . . . Fonctions . . . . . . . . . . . . . . . . . . . . . . .
Fonction depot(n: type ............)       # type du résultat : ...................
Début
  Si n==......... Alors
```

```
| | Retourner(.................)
| Sinon
| | Retourner (..................................................)
FinFonction

Fonction mois(n: type ............)   # type du résultat : .......................................
Variable(s) locale(s) et type(s) : ..........................
Début
| L ← ['janvier','février','mars','avril','mai','juin','juillet','août',
|                     'septembre','octobre','novembre','décembre']
|
FinFonction

Fonction contenu(n: type ............)        # type du résultat : .............................
Variable(s) locale(s) et type(s) :
............................................................................
Début
|
FinFonction
. . . . . . . . . . . . . . . . . . . . . . . . . .Partie principale . . . . . . . . . . . . . . . . . . . . . . . .
Début
|
Fin
```

Résultats obtenus après exécution : ……..

Rôle de l'algorithme de la question 3 : …...

Solutions

Partie A

Rôle de l'algorithme de la question 3 : calcule puis affiche la moyenne des dépôts effectués la première semaine.

```
-----------------------------------------------------------------------------------------
Nom de l'algorithme : Algorithme_tirelire
Rôle : Détermination du contenu de la tirelire et du jour lorsqu'elle sera cassée
Entrée(s) : ----------------
Sortie(s) : Affichage contenu et jour
-----------------------------------------------------------------------------------------
Variable(s) globale(s) et type(s) :  N (entier)
```

.... Fonctions ....

Fonction depot(n: entier) # type du résultat : réel
Début
  Si n==1 **Alors**
    Retourner(1) # depot(1)=1
  Sinon
    Retourner (depot(n-1)+1.01) # depot(n)=depot(n-1)+1.01
  FinSi
FinFonction

Fonction jour(n: entier) # type du résultat : chaîne de caractères
Variable locale : L (liste)
Début
  L ← ['lundi','mardi','mercredi','jeudi','vendredi','samedi','dimanche']
  n ← (n-1)%7
  Retourner(L[n])
FinFonction

Fonction contenu(n: entier) # type du résultat : réel
Variable locale : s (réel)
Début
  s ← 0 # initialisation : somme s=0
  Pour k de 1 à n **Faire**
    s ← s + depot(n)
  FinPour
  Retourner(s)
FinFonction

..........................Partie principale ....

Début
  N ← 1 # initialisation de N au premier jour
  TantQue (contenu(N)<=1500) **Faire**
    N ← N+1
  FinTantQue
  Afficher "Je casserai la tirelire un ",**jour**(N)," et son contenu sera de ",
  contenu(N)," €."
Fin

Partie B : implémentation en Python et exécution

```
######################### Fonctions #############################
def depot(n) :
    if n==1:
       return(1)
    else :
        return(depot(n-1)+1.01)

def jour(n):
   L=['lundi','mardi','mercredi','jeudi','vendredi',
                              'samedi','dimanche']
   n=(n-1)%7
   return(L[n])
```

```
def contenu(n):
    s=0
    for k in range(1,n+1):
        s=s+depot(k)
    return(s)
#################### Partie principale ########################
N=1
while contenu(N)< 1500:
    N=N+1
print("Je casserai la tirelire un",jour(N),"et son contenu sera
de",contenu(N),"€.")
```

Résultats obtenus après exécution

Je casserai la tirelire un samedi et son contenu sera de 1 554.85 €.

Examen 2 – La suite de Syracuse

Le problème initial

On part d'un nombre entier N supérieur à 1. S'il est pair, on le divise par 2, sinon on le multiplie par 3 puis on ajoute 1. Puis on recommence avec le résultat et ainsi de suite … on s'arrête dès qu'on arrive à 1.

Par exemple : on part du nombre 10.

- 10 est pair donc on le divise par 2 : on obtient 5.
- 5 est impair donc on le multiplie par 3 puis on ajoute 1 : on obtient 16.
- 16 est pair donc on le divise par 2 : on obtient 8.
- 8 est pair donc on le divise par 2 : on obtient 4.
- 4 est pair donc on le divise par 2 : on obtient 2.
- 2 est pair donc on le divise par 2 : on obtient 1.

On a obtenu 1 : On arrête.

En partant du nombre 10, on a donc obtenu successivement 5, 16, 8, 4, 2 puis 1, donc six étapes pour arriver à 1 et le plus grand nombre rencontré est 16.

Autre exemple : si on part de 7, on obtient successivement 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2 puis 1, donc 16 étapes pour arriver à 1 et le plus grand nombre rencontré est 52.

Le but du sujet est de déterminer le plus petit entier de départ N nécessitant 30 étapes pour arriver à 1 en précisant le plus grand nombre alors rencontré.

Partie A (sur table) (30 minutes)

1) Compléter l'écriture de la fonction **suivant**(n) ci-dessous afin qu'elle renvoie le nombre suivant n. Par exemple, on aura suivant(10)=5 et suivant(1)=1.

```
Fonction suivant (n: type ...........) # type du résultat : ..............
Début
  Si n== ....... Alors
    Retourner(.......)
  SinonSi n%2== ...........
    Retourner(.............)
  Sinon
    Retourner(.............)
  FinSi
FinFonction
```

2) Quel est le rôle de l'algorithme suivant ?

```
Variables : n (entier), ch (chaine)
Début
  n ← 10
  ch ← ''                              # chaîne vide
  TantQue suivant(n)!=1 Faire
    ch = ch + suivant(n) + '-'          # concaténation
    n ← suivant(n)
  FinTantQue
  Afficher ch+'1'
Fin
```

3) Écrire une fonction **trajet(n)** qui retourne la liste de tous les nombres rencontrés lorsqu'on part de l'entier n. Par exemple on devra avoir trajet(10)=[5,16,8,4,2,1].
4) Écrire une fonction **max(n)** qui à un nombre de départ n, retourne le plus grand nombre alors rencontré.
5) Écrire un algorithme déterminant puis affichant le plus petit entier N nécessitant 30 étapes pour arriver à 1 ainsi que le plus grand nombre alors rencontré.

Partie B (sur machine) (30 minutes)

1) Implémenter l'algorithme de la partie A en corrigeant les éventuelles erreurs.
2) Exécuter le programme puis noter les résultats obtenus.

Fiche réponse

Nom de l'algorithme ....

Rôle : ....

Entrée(s) : -------------

Sortie(s) : ....

Variable(s) globale(s) et type(s) : ....

. . . . Fonctions . . . .

```
Fonction suivant(n: type .........)        # type du résultat : ..................
Début
  Si n== ....... Alors
    Retourner(.......)
  SinonSi  n%2== ...........
    Retourner(.............)
  Sinon
```

```
  | Retourner(.............)
  FinSi
FinFonction

Fonction trajet(n: type ............)          # type du résultat : .................
Variable(s) locale(s) et type(s) : ...............................................................
Début



FinFonction

Fonction max(n: type ............)          # type du résultat : ..................
Variable(s) locale(s) et type(s) : ..............................................................
Début



FinFonction
. . . . . . . . . . . . . . . . . . . . . . . . . .Partie principale . . . . . . . . . . . . . . . . . . . . . . . .

Début


Fin
```

Résultats obtenus après exécution : ....
Rôle de l'algorithme de la question 2 : ....

Solutions

Partie A

Rôle de l'algorithme de la question 2 : détermine puis affiche les nombres rencontrés séparés par des tirets, si l'on part de l'entier 10 (c'est-à-dire "5-16-8-4-2-1").

```
----------------------------------------------------------------------------------------------
Nom de l'algorithme : Suite de Syracuse
Rôle : Détermine plus petit entier nécessitant 30 étapes et plus grand nb
rencontré
Entrée(s) : ---------
Sortie(s) : Affichage plus petit entier nécessitant 30 étapes et plus grand nb
rencontré
----------------------------------------------------------------------------------------------

Variable(s) globale(s) et type(s) : N (entier)
```

. . . . Fonctions . . . .

```
Fonction suivant(n: entier)          # type du résultat : entier
Début
    Si n==1 Alors
        Retourner(1)
    SinonSi n%2==0 Alors
        retourner(n//2)
    Sinon
        Retourner(3*n+1)
    FinSi
FinFonction

Fonction trajet(n: entier)           # type du résultat : liste
Variable(s) locale(s) et type(s) : L (liste)
Début
    L ← []                           # Initialisation : liste vide
    TantQue suivant(n)!=1 Faire
        Ajouter suivant(n) à L
        n ← suivant(n)               # n devient suivant(n)
    FinTantQue
    Ajouter 1 à L
    Retourner(L)
FinFonction

Fonction max(n: entier)              # type du résultat : entier
Variable(s) locale(s) et type(s) : k, maxi (entiers)
Début
    maxi ← 0                         # Initialisation : maxi=0
    Pour k de 0 à longueur(trajet(n))-1 Faire  # parcours des éléments de trajet(n)
        Si maxi < trajet(n)[k] Alors
            maxi ← trajet(n)[k]      # maxi devient trajet(n)[k] si  maxi < trajet(n)[k]
        FinSi
    Retourner(maxi)
FinFonction
```

. . . . .Partie principale . . . .

```
Début
    N ← 2                  # initialisation de N : plus petit nombre de départ
    TantQue longueur(trajet(N))!=30 Faire
        N ← N+1
    FinTantQue
    Afficher "Le nombre cherché est ",N,". Le plus grand nombre rencontré
      est",max(N),"."
Fin
```

Partie B : implémentation Python et exécution

```
######################### Fonctions ############################
def suivant(n):
    if n==1:
        return(1)
    elif n%2==0:
        return(n//2)
    else :
        return(3*n+1)
```

```
def trajet(n):
    L=[]
    while suivant(n)!=1:
        L.append(suivant(n))
        n=suivant(n)
    L.append(1)
    return(L)

def max(n):
    maxi=0
    for k in range(0,len(trajet(n))):
       if maxi<trajet(n)[k]:
          maxi=trajet(n)[k]
    return(maxi)

