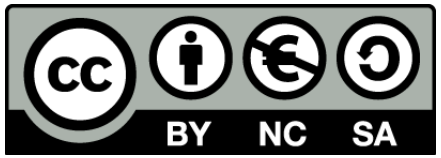
Sciences Numériques et Technologie

Brochure « professeur »

https://www.xm1math.net/

Version 1.2 Mars 2022



Cette brochure a été réalisée avec le système de composition LaTeX et l'éditeur TeXMAKER .
https://www.xm1math.net/texmaker/index_fr.html

Sommaire

1 Algorithmique - Initiation à python

1. Structures de base en langage naturel

1) Variables et affectations

Les variables en algorithmique

- Les variables algorithmiques peuvent servir à stocker des données de différents types, mais nous nous contenterons ici d'utiliser des variables du type NOMBRE.
- La valeur d'une variable peut changer au fil des instructions de l'algorithme.
- Les opérations sur les variables s'effectuent ligne après ligne et les unes après les autres.
- Quand une ligne du type « mavariable ← un calcul », on effectue d'abord le calcul et on stocke ensuite le résultat dans mavariable.

► Exercice n°1
On considère l'algorithme suivant :

```
Variables: x,y,z
1: DEBUT_ALGORITHME
2:     x ← 2
3:     y ← 3
4:     z ← x+y
5: FIN_ALGORITHME
```

Après exécution de l'algorithme : la variable x contient la valeur : 2 ; la variable y contient la valeur : 3 ; la variable z contient la valeur : 5

► Exercice n°2
On considère l'algorithme suivant :

```
Variables: x
1: DEBUT_ALGORITHME
2:     x ← 2
3:     x ← x+1
4: FIN_ALGORITHME
```

Après exécution de l'algorithme, la variable x contient la valeur : 3

► Exercice n°3
Ajoutons la ligne « x ← 4*x » à la fin du code précédent. Ce qui donne :

```
Variables: x
1: DEBUT_ALGORITHME
2:     x ← 2
3:     x ← x+1
4:     x ← 4*x
5: FIN_ALGORITHME
```

Après exécution de l'algorithme, la variable x contient la valeur : 12

► Exercice n°4
On considère l'algorithme suivant :

```
Variables: A,B,C
1: DEBUT_ALGORITHME
2:     A ← 5
3:     B ← 3
4:     C ← A+B
5:     B ← B+A
6:     A ← C
7: FIN_ALGORITHME
```

Après exécution de l'algorithme : la variable A contient la valeur : 8 ;
la variable B contient la valeur : 8
la variable C contient la valeur : 8

► Exercice n°5

On considère l'algorithme suivant :

```
Variables: x,y,z
1: DEBUT_ALGORITHME
2:     LIRE x
3:     y ← x-2
4:     z ← -3*y-4
5:     AFFICHER z
6: FIN_ALGORITHME
```

On cherche maintenant à obtenir un algorithme équivalent sans utiliser la variable y. Compléter la ligne 3 dans l'algorithme ci-dessous pour qu'il réponde au problème.

```
Variables: x,z
1: DEBUT_ALGORITHME
2:     LIRE x
3:     z ← −3x + 2
4:     AFFICHER z
5: FIN_ALGORITHME
```

2) Instructions conditionnelles

SI...ALORS...SINON

Comme nous l'avons vu ci-dessus, un algorithme permet d'exécuter une liste d'instructions les unes à la suite des autres. Mais on peut aussi demander à un algorithme de n'exécuter des instructions que si une certaine condition est remplie. Cela se fait grâce à au bloc d'instructions SI...ALORS :

```
SI...ALORS
  ...
  FIN_SI
```

Il est aussi possible d'indiquer en plus à l'algorithme de traiter le cas où la condition n'est pas vérifiée. On obtient alors la structure suivante :

```
SI...ALORS
  ...
  FIN_SI
SINON
  ...
  FIN_SINON
```