#################### Partie principale ########################
N=2
while len(trajet(N))!=30:
     N=N+1
print("Le nombre cherché est",N,". Le plus grand nombre rencontré
est",max(N),".")
```

Résultats obtenus après exécution

Le nombre cherché est 86. Le plus grand nombre rencontré est 196.

Partie 3

Annexe

Exercices supplémentaires

Chapitre 1

1.50 Écrire en base 10 les nombres suivants : $(11001)_2$, $(100100)_2$, $(10001011)_2$, $(101100110)_2$, $(AB)_{16}$, $(B9)_{16}$, $(281)_{16}$ et $(52AD)_{16}$.

1.51 Écrire en base 2 les nombres suivants : 27, 80, 140, 213, $(2D)_{16}$, $(C1)_{16}$, $(3D9)_{16}$ et $(28C)_{16}$.

1.52 Écrire en base 16 les nombres suivants : 136, 539, 1 755, 2 189, $(111001)_2$, $(1010111)_2$, $(11001101)_2$ et $(10001111)_2$.

1.53 On considère l'opération 141 + 93 = 234 écrite en base 10. Écrire en base 2 puis en base 16 cette opération.

1.54 Écrire en base 10 les nombres suivants : $(0{,}11)_2$, $(10{,}1)_2$, $(11{,}011)_2$, $(101{,}01)_2$, $(6{,}C)_{16}$, $(12{,}F)_{16}$, $(A0{,}7)_{16}$ et $(A3{,}B8)_{16}$.

1.55 Écrire en base 2 puis en base 16 les nombres suivants : 7,75 ; 28,5 ; 94,25 ; 100,5625.

1.56 a) On donne $a = (11101)_2$ et $b = (1001)_2$. Calculer $a + b$, $a - b$, ab.

b) Même exercice avec $a = (11000110)_2$ et $b = (1100)_2$.

1.57 a) On donne $a = (B6)_{16}$ et $b = (41)_{16}$. Calculer $a + b$, $a - b$, ab.

b) Même exercice avec $a = (2A5)_{16}$ et $b = (63)_{16}$.

1.58 a) On donne $a = (110110)_2$ et $b = (11000)_2$. Calculer $a + b$, $a - b$, ab et $a{:}b$.

b) Même exercice avec $a = (11010111)_2$ et $b = (1010110)_2$.

1.59 a) On donne $a = (3\,498)_{16}$ et $b = (264)_{16}$. Calculer $a + b$, $a-b$, ab et $a{:}b$.

b) Même exercice avec $a = (B825)_{16}$ et $b = (AD5)_{16}$.

1.58 a) On donne $a = (1001)_2$, $b = (11100)_2$ et $c = (10101)_2$. Calculer $a + b+c$, $a + bc$ et $(a + b)c$.

b) Même exercice avec $a = (3DC)_{16}$, $b = (82)_{16}$ et $c = (2BE)_{16}$.

1.59 Dans chaque cas, donner le quotient q et le reste r de la division euclidienne de a par b, et écrire la division euclidienne en ligne :

a) $a = 80$ et $b = 17$ b) $a = 514$ et $b = 73$ c) $a = 2\,541$ et $b = 87$

1.60 Dans chaque cas, dire si les égalités proposées correspondent à des divisions euclidiennes. Préciser alors les valeurs du quotient et du reste.

a) $114 = 13 \times 8 + 10$ b) $283 = 23 \times 12 + 7$ c) $465 = 19 \times 23 + 28$

1.61 Dans une division euclidienne par 8, quels sont les restes possibles ?

On effectue la division euclidienne d'un entier naturel n par 8. Le quotient est 63. Quelles sont les valeurs possibles de n ?

1.62 Le nombre 1/13 vaut 0,076923076923076923... où l'on observe une répétition périodique de la séquence 076923. Quel est le 1 000^e chiffre après la virgule ? le 11 111^e ?

1.63 On dispose de n données numériques à ranger dans des listes de même taille :
- Si on fait des listes de 7 données, on peut faire q listes et il reste 6 données.
- Si on fait des listes de 11 données, on fait 20 listes de moins mais il reste toujours 6 données.

a) Combien a-t-on de données numériques à classer ?

b) Quelle peut être la taille des listes afin qu'il ne reste aucune donnée ?

1.64 Soit a et b deux entiers naturels. On sait que la division euclidienne de a par 5 donne pour quotient q et pour reste 3, et que la division euclidienne de b par 7 donne le même quotient et un reste égal à 1. Déterminer les valeurs de a et de b sachant que $a + b = 448$.

1.65 Faire la liste des diviseurs de 18, de 26 et de 40.

1.66 Chercher tous les couples $(a;b)$ tels que $a \neq b$ et a est divisible par b dans la liste suivante : 9-15-27-36-45-135.

1.67 Vrai ou faux ? Justifier en donnant un contre-exemple lorsque c'est faux, et en faisant une démonstration lorsque c'est vrai.

a) Si a est divisible par b et par c, alors a est divisible par $b + c$.

b) Si $a + b$ est divisible par c et si a est divisible par c, alors b est divisible par c.

c) Si a divise b, alors a divise b^2.

d) Si a divise b^2, alors a divise b.

1.68 Les nombres suivants sont-ils premiers ? Si oui, dire quel est le dernier diviseur testé, si non, donner un diviseur.

109 – 167 – 183 – 287 – 319 – 431 – 527 – 641 – 667 – 961 – 1009

1.69 Soit n un entier naturel non multiple de 6.

a) Expliquer pourquoi n est de la forme $6k + 1$, $6k + 2$, $6k + 3$, $6k + 4$ ou $6k + 5$, avec $k \in \mathbb{N}$.

b) Montrer que si $n = 6k + 2$, alors n n'est pas un nombre premier. Dans quels autres cas n n'est-il pas un nombre premier ?

c) En déduire que tout nombre premier est de la forme $6k + 1$ ou $6k + 5$, avec $k \in \mathbb{N}$.

d) Inversement, tout entier naturel de la forme $6k + 1$ ou $6k + 5$ est-il un nombre premier ?

1.70 Pour tout n entier naturel, on pose $A_n = n^2 + 7$.

a) Montrer que si n est impair, alors A_n n'est pas un nombre premier.

b) Déterminer le plus petit entier naturel n pair tel que A_n ne soit pas un nombre premier.

1.71 On se propose de chercher s'il existe des nombres premiers de la forme $n^3 - 1$, pour n entier naturel supérieur ou égal à 2, et si possible, d'en faire la liste. On note A_n le nombre $n^3 - 1$.

a) Tester si A_n est premier pour n valant 2, 3, 4, 5 puis 6.

b) Développer $(n - 1)(n^2 + n + 1)$. En déduire que, à part pour $n = 2$, A_n n'est jamais premier.

1.72 Écrire la décomposition en produit de facteurs premiers des nombres suivants :

a) 240 b) 450 c) 810 d) 980

1.73 Écrire la décomposition en produit de facteurs premiers des nombres suivants :

a) 1 134 b) 3 575 c) 2 057 d) 4 123

1.74 Écrire la liste des diviseurs des nombres suivants en utilisant leurs décompositions en produit de facteurs premiers :

a) 54 b) 156 c) 324 d) 1 089

1.75 Écrire la liste des diviseurs des nombres suivants en utilisant leurs décompositions en produit de facteurs premiers :

a) 1 617 b) 5 915 c) 3 127 d) 7 889

1.76 Écrire la décomposition en produit de facteurs premiers des nombres $a = 1\,100$ et $b = 1\,750$. En déduire la décomposition de ab et de a^2.

1.77 On donne les nombres $a = 3^4 \times 11 \times 13^2$ et $b = 3^3 \times 13^2$.

a) Déterminer la décomposition en produit de facteurs premiers de ab.

b) Quel est le quotient de a par b ?

1.78 Décomposer les nombres a et b en produits de facteurs premiers et déterminer leur PGCD dans les cas suivants :

a) $a = 52$ et $b = 117$ b) $a = 9\,900$ et $b = 9\,075$

1.79 Décomposer les nombres a et b en produits de facteurs premiers et déterminer leur PGCD dans les cas suivants :

a) $a = 949$ et $b = 730$ b) $a = 5\,760$ et $b = 7\,200$

1.80 En utilisant la règle PGCD(a;b) = PGCD(b;r) où r est le reste de la division euclidienne de a par b, déterminer le PGCD des nombres a et b dans les cas suivants :
a) $a = 300$ et $b = 228$ b) $a = 4\,774$ et $b = 2\,618$

1.81 En utilisant l'algorithme d'Euclide, chercher PGCD(a;b) dans les cas suivants : a) $a = 1\,120$ et $b = 294$ b) $a = 3\,300$ et $b = 792$

1.82 Vrai ou Faux ?

$93 \equiv 5[8]$ $78 \equiv 4[12]$ $101 \equiv 76[5]$ $115 \equiv 63[26]$

$1\,000 \equiv 100[11]$ $14 \equiv 104[9]$ $48 \equiv 132[7]$ $204 \equiv 88[15]$

1.83 Compléter les congruences avec un entier naturel, le plus petit possible :

$108 \equiv ......[5]$ $141 \equiv ......[9]$ $200 \equiv ......[13]$ $112 \equiv ......[4]$

1.84 a et b sont deux entiers naturels tels que $a \equiv 4[9]$ et $b \equiv 7[9]$ et $S = \{0;1;2;3;4;5;6;7;8\}$.

1) Compléter les congruences suivantes, en mettant un nombre de l'ensemble S :

$a + b \equiv ......[9]$ $a \times b \equiv ......[9]$ $5a + 3b \equiv ......[9]$

$a^2 \equiv ......[9]$ $b^3 \equiv ......[9]$ $a^2b^3 \equiv ......[9]$

2) Donner un nombre a et un nombre b, supérieurs à 50, tels que $a \equiv 4[9]$ et $b \equiv 7[9]$.

1.85 Soit x un entier naturel tel que $x \equiv 4[7]$.

a) Compléter les congruences suivantes en mettant un nombre de l'ensemble $S = \{0;1;2;3;4;5;6\}$.

$x^2 \equiv ....[7]$ $x^3 \equiv ....[7]$ $x^4 \equiv ....[7]$ $x^5 \equiv ....[7]$ $x^6 \equiv ....[7]$

b) Expliquer pourquoi $3\,000 \equiv 4[7]$.

c) Montrer que $3\,000^{1515} = (3\,000^6)^{252} \times 3\,000^3$.

d) En utilisant les résultats de la question a, déterminer le reste de la division euclidienne de 3000^{1515} par 7.

1.86 On considère l'équation $2x \equiv 9[21]$.

a) Déterminer un nombre entier a tel que $2 \times a \equiv 1[21]$.

b) Résoudre l'équation.

c) Citer toutes les solutions de l'équation qui sont comprises entre 100 et 200.

1.87 On considère l'équation $5x + 6 \equiv 0[26]$.

a) Vérifier que $5 \times 21 \equiv 1[26]$.

b) Résoudre l'équation.

c) Citer toutes les solutions de l'équation qui sont comprises entre 200 et 300.

1.88 À chaque lettre de l'alphabet, on associe un nombre entier entre 0 et 25 selon la table de correspondance proposée par le tableau 1.16.

Tableau 1.16

Lettre	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Nombre	0	1	2	3	4	5	6	7	8	9	10	11	12
Lettre	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Nombre	13	14	15	16	17	18	19	20	21	22	23	24	25

Pour crypter (ou coder) un message, on crypte les lettres une par une selon le procédé suivant :

- On repère le nombre x associé à la lettre, dans le tableau 1.16.
- On calcule le nombre $ax + b$ (les nombres a et b sont les clés de cryptage).
- On détermine le reste y de la division euclidienne de $ax + b$ par 26.
- On repère la lettre associée au nombre y dans le tableau : c'est la lettre cryptée.
- Ainsi, on a $y \equiv ax + b[26]$. Ce type de cryptage est un codage affine.

Par exemple, pour crypter la lettre F avec les clés $a = 9$ et $b = 2$:

- Le nombre x associé à la lettre F est 5.
- On calcule $9 \times 5 + 2$, ce qui donne 47.
- On détermine le reste de la division euclidienne de 47 par 26 : on trouve 21.
- On repère la lettre associée à 21 dans le tableau 1.16 : c'est V.

Ainsi, avec les clés $a = 9$ et $b = 2$, F est cryptée par V.

Partie A – Cryptage d'un mot avec les clés $a = 9$ et $b = 2$

a) Déterminer quelle est la lettre qui crypte la lettre S en détaillant toutes les étapes du processus de cryptage.

b) Crypter l'expression « BTS SIO ».

Partie B – Décryptage avec les clés $a = 9$ et $b = 2$

a) Trouver l'entier c compris entre 0 et 25 tel que $9 \times c \equiv 1[26]$.

b) Montrer que $y \equiv 9x + 2[26] \Rightarrow x \equiv 3y + 20[26]$.

c) Décrypter le mot « UZKHR Y ».

Partie C – Étude d'autres clés de cryptage

a) Que pensez-vous d'un cryptage utilisant les clés $a = 1$ et $b = 0$?

b) On prend maintenant les clés $a = 4$ et $b = 0$.

Comment est cryptée la lettre E ? la lettre R ? Que pensez-vous de ce cryptage ?

c) Pour qu'un cryptage affine soit considéré comme acceptable, il faut qu'il soit bijectif.

Il est possible de prouver que des bonnes clés de cryptage sont deux entiers naturels inférieurs ou égaux à 25 avec la clé a qui est un nombre premier avec 26.

Faire la décomposition de 26 en produit de facteurs premiers. En déduire la liste des valeurs possibles de a.

Combien peut-on trouver de couples $(a;b)$ permettant un bon cryptage ?

1.89 Tous les codes qui substituent une lettre à une autre lettre peuvent facilement être déchiffrés par une analyse fréquentielle. Pour éviter cela et rendre le code plus sur, on peut coder les mots par blocs de deux lettres. À chaque bloc de deux lettres, appelé bigramme, on associe un entier entre 0 et 675 à l'aide du tableau 1.17.

La première lettre du bigramme est écrite dans la première ligne, la deuxième dans la première colonne.

Tableau 1.17

	A	B	C	D	E	F	...
A	0	1	2	3	4	5	
B	26	27	28	29	30	31	
C	52	53	54	55	56	57	
...							

Par exemple, le bigramme BC correspond à 53 et CB correspond à 28.

a) À quel entier correspond le bigramme DF ? le bigramme KG ? le bigramme SP ?

b) Quel bigramme correspond à 440 ? à 115 ? à 307 ?

c) Des clés a et b étant données, on utilise la fonction de codage $y \equiv ax + b[676]$, dans laquelle x est le nombre correspondant au bigramme à coder, y étant la valeur correspondant au bigramme codé.

Dans cet exercice, on prendra les clés $a = 21$ et $b = 8$ et, par conséquent, la fonction de codage $y \equiv 21x + 8[676]$. Coder le mot BIGRAMME.

d) Vérifier que $21 \times 161 \equiv 1[676]$. En déduire la fonction de décodage. Décoder le message BOUUQY.

1.90 Soit S l'ensemble des entiers naturels inférieurs ou égaux à 10.

a) Déterminer les entiers naturels a et b appartenant à S tels que $\begin{cases} 2a+b \equiv 0[11] \\ 7a+b \equiv 4[11] \end{cases}$.

b) Même question avec le système $\begin{cases} 3a+2b \equiv 9[11] \\ 5a+b \equiv 6[11] \end{cases}$.

CHAPITRE 2

2.33 Calculer les quatre premiers termes de la suite (u_n) définie pour tout entier $n \geq 0$ par $u_n = n + 1 - 1/(n + 1)$.

2.34 Calculer les trois premiers termes des suites (v_n) définies par $v_1 = 4$ et $v_{n+1} = v_n^2 + n$.

2.35 Soit (u_n) la suite définie par $u_0 = 3$ et pour tout entier $n \geq 0$ par $u_{n+1} = -3u_n + 8$. Pour tout $n \in \mathbb{N}$, on pose $S_n = u_0 + u_1 + \ldots + u_n$.

On souhaite déterminer, à l'aide d'un tableur, quelques valeurs particulières de u_n et de S_n. On adopte pour cela la disposition la disposition de la fig. 2.13.

	A	B	C
1	n	Un	Sn
2	0		
3	1		
4	2		
5	3		
6	4		

Figure 2.13

a) Que doit-on saisir dans la cellule B2 ?

b) Quelle formule doit-on saisir dans la cellule B3, avant de l'étirer vers le bas ?

c) Que peut-on saisir dans la cellule C2 ?

d) Quelle formule peut-on saisir dans la cellule C3, avant de l'étirer vers le bas ?

e) Déterminer, à l'aide du tableur, les valeurs de u_{15} et S_{10}.

2.36 Soient (u_n), (v_n) et (w_n) les suites définies pour tout entier naturel n par $u_n = (3n+2)/2$, $v_n = 2/10^n$ et $w_n = -4n$. Pour chacune de ces suites, dire si elle est arithmétique, géométrique ou ni arithmétique et ni géométrique.

2.37 Dans chacun des cas suivants, déterminer u_k.

a) (u_n) est une suite arithmétique de raison 3 et on a $u_0 = 2$ et $k = 8$.

b) (u_n) est une suite arithmétique et on a $u_0 = -5$, $u_2 = -17$ et $k = 7$.

2.38 Dans chacun des cas suivants, déterminer u_k.

a) (u_n) est une suite géométrique de raison 4 et on a $u_0 = 2{,}5$ et $k = 5$.

b) (u_n) est une suite géométrique de raison 0,2 et on a $u_2 = 8$ et $k = 6$.

2.39 Le nombre 22 peut-il être le premier terme d'une suite arithmétique de raison -4 dont un des termes est 50 ?

2.40 On considère une suite (u_n) telle que $u_0 = 2$, $u_1 = 8$ et $u_2 = 32$. Donner la valeur de vérité des propositions suivantes (justifier) :

- P = « Il est possible que la suite (u_n) soit arithmétique »
- Q = « Il est possible que la suite (u_n) soit géométrique »
- S = « La suite (u_n) est obligatoirement arithmétique »
- T = « La suite (u_n) est obligatoirement géométrique »

2.41 Soit (u_n) la suite arithmétique de premier terme $u_0 = -20$ et de raison 2.

Pour tout entier naturel n, on pose $S_n = u_0 + u_1 + \ldots + u_n$.

a) Déterminer u_1 et S_1.

b) Donner l'expression de u_n en fonction de n.

c) Existe-il un entier naturel n tel que $u_n = 11$?

d) Quel est le plus petit entier n tel que $u_n > 555{,}5$?

e) Calculer S_{30}.

f) Déterminer, à l'aide d'un tableur, le plus petit entier n tel que $S_n \geq 1\,200$.

2.42 Soit (u_n) la suite géométrique de premier terme $u_0 = 4$ et de raison 0,4.

a) Déterminer u_1.

b) Donner l'expression de u_n en fonction de n.

c) Déterminer l'entier naturel n tel que $u_n = 0{,}1024$.

d) Calculer $u_0 + u_1 + \ldots + u_7$ (arrondir à 10^{-3} près).

2.43 Calculer les sommes suivantes :

- $S = 1 + 3 + 3^2 + \ldots + 3^{15}$
- $T = 10 + 20 + 30 + \ldots + 410 + 420$

2.44 Calculer les sommes suivantes :

- $U = 1 + 3 + 5 + 7 + \ldots + 149 + 151$
- $V = v_0 + v_1 + v_2 + \ldots + v_{12}$ où (v_n) est une suite géométrique de raison 2,5 telle que $v_2 = 0{,}25$.

2.45 Soit (w_n) la suite définie par $w_0 = 500$ et pour tout entier naturel n par $w_{n+1} - w_n + 52 = 0$.

a) Déterminer w_1 et w_2.

b) Quelle est la nature de la suite (w_n) ?

c) En déduire l'expression de w_n en fonction de n.

d) Calculer w_{11}.

e) Quel est le plus petit entier naturel n tel que $w_n < 0$?

f) Calculer $w_0 + w_1 + w_2 + \ldots + w_{16}$.

2.46 Soit (u_n) la suite définie par $u_0 = 4$ et pour tout entier naturel n par $u_{n+1} = u_n/2$.

1) a) Quelle est la nature de cette suite ?

b) En déduire l'expression de u_n en fonction de n.

2) Pour tout $n \in \mathbb{N}$, on pose $S_n = u_0 + u_1 + \ldots + u_n$.

a) Déterminer une expression simplifiée de S_n en fonction de n.

b) En déduire l'arrondi à 10^{-3} près de S_8.

c) Vrai ou faux ? (Justifier) : « il existe un entier naturel n tel que $S_n \geq 10$ ».

2.47 Donner le sens de variations de chacune des suites (u_n), (v_n) et (w_n) définies pour tout $n \in \mathbb{N}$ par : $u_n = -10 + 15n$, $v_n = 3^n / 300$, $w_n = 0{,}98^{-n} + 2$.

2.48 Soit (u_n) la suite définie par $u_n = u_n = 5 \times 2^n$.

a) Déterminer le plus petit entier naturel n tel que $u_n > 10\,000$.

b) Écrire un algorithme permettant de déterminer le plus petit entier n tel que $u_n > 10^{20}$.

2.49 Soit (u_n) la suite définie par $u_n = 0{,}5^{n-4}$.

a) Déterminer le plus petit entier naturel n tel que $u_n < 0{,}01$.

b) Écrire un algorithme permettant de déterminer le plus petit entier n tel que $u_n < 10^{-8}$.

2.50 Déterminer les limites suivantes :

$$A = \lim_{n \to +\infty} 0{,}99^n \qquad B = \lim_{n \to +\infty} \left(1000 \times 0{,}07^n\right) \qquad C = \lim_{n \to -\infty} (-0{,}01 \times 10^n)$$

$$D = \lim_{n \to +\infty} \frac{4^n}{2^n \times 12} \qquad E = \lim_{n \to +\infty} \left(2 - \sqrt{3}\right)^n$$

2.51 Déterminer les limites suivantes :

$$A = \lim_{n \to +\infty} \left(7 - 0{,}25^n\right) \qquad B = \lim_{n \to +\infty} \left(1 + 0{,}97^n\right) \qquad C = \lim_{n \to +\infty} (3 + 11 \times 0{,}8^n)$$

$$D = \lim_{n \to +\infty} \left(14 - \frac{2}{9^n}\right) \qquad E = \lim_{n \to +\infty} \left(8 + 4^{-3n}\right)$$

2.52 Deux entreprises de services informatiques Sio1 et Sio2, viennent d'être créées début 2014. Les chiffres d'affaires obtenus en 2014 par Sio1 et Sio2 sont respectivement 80 000 € et 90 000 €. Les gérants de ces entreprises prévoient des hausses constantes de leurs chiffres d'affaires : + 5 % par an pour Sio1 et + 4 % par an pour Sio2. On note respectivement u_n et v_n les chiffres d'affaires, en euro, de l'année 2014 + n des entreprises Sio1 et Sio2.

1) Déterminer u_1 et v_1 puis interpréter les résultats obtenus.

2) a) Montrer que les suites (u_n) et (v_n) sont géométriques et déterminer leurs raisons.

b) En déduire l'expression de u_n en fonction de n et celle de v_n en fonction de n.

c) Quels sont les chiffres d'affaires prévisibles de ces entreprises en 2020 ? (arrondir à l'euro près)

3) Pour tout entier naturel n, on pose $w_n = u_n / v_n$.

a) Vérifier que $w_0 < 1$.

b) Déterminer le plus petit entier naturel n tel que $w_n > 1$.

c) À partir de quelle année le chiffre d'affaires de Sio1 dépassera-t-il celui de Sio2 ?

2.53 Soit (u_n) la suite définie par $u_0 = 3$ et pour tout entier naturel n par $u_{n+1} = 0{,}5u_n - 2$.

1) a) Déterminer u_1 et u_2.

b) Montrer que (u_n) n'est ni arithmétique, ni géométrique.

c) À partir d'une table de valeur éditée sur une calculatrice graphique, émettre une conjecture sur la limite de la suite (u_n).

2) Soit (v_n) la suite définie pour tout entier naturel n par $v_n = u_n + 4$.

a) Calculer les trois premiers termes de la suite (v_n).

b) Montrer que (v_n) est une suite géométrique de raison 0,5.

c) Exprimer v_n en fonction de n. En déduire l'expression de u_n en fonction de n.

d) Déterminer $\lim\limits_{n \to +\infty} u_n$.

2.54 Un ordinateur est infecté par un virus. Le nombre u_n de fichiers infectés après n allumages vérifie $u_1 = 1$ et $u_{n+1} = 3u_n + 1$.

1) a) Quel est le nombre de fichiers infectés après trois allumages ?

b) Écrire un premier algorithme donnant le nombre de fichiers infectés après 20 allumages. Implémenter cet algorithme sur calculatrice ou en Python.

c) Écrire un second algorithme déterminant à partir de combien d'allumages le nombre de fichiers infectés dépassera 50 000. Implémenter cet algorithme sur calculatrice ou en Python.

2) Pour tout entier naturel $n \geq 1$, on pose $v_n = u_n + 0{,}5$.

a) Calculer v_1, v_2 et v_3. Quelle conjecture peut-on émettre sur la nature de la suite (v_n) ?

b) Montrer que pour tout entier $n \geq 1$, on a $v_{n+1} = 3v_n$.