► Exercice n°6

On cherche à créer un algorithme qui demande un nombre à l'utilisateur et qui affiche la racine carrée de ce nombre s'il est positif. Compléter la ligne 3 dans l'algorithme ci-dessous pour qu'il réponde au problème.

```
Variables: x,racine
1: DEBUT_ALGORITHME
2:     LIRE x
3:     SI (x⩾ 0) ALORS
4:         racine ← √x
5:         AFFICHER racine
6:         FIN_SI
7: FIN_ALGORITHME
```

► Exercice n°7

On cherche à créer un algorithme qui demande à l'utilisateur d'entrer deux nombres (stockés dans les variables x et y) et qui affiche le plus grand des deux. Compléter les ligne 5 et 8 dans l'algorithme ci-dessous pour qu'il réponde au problème.

```
Variables: x,y
1: DEBUT_ALGORITHME
2:     LIRE x
3:     LIRE y
4:     SI (x>y) ALORS
5:         AFFICHER x
6:         FIN_SI
7:     SINON
8:         AFFICHER y
9:         FIN_SINON
10: FIN_ALGORITHME
```

► Exercice n°8

On considère l'algorithme suivant :

```
Variables: A,B
DEBUT_ALGORITHME
    A ← 1
    B ← 3
    SI (A>0) ALORS
        A ← A+1
        FIN_SI
    SI (B>4) ALORS
        B ← B-1
        FIN_SI
FIN_ALGORITHME
```

Après exécution de l'algorithme :

- La variable A contient la valeur : 2
- La variable B contient la valeur : 3

► Exercice n°9

On cherche à concevoir un algorithme correspondant au problème suivant :

- on demande à l'utilisateur d'entrer un nombre (représenté par la variable x)
- si le nombre entré est différent de 1, l'algorithme doit stocker dans une variable y la valeur de 1/(x-1) et afficher la valeur de y (note : la condition x différent de 1 s'exprime avec le code x!=1). On ne demande pas de traiter le cas contraire.

Compléter l'algorithme ci-dessous pour qu'il réponde au problème.

```
Variables: x,y
DEBUT_ALGORITHME
    LIRE x
    SI (x≠1) ALORS
        y ← 1/(x-1
        AFFICHER y
        FIN_SI
FIN_ALGORITHME
```

3) Boucles

Boucles POUR...DE...A

- Les boucles permettent de répéter des instructions autant de fois que l'on souhaite.
- Lorsqu'on connait par avance le nombre de fois que l'on veut répéter les instructions, on utilise une boucle du type POUR...DE...A dont la structure est la suivante :

```
POUR...ALLANT_DE...A...
    ...
FIN_POUR
```

- Exemple : l'algorithme ci-dessous permet d'afficher la racine carrée de tous les entiers de 1 jusqu'à 50.

```
Variables: n et racine
DEBUT_ALGORITHME
    POUR n ALLANT_DE 1 A 50
        racine ← √n
        AFFICHER racine
        FIN_POUR
FIN_ALGORITHME
```

La variable n est appelée « compteur de la boucle ».

- Remarques :
  - La variable servant de compteur pour la boucle doit être du type NOMBRE et doit être déclarée préalablement (comme toutes les variables).
  - Par défaut, cette variable est automatiquement augmentée de 1 à chaque fois.
  - On peut utiliser la valeur du compteur pour faire des calculs à l'intérieur de la boucle, mais les instructions comprises entre POUR et FIN_POUR ne doivent en aucun cas modifier la valeur de la variable qui sert de compteur.

► Exercice n°10

On cherche à concevoir un algorithme qui affiche, grâce à une boucle POUR...DE...A, les résultats des calculs suivants : 8*1 ; 8*2 ; 8*3 ; 8*4 ; ... jusqu'à 8*10 .