c) En déduire l'expression de v_n puis de u_n en fonction de n.

d) Retrouver à l'aide de l'expression de u_n le résultat de la question 1c).

2.55 La loi de Moore affirme que depuis 1975, le nombre de transistors par microprocesseur proposés à la vente au grand public, double tous dix-huit mois. En 1975, un tel microprocesseur comportait 9 000 transistors.

1) Justifier que le nombre de transistors des microprocesseurs quadruple tous les trois ans.

2) On note u_n le nombre de transistors (en milliers) d'un microprocesseur fabriqué lors de l'année $1975 + 4n$.

a) Justifier que $u_0 = 9$.

b) Calculer u_1 et u_2 puis interpréter ces nombres.

c) Montrer que (u_n) est une suite géométrique, puis donner l'expression de u_n en fonction de n.

d) Déterminer le nombre de transistors dans un microprocesseur fabriqué en 1991.

e) En imaginant que cette évolution se poursuive ainsi, à partir de quelle année le nombre de transistors par microprocesseur devrait-il dépasser les 150 milliards ?

2.56 Le but de cet exercice est d'étudier la dépréciation d'un modèle d'ordinateur en fonction du temps écoulé, exprimé en trimestre, depuis sa mise sur le marché. L'entreprise conceptrice de ce modèle souhaite déterminer l'évolution trimestrielle du prix de vente de cet ordinateur, exprimé en euro. On appelle n le nombre de trimestres écoulés depuis la mise sur le marché de ce produit. Ainsi, à la mise sur le marché, on a $n = 0$. Deux modélisations ont été retenues par cette entreprise :

A) Première modélisation : le prix de vente initial à la mise sur le marché de ce modèle d'ordinateur est de 795 €. Chaque trimestre, le prix de vente de ce modèle diminue de 10 % en raison des progrès technologiques.

On note (u_n) la suite telle que, pour tout entier naturel n, u_n désigne le prix de vente, exprimé en euro, de ce modèle d'ordinateur, n trimestres après sa mise sur le marché.

1. Donner u_0 puis calculer u_1 et u_2.
2. Déterminer la nature de la suite (u_n) et préciser sa raison.
3. En déduire que, pour tout entier n, on a $u_n = 795 \times 0{,}9^n$.
4. À partir de combien de trimestres le prix de vente d'un tel ordinateur devient-il strictement inférieur à 300 € ?

B) Deuxième modélisation : le prix de vente, exprimé en euro, de ce modèle d'ordinateur au bout de n trimestres écoulés depuis sa mise sur le marché, noté v_n est donné par $v_n = 525 \times 3^{-0{,}25n} + 270$.

1. Vérifier que le prix de vente de ce modèle d'ordinateur à sa mise sur le marché est de 795 €.
2. Déterminer le nombre minimal de trimestres écoulés depuis sa mise sur le marché à partir duquel le prix de vente de ce modèle d'ordinateur deviendra inférieur ou égal à 300 €.

C) Comparaison des deux modèles :

1. Déterminer les prix de vente, dans chacune des modélisations, cinq trimestres après la mise sur le marché du modèle d'ordinateur.

2. À long terme, laquelle des deux modélisations donne le prix de vente le plus bas ? Justifier la réponse.

CHAPITRE 3

3.38 On considère les matrices $A=(a_{i,j})$, $B=(b_{i,j})$ et $C=(c_{i,j})$, carrées d'ordre 4, définies par $a_{i,j}=i^2$, $b_{i,j}=i+j$ et $c_{i,j}=Min(i;j)$. Écrire ces matrices sous la forme de tableaux de nombres.

3.39 On considère la matrice $A=\begin{pmatrix}5 & -1\\ 2 & -1\end{pmatrix}$.

a) Déterminer $A+2A$ et $-10A$.

b) Est-il possible de trouver un nombre réel x tel que $xA+I_2=0$? (Justifier)

3.40 Soit A la matrice définie par $A=\begin{pmatrix}3 & 2\\ 5 & 10\end{pmatrix}$. Déterminer le plus petit entier naturel n tel que tous les éléments de $n\times A$ soient strictement supérieurs à 10^3.

3.41 Soient $A=\begin{pmatrix}3 & 1 & 2\\ 1 & 3 & -1\end{pmatrix}$ et $B=\begin{pmatrix}0 & 2 & 1\\ -2 & 1 & 3\end{pmatrix}$.

a) Calculer $A-B$ et $2A-5B$.

b) Déterminer les deux matrices C et D telles que $C-D=3B$ et $C+D=2A$.

3.42 Une entreprise emploie trois catégories de salariés (ouvriers, commerciaux et administratifs) dans deux annexes (annexe 1 et annexe 2). On associe la matrice A suivante à la répartition de ces salariés :

$$A=\begin{pmatrix}56 & 3 & 7\\ 25 & 2 & 5\end{pmatrix}$$

Par exemple, on compte trois commerciaux dans l'annexe 1.

a) Donner, pour chaque annexe le nombre de salariés de chaque catégorie.

b) On peut obtenir, à partir de deux matrices B et C, et du produit $B\times A\times C$, le nombre total de salariés dans l'ensemble des deux annexes. De quelles matrices s'agit-il ?

3.43 La matrice A suivante donne les notes de trois étudiants que l'on appellera E1, E2 et E3 (les notes de E_i sont sur la $i^{\text{ème}}$ ligne) dans quatre disciplines : mathématiques, algorithmique, français et droit (dans cet ordre) :

$$A=\begin{pmatrix}8 & 10 & 14 & 12\\ 12 & 6 & 12 & 18\\ 14 & 11 & 10 & 10\end{pmatrix}$$

Par exemple, E1 a 8 en mathématiques et 14 en français.

a) Quelles sont les notes de E2 dans chacune des disciplines ?

Les coefficients de ces disciplines sont : 2 pour les mathématiques, 1 pour l'algorithmique, 3 pour le français et 5 pour le droit.

b) On peut obtenir, à partir d'une matrice A, et du produit $A \times B$, la moyenne de chaque étudiant. De quelle matrice s'agit-il ? Effectuer le calcul et donner la matrice C obtenue.

c) Donner la matrice M telle que $M \times C$ représente la moyenne de ce groupe de trois étudiants.

3.44 On considère les matrices $A = \begin{pmatrix} 2 & 0 \\ 3 & -1 \end{pmatrix}$, $B = \begin{pmatrix} 2 \\ -1 \end{pmatrix}$ et $C = \begin{pmatrix} 0 & -2 & 5 \\ 2 & 1 & 3 \end{pmatrix}$.

Faire tous les produits possibles de deux matrices.

3.45 On considère les matrices $A = \begin{pmatrix} 4 \\ -2 \end{pmatrix}$, $B = \begin{pmatrix} 2 & 1 & 1 \\ -3 & 1 & 0 \\ 0 & 7 & 2 \end{pmatrix}$ et $C = \begin{pmatrix} 2 & 1 \\ 1 & -2 \\ 3 & 0 \end{pmatrix}$.

Faire tous les produits possibles de deux matrices.

3.46 Dans cet exercice, A, B et C désignent des matrices carrées d'ordre 2. Vrai ou faux ? (Justifier)

a) $\forall A$, $\forall B$, $A \times B = B \times A$

b) $\exists A$, $\exists B$, $A \times B = B \times A$

c) $\exists A$, $A^2 = O$

d) $\exists A$, $A^2 = I_2$

e) $\forall A$, $\exists B$, $A \times B = I_2$

3.47 On considère les matrices $A = \begin{pmatrix} 3 & 1 \\ -1 & 6 \end{pmatrix}$ et $B = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 3 & -1 \end{pmatrix}$.

a) Calculer A^2 puis A^3.

b) Calculer B^2 puis B^3.

3.48 Soit $B = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 2 & 3 & 0 \end{pmatrix}$.

a) Calculer B^2 et B^3.

b) Montrer que $B^4 = 0$ (matrice nulle) puis que, pour tout $n \geq 3$, $B^n = 0$.

3.49 Une entreprise emploie trois catégories de personnels A, B et C répartis en trois services. En voici la répartition :

Tableau 3.4

	A	B	C
Service 1	4	6	5
Service 2	3	8	5
Service 3	1	3	10

Dans cette entreprise, les salaires mensuels nets et les primes de fin d'années sont données par le tableau suivant :

	A	B	C
Salaire mensuel (en milliers d'euros)	2,5	2	1,6
Prime (en milliers d'euros)	1	1	0,8

On considère les matrices $A = \begin{pmatrix} 4 & 6 & 5 \\ 3 & 8 & 5 \\ 1 & 3 & 10 \end{pmatrix}$, $B = \begin{pmatrix} 2{,}5 & 1 \\ 2 & 1 \\ 1{,}6 & 0{,}8 \end{pmatrix}$ et $J = \begin{pmatrix} \frac{1}{15} & 0 & 0 \\ 0 & \frac{1}{16} & 0 \\ 0 & 0 & \frac{1}{14} \end{pmatrix}$.

a) Calculer $A \times B$ et interpréter ce que représentent les colonnes de la matrice obtenue.

b) Calculer $J \times A \times B$ et interpréter ce que représentent les colonnes de la matrice obtenue.

3.50 On considère les matrices $A = \begin{pmatrix} 1 & 2 \\ 3 & 1 \end{pmatrix}$ et $B = \begin{pmatrix} -0{,}2 & 0{,}4 \\ 0{,}6 & -0{,}2 \end{pmatrix}$.

Calculer $A \times B$ et $B \times A$. Que peut on dire des matrices A et B ?

3.51 On considère les matrices $A = \begin{pmatrix} 1 & 0 & 2 \\ 0 & -1 & 2 \\ 2 & 4 & 1 \end{pmatrix}$ et $A' = \begin{pmatrix} 1{,}8 & -1{,}6 & -0{,}4 \\ -0{,}8 & 0{,}6 & 0{,}4 \\ -0{,}4 & 0{,}8 & 0{,}2 \end{pmatrix}$.

a) Montrer que A' est l'inverse de A.

b) En déduire l'inverse des matrices suivantes :

$$\begin{pmatrix} 10 & 0 & 20 \\ 0 & -10 & 20 \\ 20 & 40 & 10 \end{pmatrix} \qquad \begin{pmatrix} 18 & -16 & -4 \\ -8 & 6 & 4 \\ -4 & 8 & 2 \end{pmatrix}$$

3.52 On considère les matrices $A = \begin{pmatrix} 1 & 2 & 0 \\ -1 & 2 & 0 \\ 1 & 1 & 3 \end{pmatrix}$ et $B = \begin{pmatrix} 6 & -6 & 1 \\ 3 & 3 & 0 \\ -3 & 1 & 4 \end{pmatrix}$.

Calculer $A \times B$. Quelle est la matrice inverse de A ?

3.53 On considère les matrices $A = \begin{pmatrix} 2 & 0 & 4 \\ 1 & 1 & 1 \\ 3 & 4 & 5 \end{pmatrix}$ et $B = \begin{pmatrix} 3 & -48 & -12 \\ -6 & -6 & 6 \\ 3 & -24 & 6 \end{pmatrix}$.

a) Calculer $A \times B$.

b) Quelle est la matrice inverse de A ?

c) Quelle est la matrice inverse de $0,25A$?

3.54 On considère les matrices $A = \begin{pmatrix} 5 & 4 \\ 11 & 9 \end{pmatrix}$ et $B = \begin{pmatrix} 9 & -4 \\ -11 & 5 \end{pmatrix}$.

a) Calculer $B \times A$. Que peut-on en déduire ?

b) Peut-on en déduire $A \times B$ sans aucun calcul ?

3.55 On considère les matrices $A = \begin{pmatrix} 10 & 6 \\ 4 & 2 \end{pmatrix}$ et $B = \frac{1}{2}\begin{pmatrix} -1 & 3 \\ 2 & a \end{pmatrix}$.

a) Calculer le produit $A \times B$.

b) Déterminer alors la valeur de a pour que B soit la matrice inverse de A.

3.56 Soit $A = \begin{pmatrix} 4 & 2 \\ 3 & 6 \end{pmatrix}$.

a) Calculer $10A - A^2$.

b) Montrer que A est inversible et déterminer A^{-1}.

3.57 Soit $J = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$.

a) Calculer J^2. En déduire J^{-1}.

b) Calculer $I_2 + J + J^2 + J^3$.

c) Calculer J^{2014}.

3.58 a) Écrire le système $\begin{cases} 3x - 5y = 21 \\ 5x - 7y = 31 \end{cases}$ sous la forme matricielle $AX = B$.

Déterminer A^{-1} à la calculatrice et résoudre le système.

b) Résoudre le système $\begin{cases} 2x + y = 3 \\ 5x + 2y = 10 \end{cases}$ à l'aide de matrices.

c) Résoudre le système $\begin{cases} x + y = 3 \\ 3x + y = 5 \\ x - 2z = 7 \end{cases}$ à l'aide de matrices.

3.59 Résoudre chaque système à l'aide des matrices :

a) $\begin{cases} 3x + y + z = 9 \\ x + 3y + z = 29 \\ x + y + 3z = -7 \end{cases}$ b) $\begin{cases} 3x + 2y + z = 14 \\ 2x + 5y + z = 3 \\ x + 6y - z = -10 \end{cases}$ c) $\begin{cases} x + y = -4 \\ -2y + 2z = -8 \\ 3x + y + 3z = -21 \end{cases}$

3.60 Une facture donne :

```
*****************************************************
*              matériel              *  Total TTC   *
*****************************************************
*  3 USB 4Go + 2 USB 8Go + 4 USB 16Go *   95.70 €   *
*  3 USB 4Go + 2 USB 8Go + 4 USB 16Go *   83.00 €   *
*  3 USB 4Go + 2 USB 8Go + 4 USB 16Go *  101.80 €   *
*****************************************************
```

Retrouver le prix d'une clé USB 4 Go, d'une clé USB 8 Go et d'une clé USB 16 Go.

3.61 Un nombre entier de trois chiffres est tel que :

- La somme de ces chiffres est 7.
- La somme du nombre formé par ces deux premiers chiffres et de son dernier chiffre est 25.
- La somme de son premier chiffre est du nombre formé par ces deux derniers chiffres est 16.

Quel est ce nombre ?

3.62 On appelle matrice booléenne toute matrice dont les éléments sont des booléens (0 ou 1). Dans cet exercice toute matrice sera booléenne et carrée d'ordre 2.

On pose $A=\begin{pmatrix}1 & 1\\ 1 & 1\end{pmatrix}$, $B=\begin{pmatrix}1 & 1\\ 1 & 0\end{pmatrix}$ et $C=\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$.

1) a) Calculer $A+A$, $B+B$ et $C+C$.

b) Si M est une matrice booléenne, que vaut $M+M$?

2) Calculer A^2, B^2 et C^2.

3) Si M une matrice booléenne. On note $\overline{M}$ la matrice booléenne obtenue en transformant chaque élément m_{ij} de M par $\overline{m_{ij}}$.

a) Que vaut $\overline{B}$? Que vaut $\overline{C}$?

b) Si M est une matrice booléenne, que vaut $M+\overline{M}$?

3.63 Nous appellerons matrice de chaînes de caractères, toute matrice dont les éléments sont des chaînes de caractères.

Par exemple, $\begin{pmatrix}'hh' & 'sio'\\ 'd55' & ''\end{pmatrix}$ est une matrice de chaînes de caractères.

On définit la somme de deux chaînes de caractères par la concaténation.
Par exemple : 'az'+'erty'='azerty'.

On définit le produit d'un entier naturel par une chaîne de caractère de la manière suivante : 0×**ch**='' (chaîne vide); 1×**ch**=**ch**; 2×**ch**=**ch**+**ch**; 3×**ch**=**ch**+**ch**+**ch**; etc. (où **ch** est une chaîne de caractères).

Si k est un entier naturel, si A et B sont des matrices de chaînes de caractères, et si N est une matrice dont les éléments sont des entiers naturels, on peut alors définir les matrices $A+B$, kA et $N\times A$.

Par exemple, on a :

$$\begin{pmatrix}'abc' & 'ze'\\ 't' & 'hh'\end{pmatrix}+\begin{pmatrix}'55' & ''\\ 'eee' & 'bk'\end{pmatrix}=\begin{pmatrix}'abc55' & 'ze'\\ 'teee' & 'hhbk'\end{pmatrix}$$

$$3\times\begin{pmatrix}'dd' & ''\\ 'abc' & 'k'\end{pmatrix}=\begin{pmatrix}'dddddd' & ''\\ 'abcabcabc' & 'kkk'\end{pmatrix}$$

$$\begin{pmatrix}3 & 0\\ 1 & 2\end{pmatrix}\times\begin{pmatrix}'a' & 'bc'\\ 'efg' & 'z'\end{pmatrix}=\begin{pmatrix}'aaa' & 'bcbcbc'\\ 'aefgefg' & 'bczz'\end{pmatrix}$$

On pose $A=\begin{pmatrix}'a' & 'b'\\ 'c' & 'd'\end{pmatrix}$, $B=\begin{pmatrix}'' & 'e'\\ 'f' & 'g'\end{pmatrix}$, $N=\begin{pmatrix}2 & 1\\ 0 & 4\end{pmatrix}$ et $J=\begin{pmatrix}'x' & ''\\ '' & 'x'\end{pmatrix}$.

a) Déterminer $A+A$, $3A$ et $0A$.

b) Déterminer $I_2 \times A$, $N \times A$ et $N \times B$

c) Déterminer $N \times J$.

d) On pose $M = \begin{vmatrix} m & n \\ p & q \end{vmatrix}$, où m, n, p et q sont des entiers naturels. Quels sont les éléments de $M \times J$?

CHAPITRE 4

4.40 On considère la proposition P : « $\forall n \in \mathbb{N}, \exists p \in \mathbb{N}, np > 100$ ». Énoncer la négation de P puis donner, en justifiant, la valeur de vérité de P.

4.41 On considère la proposition P : « $\forall x \in]0;1[, \exists y \in]0;1[, x < y$ ». Énoncer la négation de P puis donner, en justifiant, la valeur de vérité de P.

4.42 On considère la proposition P : « $\exists x \in]0;1[, \forall y \in]0;1[, x < y$ ». Énoncer la négation de P puis donner, en justifiant, la valeur de vérité de P.

4.43 Écrire la négation de chacun des énoncés suivants :

- A= « $x > 4$ et $x \neq 8$ »
- B= « $x \in [1;2]$ ou $x < 0$ »
- C= « $(x^2 \leq 4$ et $x \geq 0)$ ou $x^2 \geq 5$ »

4.44 Étudier la valeur de vérité de chacune des propositions suivantes, puis en écrire la négation :

a) $\exists x \in \mathbb{R}, \sqrt{x} > 3 \wedge x^2 < 50$

b) $\forall x \in \mathbb{R}, x \geq 0 \vee x \leq 1$

c) $\forall x \in \mathbb{R}, \exists y \in [0;+\infty[, x = \sqrt{y} \vee xy \leq 0$

4.45 a) On considère les propositions suivantes : « $7 < 4$ » et « $\sqrt{5} > 2$ ». Quelles sont leurs valeurs de vérité ?

b) En déduire quelles sont les valeurs de vérité des propositions suivantes :

A = « $7 < 4$ et $\sqrt{5} > 2$ » B = « $7 < 4$ ou $\sqrt{5} > 2$ »

C = « $7 < 4 \Rightarrow \sqrt{5} > 2$ » D = « $\sqrt{5} > 2 \Rightarrow 7 < 4$ »

E = « $7 < 4 \Leftrightarrow \sqrt{5} > 2$ »

4.46 a) Quelles sont les valeurs de vérité des propositions suivantes :

A = « $\sqrt{3} = 1{,}732$ », B = « $0{,}5^2 > 0{,}5$ », C = « $0{,}42 < \frac{3}{7} < 0{,}43$ ».

b) En déduire la valeur de vérité des propositions suivantes : A et B, A ou B, A$\Rightarrow$B, B$\Rightarrow$A, A$\Leftrightarrow$B, (A et C)$\Rightarrow$B, (A ou C)$\Rightarrow$B, A$\Rightarrow$(B ou C), (A$\Leftrightarrow$B)$\Leftrightarrow$C.

4.47 Vrai ou faux ? (Justifier)

a) $\exists\, x \in \mathbb{R},\ x = \sqrt{x}$

b) $\forall\, x \in \mathbb{R},\ 3(x/4) = 3x/12$

c) $\forall\, x \in \mathbb{R},\ (2-x)^2 = (x-2)^2$

d) $\exists\, x \in \mathbb{R},\ x < 2/x$

4.48 Énoncer deux propositions de valeurs de vérité différentes dans lesquelles apparaît le prédicat « $x^3 < x$ ».

4.49 Dans chacun des cas suivants, énoncer deux propositions dans lesquelles apparaît le prédicat donné : l'une vraie, l'autre fausse.

a) « $x^2 + 1 > 2x$ » b) « $x + y = 4$ »

4.50 Pour chaque proposition, donner sa valeur de vérité, énoncer sa négation puis donner la valeur de vérité de celle-ci.

a) $\exists\, x \in \mathbb{R},\ 1/x = 0$

b) $\forall\, n \in \mathbb{N},\ 1^n = 1$

c) $\exists\, x \in [0;1],\ \forall\, y \in [0;1],\ y^2 \leq x$

d) $\forall\, x \in \mathbb{R},\ \exists\, y \in \mathbb{R},\ (x \neq y \text{ et } x^2 = y^2)$

4.51 Pour chaque proposition, donner sa valeur de vérité, énoncer sa négation puis donner la valeur de vérité de celle-ci.

a) $\forall\, z \in \mathbb{R},\ \exists\, x \in \mathbb{R},\ \exists\, y \in \mathbb{R},\ xy = z$

b) $\exists\, x \in \mathbb{R},\ \exists\, y \in \mathbb{R},\ \forall\, z \in \mathbb{R},\ xy = z$

4.52 Pour les valeurs de a, b et c données, calculer chacune des expressions A, B et C :

$$A = a + \bar{b} + \bar{c}, \qquad B = ab + \bar{c}, \qquad C = (\bar{a} + b)c$$

a) $a = 1, b = 1, c = 0$ b) $a = 1, b = 1, c = 1$ c) $a = 0, b = 1, c = 0$

4.53 Pour les valeurs de a, b et c données, calculer chacune des expressions A, B et C :

$$A = a + \overline{b + \bar{c}}, \qquad B = \left(a + \bar{b}\right)\left(\bar{a} + c\right) \qquad C = \left(\bar{a} + b\right)\bar{c} + \left(a + \bar{b}\right)c$$

a) $a = 1, b = 1, c = 0$ b) $a = 1, b = 1, c = 1$ c) $a = 0, b = 1, c = 0$

4.54 Démontrer à l'aide d'une table de vérité les formules suivantes :

a) $\bar{b} + a\bar{b} = \bar{b}$ b) $a(a + \bar{b}) = a$

4.55 Démontrer à l'aide d'une table de vérité les formules suivantes :

a) $\overline{a}\,\overline{b}+\overline{a}b+a\overline{b}=\overline{ab}$ b) $(a+b)a\overline{b}=a\overline{b}$

4.56 Démontrer à l'aide d'une table de vérité les formules suivantes :

a) $(ab+c)(ab+\overline{c})=ab$ b) $(a+\overline{b})(b+\overline{c})(c+\overline{a})=abc+\overline{a}\,\overline{b}\,\overline{c}$

4.54 Démontrer par calculs les formules suivantes :

a) $\overline{b}+a\overline{b}=\overline{b}$ b) $a(a+\overline{b})=a$

4.55 Démontrer par calculs les formules suivantes :

a) $\overline{a}\,\overline{b}+\overline{a}b+a\overline{b}=\overline{ab}$ b) $(a+b)a\overline{b}=a\overline{b}$

4.56 Démontrer par calculs les formules suivantes :

a) $(ab+c)(ab+\overline{c})=ab$ b) $(a+\overline{b})(b+\overline{c})(c+\overline{a})=abc+\overline{a}\,\overline{b}\,\overline{c}$

4.57 a) Représenter par un tableau de Karnaugh les expressions $A=a\overline{c}+b$ et $B=ab+\overline{a}\,\overline{c}$.

b) Utiliser le tableau pour déterminer les expressions de $\overline{A}$ et $\overline{B}$.

4.58 a) Représenter par un tableau de Karnaugh les expressions :

$$A=a\overline{b}+\overline{a}b+ab\overline{c} \text{ et } B=ab+ac+\overline{a}\,\overline{b}$$

b) Utiliser le tableau pour déterminer les expressions de $\overline{A}$ et $\overline{B}$.

4.59 Soit $A=a+bc$ et $B=\overline{a}b+\overline{c}$. Montrer par calculs que $\overline{A}=\overline{a}\,\overline{b}+\overline{a}\,\overline{c}$ et que $\overline{B}=a\overline{c}+\overline{b}\,\overline{c}$.

4.60 On définit une loi * par $a*b=\overline{a}\cdot b$

a) Montrer que $a*a=0$

b) Calculer $a*\overline{a}$

c) Est-ce que $a*1=1*a$?

d) Est-ce que $a*0=0*a$?

e) Est-ce que $(a*b)*c=a*(b*c)$?

4.61 Un centre téléphonique reçoit les appels des clients de l'entreprise iTélé. Pour chaque appel on considère les variables booléennes a, b, c telles que :

- $a=1$ quand une commande est passée lors de l'appel ($a=0$ sinon) ;
- $b=1$ quand une demande de catalogue est faite lors de l'appel ($b=0$ sinon) ;
- $c=1$ quand le SAV est demandé lors de l'appel ($c=0$ sinon).

Soit l'expression booléenne $A=a.(\overline{b}+c)+\overline{a}$.

a) À l'aide d'un tableau de Karnaugh, montrer que $A=\overline{a}+\overline{b}+c$

b) Retrouver cette simplification par le calcul.

c) Donner l'expression de $\overline{A}$.

d) On considère un appel qui correspond à $\overline{A}$. Préciser en français de quel type d'appel il s'agit.

4.62 Un immeuble de 30 étages ne comporte qu'un seul ascenseur A. Pour limiter le temps d'attente au rez-de-chaussée, on décide de mettre un deuxième ascenseur B en service avec un logiciel qui le mettra en service sous certaines conditions.

On considère les trois variables booléennes s, m et r définies de la façon suivante :

- $s = 1$ lorsque l'ascenseur A est à un étage supérieur au 15ᵉ étage ;
- $m = 1$ lorsque l'ascenseur A monte ;
- $r = 1$ lorsque l'ascenseur A est appelé.

a) Traduire par une expression booléenne chacune des situations suivantes :

- 1) « *L'ascenseur A est appelé alors qu'il est à un étage supérieur au 15ᵉ étage.* »
- 2) « *L'ascenseur A est appelé ou il ne monte pas.* »

b) Quelle situation est traduite par l'expression booléenne $mr + \overline{s}$?

c) À la suite d'une étude, on fait en sorte que l'ascenseur B descende au rez-de-chaussée chaque fois que l'expression booléenne $F = s \cdot r + m \cdot \overline{r} + \overline{s} \cdot m + s \cdot \overline{m} \cdot \overline{r}$ vaut 1.

- 1) Quelle est la valeur de F lorsque $s = 1$, $m = 0$ et $r = 0$?
- 2) L'ascenseur B descendra-t-il au rez-de-chaussée si l'ascenseur A est appelé et qu'il descend alors qu'il se trouve au 10ᵉ étage ?
- 3) À l'aide d'un tableau de Karnaugh, écrire F comme somme de deux variables booléennes.
- 4) Traduire l'expression simplifiée de F par une phrase.
- 5) Quelle est l'expression de $\overline{F}$? À quelle situation correspond-elle ?

4.63 Bob et Alice organisent une soirée pour les membres de leur club informatique. Ils décident que pour être invité, il faut :

- être ami d'Alice et Bob,
- ou ne pas être ami de Bob, mais être ami d'Alice,
- ou ne pas être ami d'Alice mais jouer au poker.

Pour un membre quelconque, on considère les variables booléennes suivantes : $a = 1$ s'il est ami de Bob, $b = 1$ s'il joue au poker, $c = 1$ s'il est ami d'Alice.

a) Écrire la fonction booléenne $f(a,b,c)$ qui traduit le fait qu'un membre du club soit invité.

b) Faire le tableau de Karnaugh de cette fonction.

c) Charles est ami de Bob, mais pas d'Alice. Est-il nécessairement invité ? Justifier.

David n'est pas ami de Bob mais joue au poker. Est-il nécessairement invité ? Justifier.

d) Simplifier l'expression de $f(a,b,c)$, à l'aide du tableau.

e) Énoncer la règle de décision d'inviter un membre du club, de la façon la plus simple possible.

Chapitre 5

5.36 Donner en extension l'ensemble A des multiples de 6 compris entre 50 et 80. Définir A en compréhension avec une formulation mathématique.

5.37 Soit $A = \{1;4;9;16;25;36\}$. Définir A en compréhension avec une phrase de français, puis avec une formulation mathématique.

5.38 Soit $A = \{k(k+1), k \in \mathbb{N}, 0 \leq k \leq 7\}$. Écrire A en extension.

5.39 Donner en extension l'ensemble A des valeurs affichées par cet algorithme :