La variable n sert de compteur à la boucle et la variable produit sert à stocker et afficher les résultats. Compléter les lignes 2 et 3 dans l'algorithme ci-dessous pour qu'il réponde au problème :

```
Variables: n,produit
DEBUT_ALGORITHME
    POUR n ALLANT_DE 1 A 10
        produit ← 8*n
        AFFICHER produit
        FIN_POUR
FIN_ALGORITHME
```

► Exercice n°11

On considère l'algorithme suivant :

```
Variables: n,somme
1: DEBUT_ALGORITHME
2:     somme ← 0
3:     POUR n ALLANT_DE 1 A 100
4:         somme ← somme+n
5:         FIN_POUR
6:     AFFICHER somme
7: FIN_ALGORITHME
```

Compléter les phrases suivantes :

- Après exécution de la ligne 2, la variable somme contient la valeur : 0
- Lorsque le compteur n de la boucle vaut 1 et après exécution du calcul ligne 4, la variable somme vaut : 0+1
- Lorsque le compteur n de la boucle vaut 2 et après exécution du calcul ligne 4, la variable somme vaut : 0+1+2

Que permet de calculer cet algorithme ?

la somme $1+2+\cdots+100$

► Exercice n°12

Compléter les lignes 3 et 4 de l'algorithme ci-dessous pour qu'il permette de calculer la somme $5^2+6^2+7^2+\cdots+24^2+25^2$.

```
Variables: n, somme
1: DEBUT_ALGORITHME
2:     somme ← 0
3:     POUR n ALLANT_DE 5 A 25
4:         somme ← somme+n^2
5:         FIN_POUR
6:     AFFICHER somme
7: FIN_ALGORITHME
```

Boucles TANT QUE...

- Il n'est pas toujours possible de connaitre par avance le nombre de répétitions nécessaires à un calcul. Dans ce cas là, il est possible d'avoir recours à la structure TANT QUE... qui se présente de la façon suivante :

```
TANT_QUE...FAIRE
    ...
    FIN_TANT_QUE
```

Cette structure de boucle permet de répéter une série d'instructions (comprises entre TANT_QUE...FAIRE et FIN_TANT_QUE) tant qu'*une certaine condition est vérifiée*.

- Exemple : Comment savoir ce qu'il reste si on enlève 25 autant de fois que l'on peut au nombre 583 ? Pour cela on utilise une variable n, qui contient 583 au début, à laquelle on enlève 25 tant que c'est possible, c'est à dire tant que n est supérieur ou égal à 25.

```
Variables: n
1: DEBUT_ALGORITHME
2:     n ← 583
3:     TANT_QUE (n>=25) FAIRE
4:         n ← n-25
5:         FIN_TANT_QUE
6:     AFFICHER n
7: FIN_ALGORITHME
```

- Remarques :
  - Si la condition du TANT QUE... est fausse dès le début, les instructions entre TANT_QUE...FAIRE et FIN_TANT_QUE ne sont jamais exécutées (la structure TANT QUE ne sert alors strictement à rien).
  - Il est indispensable de s'assurer que la condition du TANT QUE... finisse par être vérifiée (le code entre TANT_QUE...FAIRE et FIN_TANT_QUE doit rendre vraie la condition tôt ou tard), sans quoi l'algorithme ne pourra pas fonctionner.

► Exercice n°13

On cherche à connaître le plus petit entier N tel que 2^N soit supérieur ou égal à 10000. Pour résoudre ce problème de façon algorithmique :

- On utilise une variable N à laquelle on donne au début la valeur 1.
- On augmente de 1 la valeur de N tant que 2^N n'est pas supérieur ou égal à 10000.

Une structure TANT QUE est particulièrement adaptée à ce genre de problème car on ne sait pas a priori combien de calculs seront nécessaires.

Compléter les lignes 3 et 4 de l'algorithme ci-dessous pour qu'il réponde au problème :

```
Variables: N
1: DEBUT_ALGORITHME
2:     N ← 1
3:     TANT_QUE (2^N<10000) FAIRE
4:         N ← N+1
5:         FIN_TANT_QUE
6:     AFFICHER N
7: FIN_ALGORITHME
```

► Exercice n°14

Un individu a emprunté à un ami une somme de 2500 euros (prêt sans intérêts). Pour rembourser son ami, il prévoit de lui remettre 110 euros par mois. Mais comme cela ne correspond pas à un nombre pile de mois, il se demande quel sera le montant à rembourser le dernier mois.

Compléter la ligne 3 dans l'algorithme ci-dessous pour qu'il réponde au problème :

```
Variables: montant
1: DEBUT_ALGORITHME
2:     montant ← 2500
3:     TANT_QUE (montant ⩾ 110) FAIRE
4:         montant ← montant-110
5:         FIN_TANT_QUE
6:     AFFICHER montant
7: FIN_ALGORITHME
```

► Exercice n°15

On considère le problème suivant :

- On lance une balle d'une hauteur initiale de 300 cm.
- On suppose qu'à chaque rebond, la balle perd 10% de sa hauteur (la hauteur est donc multipliée par 0.9 à chaque rebond).
- On cherche à savoir le nombre de rebonds nécessaire pour que la hauteur de la balle soit inférieure ou égale à 10 cm.

Compléter les lignes 4 et 6 de l'algorithme ci-dessous pour qu'il réponde au problème.

```
Variables: nombre,hauteur
1: DEBUT_ALGORITHME
2:     nombre_rebonds ← 0
3:     hauteur ← 300
4:     TANT_QUE (hauteur >10) FAIRE
5:         nombre_rebonds ← nombre_rebonds+1
6:         hauteur ← 0,9 * hauteur
7:         FIN_TANT_QUE
8:     AFFICHER nombre_rebonds
9: FIN_ALGORITHME
```

4) Fonctions

Les fonctions en algorithmique

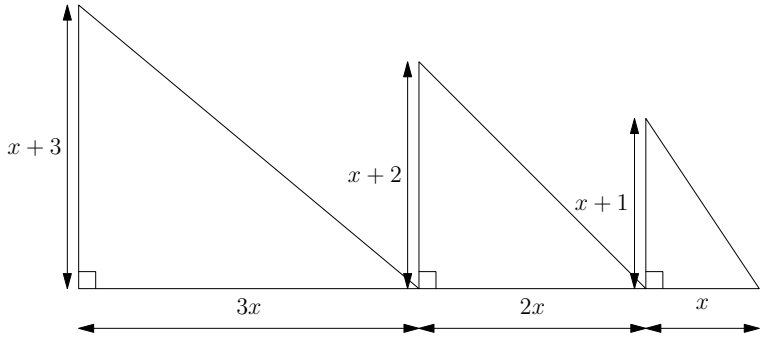
Pour éviter de répéter certaines instructions, on peut utiliser une fonction, c'est à dire une bloc d'instructions qui ne sera exécuté que quand on l'appelle. Une fonction dépend en général d'un ou plusieurs paramètres et retourne une valeur dépendant de ces paramètres.

```
FONCTION mafonction(paramètres...)
DEBUT_FONCTION
  ...
  RETOURNER...
FIN_FONCTION
```

Utiliser une fonction en algorithmique n'a d'intérêt que si on l'appelle plusieurs fois.

► Exercice n°16

On cherche à élaborer un algorithme qui donne en fonction de x la somme des hypoténuses des trois triangles rectangles de la figure ci-dessous :



Pour cela, on va utiliser une fonction `hypo` qui renvoie l'hypoténuse d'un triangle rectangle dont les côtés de l'angle droit sont `a` et `b`.