```
Variables : I, J (entiers)
Début
    Pour I de 1 à 4 Faire
        Pour J de 0 à 3 Faire
            Afficher 10I+J
        FinPour
    FinPour
Fin
```

5.40 On considère les ensembles suivants : $E = \{1;2;3;4;5;6;7;8;9\}$, $A = \{1;2;5;6;7\}$, $B = \{2;4;6;8\}$, $C = \{3;6;7;9\}$.

a) Déterminer $A \cap B$, $A \cap C$, $B \cap C$.

b) Déterminer $A \cup B$, $A \cup C$, $B \cup C$.

c) Déterminer $\overline{A}$, complémentaire de A dans E. Déterminer ensuite $\overline{B}$ et $\overline{C}$.

d) Vérifier que $\overline{A \cap B} = \overline{A} \cup \overline{B}$.

5.41 $A = \{1;4;5;6;8;9\}$, $B = \{2;3;4;5;9\}$, $C = \{1;2;4;7;10\}$.

a) Déterminer $B \cap C$ puis $A \cup (B \cap C)$.

b) Déterminer $A \cup B$, $A \cup C$ puis $(A \cup B) \cap (A \cup C)$.

c) Quelle propriété vient-on d'illustrer ?

5.42 a) Soit $E = \{1;2;3\}$. Déterminer $\mathcal{P}(E)$.

b) Soit $F = \{1;2;3;4\}$. Faire la liste des parties de E à deux éléments. Quel est le cardinal de $\mathcal{P}(F)$?

5.43 A et B étant deux parties d'un ensemble E, on définit l'ensemble $A \Delta B$ (lire A delta B), appelé différence symétrique de A et B, par :

$$A \Delta B = \{x \in E \,/\, x \in A \cup B \wedge x \notin A \cap B\}$$

a) 1) Déterminer $A \Delta B$ si $E = \{1;2;3;4;5;6;7;8;9\}$, $A = \{1;2;5;6;7\}$, $B = \{2;4;6;8\}$.
2) On note $\overline{A}$ et $\overline{B}$ les complémentaires de A et de B dans E. Déterminer $\overline{A} \Delta \overline{B}$. Que peut-on remarquer ?

b) À l'aide d'une table de vérité, montrer que, quels que soient les ensembles A et B, on a $A \Delta B = \overline{A} \Delta \overline{B}$.

5.44 A et B sont deux parties d'un ensemble E. $A \Delta B$ désigne la différence symétrique de A et de B (voir définition dans l'exercice **5.43**).

a) Déterminer $A \Delta E$, $A \Delta \varnothing$, $A \Delta \overline{A}$.

b) Montrer que $A = B \Leftrightarrow A \Delta B = \varnothing$.

5.45 A et B sont deux parties d'un ensemble E.

a) À l'aide de tables de vérité, montrer les formules suivantes :
$A \cap (\overline{A} \cup B) = A \cap B$ et $A \cup (\overline{A} \cap B) = A \cup B$.

b) Redémontrer les formules précédentes à l'aide des propriétés de distributivité de l'intersection et de la réunion.

5.46 A et B étant deux ensembles, on définit l'ensemble $A \setminus B$, appelé différence de A et de B, comme étant l'ensemble des éléments de A qui ne sont pas dans B. Autrement dit, $A \setminus B = \{x \in A, x \notin B\}$.

a) Soit $A = [0;3]$, $B = [1;4]$, $C = [2;5]$. Déterminer les ensembles $(A \cap B) \setminus C$, $A \cap (B \setminus C)$ et $(A \setminus C) \cap (B \setminus C)$.

b) A, B et C étant trois ensembles quelconques, montrer à l'aide de tables de vérité que : $(A \cap B) \setminus C = A \cap (B \setminus C) = (A \setminus C) \cap (B \setminus C)$.

5.47 A et B étant deux ensembles, $A \setminus B$ désigne la différence de A et de B, définie à l'exercice **5.46**. Montrer que, quels que soient les ensembles A, B et C, on a $A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$.

5.48 Soit E l'ensemble des multiples de 3 compris entre 50 et 100, et F l'ensemble des multiples de 4 compris entre 50 et 150. Dans le plan, on considère l'ensemble A des points de coordonnées $(x;y)$ tels que x appartient à E et y appartient à F.

a) Déterminer le cardinal de E, puis le cardinal de F.

b) En déduire le nombre de points constituant l'ensemble A.

Pour chacun des exercices de **5.49** à **5.53** étudier si la relation $\mathcal{R}$ est réflexive, symétrique, antisymétrique, transitive. Éventuellement, dire si c'est une relation d'équivalence ou d'ordre (total ou partiel).

5.49 Dans $\mathbb{R}$, $x \mathcal{R} y \Leftrightarrow |x - y| \leq 1$.

5.50 Dans $\mathbb{N}$, $x \mathcal{R} y \Leftrightarrow x + y$ est multiple de 3.

5.51 Dans $\mathbb{N}$, $x \mathcal{R} y \Leftrightarrow x \equiv y[5]$.

5.52 Soit E un ensemble non vide et A un sous-ensemble non vide de E. Dans $\mathcal{P}(E)$, $X \mathcal{R} Y \Leftrightarrow (A \cap X) = (A \cap Y)$.

5.53 Soit E un ensemble non vide. Dans $\mathcal{P}(E)$, $A \mathcal{R} B \Leftrightarrow A \cup B = E$.

5.54 On se place dans $E = \mathbb{R}^2$, ensemble de tous les couples de réels. Si on note $x = (x_1,x_2)$ et $y = (y_1,y_2)$, alors $x \mathcal{R} y \Leftrightarrow (x_1 < y_1)$ ou $(x_1 = y_1) \wedge (x_2 \leq y_2)$.

a) Soit $x = (4;8,5)$ et $y = (2,1;10)$. Est-ce que $x \mathcal{R} y$ ou $y \mathcal{R} x$?

b) Montrer que $\mathcal{R}$ est une relation d'ordre. Est-ce un ordre total ou partiel ?

5.55 Soit f la relation de $\mathbb{N}$ dans $\mathbb{N}$ telle que $f(n) = n/2$ si n est pair, et $f(n) = (n + 1)/2$ si n est impair.

a) Montrer que f est une application.

b) f est-elle injective ? Surjective ? Bijective ?

5.56 Soit f l'application de $[0;+\infty[$ dans $\mathbb{N}$ telle que, pour tout $x \in [0;+\infty[$, $f(x)$ est la partie entière de x, c'est-à-dire le plus grand entier naturel inférieur ou égal à x.

a) f est-elle injective ? Surjective ?

b) Quelle est l'image directe de $[3,2;5,7[$ par f ?

c) Quelle est l'image réciproque de $\{2;3\}$ par f ?

5.57 Soit f l'application de $[0;+\infty[$ dans $\mathbb{N}$ qui, à tout $x \in [0;+\infty[$ associe sa partie entière (voir exercice **5.56**).

a) Soit $A = [2,5;4,3]$. Déterminer $f^{-1}[f(A)]$. Est-ce que $f^{-1}[f(A)] = A$?

b) Soit $B = \{5;7\}$. Déterminer $f^{-1}[f(B)]$. Est-ce que $f^{-1}[f(B)] = B$?

5.58 Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$, définie par $f(x) = x^2$.

On considère les intervalles $A = [0;3]$ et $A' = [1;5]$

a) Est-ce que $f(A \cap A') = f(A) \cap f(A')$?

b) Est-ce que $f(A \cup A') = f(A) \cup f(A')$?

c) Est-ce que $f\left(\overline{A}\right) = \overline{f(A)}$?

5.59 Soit f l'application de $\mathbb{R}$ dans $\mathbb{R}$, définie par $f(x) = x^2$. On considère les intervalles $B = [0;4]$ et $B' = [1;9]$.

a) Est-ce que $f^{-1}(B \cap B') = f^{-1}(B) \cap f^{-1}(B')$?

b) Est-ce que $f^{-1}(B \cup B') = f^{-1}(B) \cup f^{-1}(B')$?

c) Est-ce que $f^{-1}\left(\overline{B}\right) = \overline{f^{-1}(B)}$?

5.60 Soit f et g les applications de $\mathbb{R}$ dans $\mathbb{R}$ définies par $f(x) = 10x$ et $g(x) = 3x + 5$.

a) Montrer que f et g sont des bijections.

b) Définir les applications $g \circ f$ et $f \circ g$. Sont-elles bijectives ?

5.61 Soit f et g les applications de $\mathbb{R}$ dans $\mathbb{R}$ définies par $f(x) = 4x$ et $g(x) = x - 2$.

a) Montrer que f et g sont des bijections. Définir les bijections réciproques.

b) Définir la bijection $g \circ f$.

c) Définir la bijection réciproque de $g \circ f$ de deux façons.

5.62 Soit f une bijection de E dans F, et g une bijection de F dans G.

a) Démontrer que $g \circ f$ est une bijection de E dans G.

b) Démontrer que $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$.

5.63 Soit E un ensemble non vide et A un sous-ensemble non vide de E. On définit l'application f dans $\mathcal{P}(E)$ par : $\forall X \in \mathcal{P}(E)$, $f(X) = A \cap X$. L'application f est-elle injective ? Surjective ?

5.64 Soit E un ensemble non vide et A un sous-ensemble non vide de E. Dans $\mathcal{P}(E)$, on définit l'application f par : $\forall X \in \mathcal{P}(E)$, $f(X) = A \cup X$. L'application f est-elle injective ? Surjective ?

5.65 On considère les 10 lettres A, B, C, D, E, F, G, H, I, J auxquelles on associe les dix nombres de 1 à 10, dans l'ordre. Soit $S = \{1;2;3;4;...;9;10\}$.

a) Dans S, on définit l'application f de la façon suivante : $\forall n \in S$, $f(n)$ est le reste de la division euclidienne de 2^n par 11. On souhaite utiliser la fonction f pour coder des messages Compléter le tableau 5.15. Ce type de codage est-il bijectif ?

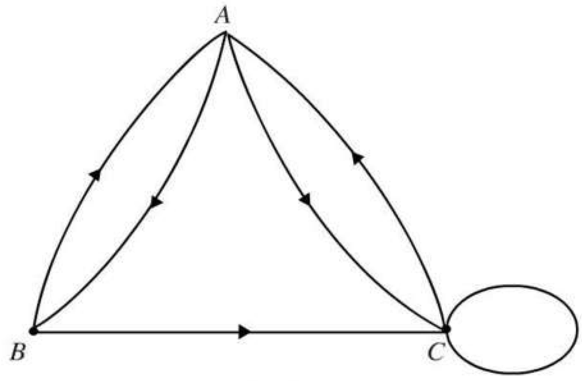
Tableau 5.15

Lettre	A	B	C	D	E	F	G	H	I	J
n	1	2	3	4	5	6	7	8	9	10
$f(n)$										
Lettre										

b) Même question si on utilise la fonction g telle que : $\forall n \in S$, $g(n)$ est le reste de la division euclidienne de 3^n par 11.

Chapitre 6

6.20 Pour chacun des graphes (fig. 6.35), faire le tableau des successeurs et des prédécesseurs, et écrire la matrice d'adjacence (les sommets seront à considérer dans l'ordre alphabétique).



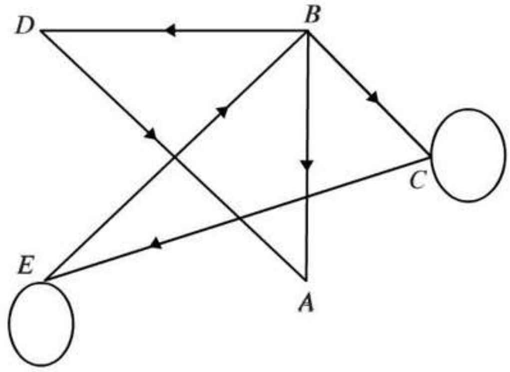


Figure 6.35

6.21 a) Faire le tableau des prédécesseurs, puis construire le graphe associé.

Tableau 6.34

Sommet	*A*	*B*	*C*	*D*
Successeurs	*B, D*	*A, C*	*A, C*	*A*

b) Écrire la matrice d'adjacence du graphe.

6.22 a) Compléter le tableau 6.35 en ajoutant les prédécesseurs, puis construire le graphe associé.

Tableau 6.35

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs	*A, E*	*C, D*	*A*	---	*B, D*
Prédécesseurs					

b) Écrire la matrice d'adjacence du graphe.

6.23 a) Compléter le tableau 6.36 en ajoutant les successeurs.

Tableau 6.36

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs					
Prédécesseurs	*A, C*	*E*	*A*	*A, B*	*C*

b) Dessiner le graphe associé au tableau 6.36 et écrire la matrice d'adjacence.

6.24 a) Compléter le tableau 6.37 en ajoutant les successeurs.

Tableau 6.37

Sommet	*A*	*B*	*C*	*D*	*E*
Successeurs					
Prédécesseurs	*D, E*	*A*	*A, C, E*	*B, C*	*B, D*

b) Dessiner le graphe associé au tableau 6.37 et écrire la matrice d'adjacence.

6.25 M est la matrice d'adjacence d'un graphe défini sur l'ensemble $S = \{A; B; C; D\}$.

$$M = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

a) Faire le tableau des successeurs et des prédécesseurs du graphe.

b) Dessiner le graphe.

6.26 M est la matrice d'adjacence d'un graphe défini sur l'ensemble $S = \{A; B; C; D; E\}$.

$$M = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

a) Faire le tableau des successeurs et des prédécesseurs du graphe.

b) Dessiner le graphe.

6.27 Un graphe est défini par le tableau 6.38.

Tableau 6.38

Sommets	*A*	*B*	*C*	*D*
Successeurs	*D*	*A*	*A,B,D*	*B,C*

a) Écrire la matrice d'adjacence M de ce graphe.

b) Calculer M^2. Expliquer la signification des nombres de la quatrième ligne de la matrice.

c) Calculer M^3. En déduire le nombre de chemins de longueur 3.

d) Parmi tous les chemins de longueur 3, combien y en a-t-il qui ne relient pas un sommet à lui-même ? Et parmi eux, y a-t-il des chemins hamiltoniens ? Lesquels ?

6.28 Un graphe est défini par le tableau 6.39.

Tableau 6.39

Sommets	*A*	*B*	*C*	*D*	*E*
Successeurs	*C*, *D*	*A*	*E*	*A*	*B*, *D*

a) Écrire la matrice d'adjacence M de ce graphe.

b) Combien y a-t-il de chemins de longueur 4 au départ de D ? Les citer.

c) Combien y a-t-il de chemins de longueur 4 arrivant sur D ? Les citer.

6.29 Un graphe est défini par le tableau 6.40.

Tableau 6.40

Sommets	*A*	*B*	*C*	*D*	*E*	*F*
Prédécesseurs	*B*	*E*	*B*, *F*	*A*	*F*	*D*

a) Écrire la matrice d'adjacence du graphe.

b) En utilisant une puissance de la matrice M, déterminer le nombre maximum de chemins hamiltoniens du graphe. S'il en existe, écrire ces chemins.

c) Le graphe possède-t-il des circuits de longueur 6 ? Si oui, lesquels ?

6.30 Un graphe défini sur l'ensemble $S = \{A; B; C\}$ est constitué de trois arcs : un arc relie A à B, un autre relie B à C, et le troisième arc relie C à A.

a) Tracer le graphe et faire sa fermeture transitive.

b) Écrire la matrice d'adjacence M du graphe.

c) Calculer les matrices booléennes $M^{[2]}$ et $M^{[3]}$

En déduire la matrice $\hat{M}$ de la fermeture transitive.

6.31 Un graphe est défini par le tableau 6.41.

Tableau 6.41

Sommets	*A*	*B*	*C*	*D*	*E*
Successeurs	*B*, *E*	---	---	*E*	*C*

a) Dessiner le graphe.

b) Calculer la matrice $\hat{M}$ de la fermeture transitive. Quels arcs faut-il ajouter au graphe pour faire sa fermeture transitive ?

6.32 Un graphe est défini par le tableau 6.42. Déterminer le niveau de chaque sommet et représenter le graphe en tenant compte des niveaux.

Tableau 6.42

Sommets	A	B	C	D	E	F	G	H
Successeurs	C,E,F	---	B	A,G	---	E	C,H	B,F

6.33 Un graphe est défini par le tableau 6.43. Déterminer le niveau de chaque sommet et représenter le graphe en tenant compte des niveaux.

Tableau 6.43

Sommets	A	B	C	D	E	F	G	H	I	J
Successeurs	F,H	F	A,B,D	G	C,J	G,I	---	---	H	A

6.34 Un graphe est défini par le tableau 6.44. Déterminer le niveau de chaque sommet et représenter le graphe en tenant compte des niveaux.

Tableau 6.44

Sommets	A	B	C	D	E	F	G	H
Successeurs	C,E,F	---	B	A,G	---	E	C,H	B,F

6.34 Sur le graphe de la fig. 6.36, déterminer le chemin de longueur minimale reliant A à H, puis le chemin de longueur maximale reliant A à H (voir exercice corrigé **6.16**).

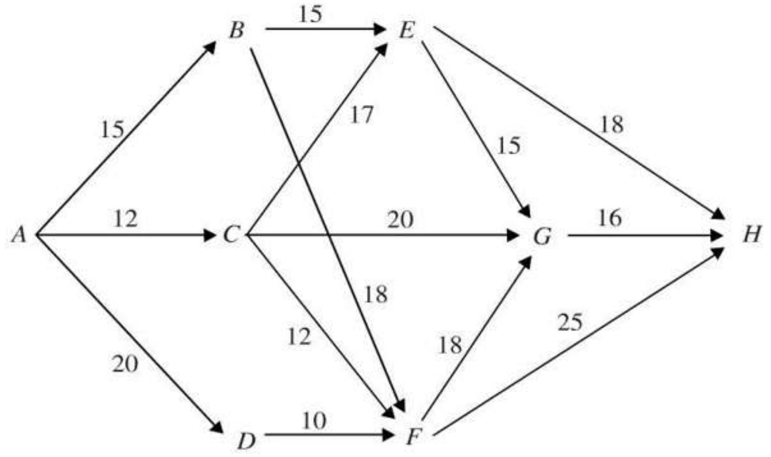


Figure 6.36

6.35 Alice, Bob, Charly, David, Emma et Fred ont chacun créé un blog pour parler de leur passion. En un clic, on peut passer :

- du blog d'Alice aux blogs de Bob, d'Emma ou de Fred.
- du blog de Bob au blog de Charly.
- du blog de Charly aux blogs de Bob ou de David.
- du blog de David au blog de Charly.
- du blog d'Emma aux blogs de Charly ou de David.
- du blog de Fred aux blogs de Bob ou d'Alice.

a) Dessiner le graphe, dans lequel un arc indiquera la possibilité de passer d'un blog à un autre.

b) Indiquer toutes les possibilités pour passer du blog d'Alice à chacun des autres blogs : en deux clics, en trois clics.

c) Écrire la matrice d'adjacence M du graphe. Calculer M^2 et M^3. Comment retrouve-t-on les nombres de possibilités trouvés à la question b ?

d) De combien de façons un internaute peut-il passer du blog d'Alice au blog de Charly en quatre clics ?

e) De combien de façons un internaute peut-il passer du blog de Charly au blog d'Alice en quatre clics ?

f) Est-il possible qu'un internaute visite les six blogs en cinq clics ?

6.36 Une société de services techniques en informatique doit mettre en place un réseau interne de 50 ordinateurs pour une entreprise. Les tâches nécessaires à la réalisation de ce projet ont été reproduites dans le tableau 6.45.

Tableau 6.45

Description de la tâche	Abréviation	Tâches antérieures	Durée (en jours)
Identification des besoins matériels/ logiciels et commandes	COM	aucune	1
Acheminement / livraisons des OS / logiciels	LOG	COM	3
Achat du matériel pour les UC + câbles réseau	MAT	COM	1
Acheminement / livraison des écrans	ECR	COM	6
Assemblage des UC	ASS	MAT	1,5
Installation des OS / logiciels	INST	LOG, ASS	2
Pose des câbles réseau dans l'entreprise	CABL	MAT	4
Mise en place des postes dans l'entreprise	POST	INST, ECR	1
Configuration du réseau interne	CONF	POST, CABL	1

On considère le graphe orienté de sommets COM, LOG,MAT, ECR, ASS, INST, CABL, POST, CONF correspondant aux conditions d'antériorités données par le tableau précédent.

a) Déterminer le niveau de chacun des sommets du graphe.

b) Construire le graphe d'ordonnancement selon la méthode MPM, et établir les dates au plus tôt et au plus tard de chaque tâche.

c) Déterminer le chemin critique et la durée de réalisation du projet.

d) Calculer la marge totale de la tâche ASS. À quoi correspond-elle ?

e) Calculer la marge libre de la tâche ASS. À quoi correspond-elle ?

6.37 Un projet est constitué de sept tâches. Les contraintes et les durées de ces tâches sont données dans le tableau 6.46.

Tableau 6.46

Tâche	Durée en jours	Tâches antérieures
A	5	aucune
B	8	aucune
C	6	aucune
D	5	A, B
E	9	C
F	3	D, E
G	6	D

a) Quels sont les prédécesseurs de la tâche D ? Quels sont ses successeurs ?

b) Déterminer le niveau de chacune des tâches.

c) Construire le graphe d'ordonnancement du projet, en donnant pour chaque tâche la date au plus tôt et la date au plus tard.

d) Déterminer le chemin critique et la durée minimale du projet.

e) Calculer la marge totale de E. À quoi correspond-elle ?

f) Calculer la marge libre de E. À quoi correspond-elle ?

6.38 Un projet nécessite la réalisation d'un certain nombre de tâches. Le nom de chaque tâche, sa durée en jours et les tâches qui lui sont antérieures, sont donnés dans le tableau 6.47.

Tableau 6.47

Tâche	Durée	Tâches antérieures
A	6	aucune
B	9	aucune
C	5	aucune
D	4	A, B
E	3	A, C
F	5	C
G	2	D
H	6	E, F
I	6	G
J	1	H, I

a) Faire le diagramme MPM et déterminer le chemin critique.

b) Quelle est la durée minimum du projet ?

c) Calculer la marge totale de la tâche H. À quoi correspond-elle ?

d) Suite à un retard dans la fourniture de matériel, la tâche F a nécessité 9 jours au lieu de 5. Cela a-t-il eu une incidence sur la date de fin du projet ?

6.39 Une entreprise souhaite implanter un nouveau réseau informatique dans ses locaux. Les tâches nécessaires à ce projet, leurs durées ainsi que les conditions d'antériorité sont présentées dans le tableau 6.48.

Tableau 6.48

Tâche	Durée	Tâches antérieures
A	6	aucune
B	7	aucune
C	4	aucune
D	4	A, C
E	10	A, B, C
F	9	A, D
G	5	E
H	3	D, F
I	6	G, H
J	3	I
K	2	I
L	5	G, J, K
M	3	L

a) Construire le graphe d'ordonnancement du projet.

b) Déterminer le chemin critique et la durée minimale de réalisation du projet.

c) Calculer la marge totale et la marge libre de chaque tâche.

d) Le responsable du projet redoute un allongement de la durée de la tâche H. Au lieu de durer 3 jours, elle pourrait nécessiter 8 jours. Indiquer si cette durée a une influence sur le chemin critique et la durée de réalisation du projet.

CHAPITRE 9

9.21 Que vaut la variable *x* après exécution de l'algorithme suivant ?