Compléter les lignes 3 et 7 pour que l'algorithme réponde au problème posé :

```
Variables: x
1: FONCTION hypo(a,b)
2: DEBUT_FONCTION
3:     Retourner √(a²+b²)
4: FIN_FONCTION
5: DEBUT_ALGORITHME
6:     Lire x
7:     AFFICHER hypo(3*x,x+3)+hypo(2*x,x+2)+hypo(x,x+1)
8: FIN_ALGORITHME
```

2. Initiation à python

1) Introduction

Langage informatique

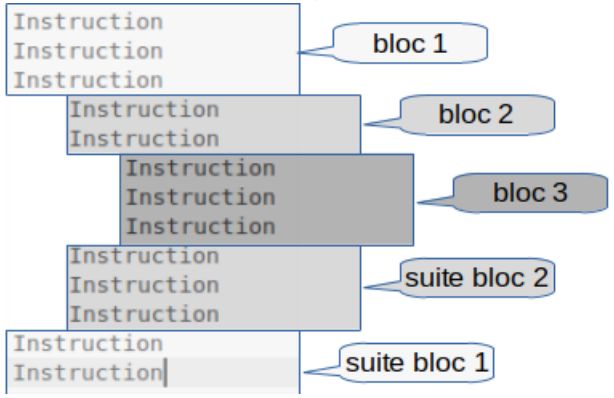
Un langage informatique permet de traduire un algorithme en une série d'instructions compréhensibles et exécutables par un ordinateur. Pour cela, les instructions écrites dans le langage informatique doivent respecter des spécificités et notamment une syntaxe plus précise et plus compliquée que celle d'un langage naturel.

Quelques principes de base à propos du langage python

- Par défaut, le langage python n'inclut pas bon nombre de fonctions mathématiques. C'est pour cela que certains scripts python commenceront par l'instruction suivante qui indique à python de charger les fonctions mathématiques standard :

```
from math import*
```

- Les débuts et fins de bloc d'instructions se marquent en langage python par un décalage horizontal (appelé indentation) et pas par des instructions particulières. Il faut donc faire très attention en python à ce que les instructions d'un même bloc soient écrites avec le même décalage horizontal par rapport à ce qui précède et suit ce bloc (on dit que ces instructions doivent avoir la même indentation).



2) Équivalences entre langage naturel et python

Affectation d'une valeur à une variable x

Langage naturel	Équivalent en python
x ← valeur	x = valeur

Lire la valeur d'une variable x entrée au clavier

Si x est un entier (et si l'algorithme ne manipule que des entiers) :

Langage naturel	Équivalent en python
LIRE x	x = int(input("x?"))

Si x est un réel :

Langage naturel	Équivalent en python
LIRE x	x = float(input("x?"))

Si on a besoin d'entrer une valeur résultant d'un calcul mathématique (fraction, racine carrée...) :

Langage naturel	Équivalent en python
LIRE x	from math import* x = eval(input("x?"))

Afficher la valeur d'une variable x

Langage naturel	Équivalent en python
AFFICHER x	print(x)

Afficher un message

Langage naturel	Équivalent en python
AFFICHER "Message"	print("message")

Les opérations mathématiques de base

Langage naturel	Équivalent en python
resultat ← a+b resultat ← a-b resultat ← $a \times b$ resultat ← $\frac{a}{b}$ resultat ← a^2 resultat ← $\sqrt{a}$	from math import* resultat=a+b resultat=a-b resultat=a*b resultat=a/b resultat=a**2 resultat=sqrt(a)

Pour a et b entiers	Équivalent en python
reste de la division euclidienne de a par b	a%b
quotient de la division euclidienne de a par b	a//b

► Exercice n°17

Compléter le code du script python ci-dessous pour qu'il corresponde à l'algorithme en langage naturel.

Langage naturel	Équivalent en python
LIRE distance LIRE temps vitesse ← $\frac{\text{distance}}{\text{temps}}$ AFFICHER vitesse	distance=float(input("distance?")) temps=float(input("temps?")) vitesse=distance/temps print(vitesse)

► Exercice n°18

Compléter le script python ci-dessous pour qu'il affiche la valeur de la diagonale d'un rectangle après avoir entré sa largeur et sa longueur :

Script python
from math import* largeur=float(input("largeur?")) longueur=float(input("longueur?")) diagonale=sqrt(largeur**2+longueur**2) print(diagonale)