```
Variables : x, y (chaînes), n (entier)
Début
  x ← 'D'
  y ← 'ix'
  n ← 10
  x ← x+y+' : '+str(n)
Fin
```

9.22 Que vaut la variable *n* après exécution de l'algorithme suivant ?

```
Variables : a (chaîne), n (entier)
Début
  a ← 'azerty'
  n ← 15
  n ← n + longueur (a)
Fin
```

9.23 Que vaut la variable *ch1* après exécution de l'algorithme suivant ?

```
Variables : ch1, ch2 (chaînes)
Début
  ch1 ← 'clavie'
  ch2 ← 'qwerty'
  ch1 ← ch1+ch2[3]+' az'+ch2[2:6]
Fin
```

9.24 Écrire un algorithme demandant à l'utilisateur une somme d'argent en euros (€) et qui affiche la meilleure décomposition de cette somme avec des pièces de 2 €, 1 €, 50 c, 20 c, 10 c, 5 c, 2 c et 1 c, c'est-à-dire utilisant le moins de pièces possible. Implémenter en Python puis exécuter le programme avec le nombre 5,77. Quelle est la meilleure décomposition de la somme 5,77 € ?

9.25 Écrire l'en-tête de l'algorithme suivant (nom, rôle, entrée(s), sortie(s)).