Instructions conditionnelles

• Avec un SI...ALORS

Langage naturel

Instructions précédant le bloc si...alors
SI *condition* ALORS
  Instructions exécutées si la condition est vérifiée
  .....
  FIN_SI
Instructions suivant le bloc si...alors

Équivalent en python

Le : est indispensable
Instructions précédant le bloc si...alors
if *condition*:
  Instructions exécutées si la condition est vérifiée
  .....
Instructions suivant le bloc si...alors
Toutes les instructions décalées (indentées) qui suivent la ligne du if seront exécutées si la condition est vérifiée

► Exemple :

Langage naturel	Équivalent en python

LIRE x
SI x>0 ALORS
  AFFICHER "x est strictement positif"
  FIN_SI
AFFICHER "son opposé est"
AFFICHER -x
	x=float(input("x?"))
	if x>0:
	  print("x est strictement positif")
	print("son opposé est",-x)

• Avec un SI...ALORS...SINON

Langage naturel

Instructions précédant le bloc si...alors...sinon
SI *condition* ALORS
  Instructions exécutées si la condition est vérifiée
  .....
SINON
  Instructions exécutées si la condition n'est pas vérifiée
  .....
  FIN_SI
Instructions suivant le bloc si...alors...sinon

Équivalent en python

Le : est indispensable
Toutes les instructions décalées (indentées) qui suivent la ligne du if seront exécutées si la condition est vérifiée
Instructions précédant le bloc si...alors...sinon
if *condition*:
  Instructions exécutées si la condition est vérifiée
  .....
else:
  Instructions exécutées si la condition n'est pas vérifiée
  .....
Instructions suivant le bloc si...alors...sinon
Le : est indispensable
Toutes les instructions décalées (indentées) qui suivent la ligne du else seront exécutées si la condition n'est pas vérifiée

• Syntaxe python pour les conditions

Situation	Syntaxe python correspondante
Si a égal à b (a et b entiers)	if a==b:
Si a est différent de b (a et b entiers)	if a!=b:
Si a est strictement supérieur à b	if a>b:
Si a est strictement inférieur à b	if a<b:
Si a est inférieur ou égal à b	if a<=b:
Si a est supérieur ou égal à b	if a>=b:

Cas particulier de la comparaison de deux réels de type float (qui ne sont pas représentés de façon exacte en informatique) :

Situation mathématique	Syntaxe pour python (>=3.5)
Si a égal à b (a et b flottants)	from math import*
Si a est différent de b (a et b flottants)	if isclose(a,b):
	if not(isclose(a,b)):

► Exercice n°19

Compléter le code du script python ci-dessous pour qu'il corresponde à l'algorithme en langage naturel.

Langage naturel	Équivalent en python

LIRE age
SI age>=18 ALORS
  AFFICHER "majeur"
  FIN_SI
	age=int(input("age?"))
	if age>=18:
	  print("majeur")

► Exercice n°20

Parmi les quatre scripts python ci-dessous, déterminer le seul valide indiquant le plus grand de deux entiers. script 3

Script 1	Script 2

a=int(input("a?"))
b=int(input("b?"))
if a>b
  print("le plus grand est a")
else:
  print("le plus grand est b")
	a=int(input("a?"))
	b=int(input("b?"))
	if a>b:
	  print("le plus grand est b")
	else:
	  print("le plus grand est a")

Script 3	**Script 4**

a=int(input("a?"))
b=int(input("b?"))
if a>b:
  print("le plus grand est a")
else:
  print("le plus grand est b")
	a=int(input("a?"))
	b=int(input("b?"))
	if a>b:
	  print("le plus grand est a")
	else:
	print("le plus grand est b")

• Opérateurs ET/OU/CONTRAIRE dans les conditions

Situation	Syntaxe python correspondante
Si *condition1* OU *condition2*	if *condition1* or *condition2*:
Si *condition1* ET *condition2*	if *condition1* and *condition2*:
Si CONTRAIRE(*condition*)	if not(*condition*):

► **Remarque :** or, and et not doivent être en minuscules

► **Exemple :**

Langage naturel	Équivalent en python

LIRE x
SI x>0 ET X<1 ALORS
  AFFICHER "x est dans]0;1["
  FIN_SI
	x=float(input("x?"))
	if x>0 and x<1:
	  print("x est dans]0;1[")

► Exercice n°21

On considère l'expression $A(x) = \dfrac{1}{(x-1)\sqrt{x}}$ où x est un réel.