```
Variables : a,b (réels)
Début
    Afficher "Saisir un nombre a :"
    Saisir a
    Afficher "Saisir un nombre b :"
    Saisir b
    b ← a + b
    a ← b - a
    b ← b - a
    Afficher "a=",a," ; b=",b
Fin
```

9.26 Écrire un algorithme demandant à l'utilisateur une note sur 20 puis affichant « *Refusé* » si elle est inférieure à 8, « *Passe l'oral de rattrapage* » si elle est comprise entre 8 et 10 (8 compris, 10 non compris) et « *Admis* » si elle est supérieure ou égale à 10.

9.27 Écrire un algorithme demandant à l'utilisateur deux nombres réels a et b ainsi qu'un un opérateur parmi +, –, et *, puis affichant le résultat du calcul $a + b$, $a - b$ et $a \times b$ selon l'opérateur choisi.

9.28 Un échiquier possède 64 cases (8×8 cases dont les couleurs sont alternées noires ou blanches). On repère ces cases à l'aide d'une liste $[i,j]$ de deux nombres entiers où i est le numéro de la ligne et j celui de la colonne. On décide d'orienter l'échiquier de sorte que la case correspondant à la liste [1,1] soit noire.

Écrire un algorithme demandant à l'utilisateur la liste correspondant à une case d'échiquier puis affichant la couleur de cette case.

9.29 Quel est le rôle de l'algorithme suivant ?

```
Variables : n, p (entiers)
Début
    Afficher "Rentrer deux nombres entiers : "
    Saisir n
    Saisir p
    Si n%10==int(str(p)[0]) Alors
        Afficher("Ça marche !")
    FinSi
Fin
```

9.30 Écrire un algorithme déterminant puis affichant le nombre de vendredi 13 d'une année non bissextile commençant le jeudi 1er janvier.

9.31 Écrire un algorithme demandant à l'utilisateur de saisir un texte, et affichant ce texte sans les espaces. Par exemple, pour la saisie de « *Il fait beau aujourd'hui* », on devra obtenir l'affichage « *Ilfaitbeauaujourd'hui* ».