Compléter le code du script python ci-dessous pour qu'il corresponde à l'algorithme en langage naturel.

Langage naturel	Équivalent en python

LIRE x
SI x>0 ET x≠1 ALORS
  AFFICHER "A(x) existe"
SINON
  AFFICHER "A(x) n'existe pas"
  FIN_SI
	from math import*
	x=float(input("x?"))
	if x>0 and x!=1:
	  print("A(x) existe")
	else:
	  print("A(x) n'existe pas")

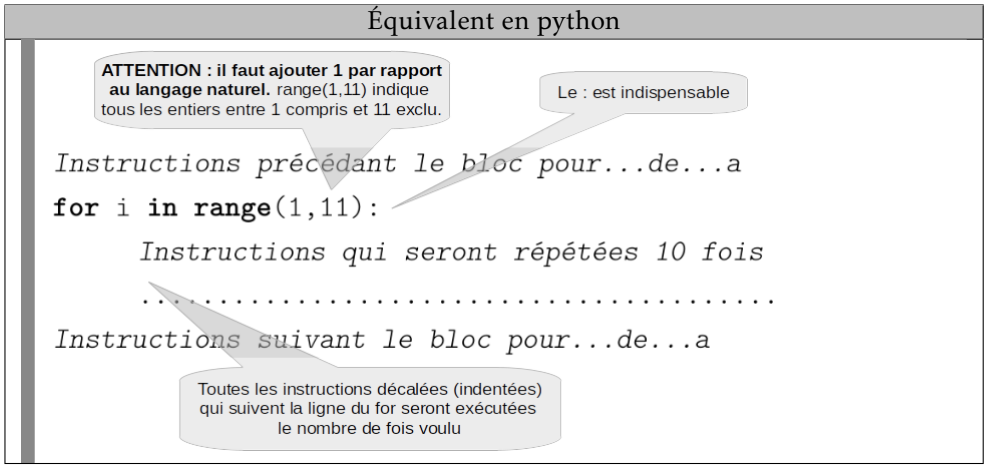
Boucles POUR...DE...A

• Rappel : ces boucles permettent de répéter des instructions un certain nombre (connu par avance) de fois.

Langage naturel

Instructions précédant le bloc pour...de...a
POUR i ALLANT_DE 1 A 10
  Instructions qui seront répétées 10 fois
  .....
  FIN_POUR
Instructions suivant le bloc pour...de...a

Équivalent en python



► **Exemple :**

Langage naturel	Équivalent en python

POUR i ALLANT_DE 1 A 10
  AFFICHER 7×i
  FIN_POUR
AFFICHER "script terminé"
	for i in range(1,11):
	  print(7*i)
	print("script terminé")

► **Exercice n°22**

Compléter le code du script python ci-dessous pour qu'il corresponde à l'algorithme en langage naturel dont le but est de calculer la somme $1+2+3+\cdots+99+100$.

Langage naturel	Équivalent en python

somme ← 0
POUR i ALLANT_DE 1 A 100
  somme ← somme+i
  FIN_POUR
AFFICHER somme
	somme=0
	for i in range(1,101):
	  somme=somme+i
	print(somme)

► **Exercice n°23**

Parmi les quatre scripts python ci-dessous, déterminer le seul valide qui permet de calculer la somme $\sqrt{1}+\sqrt{2}+\sqrt{3}+\cdots+\sqrt{49}+\sqrt{50}$. script 2

Script 1	Script 2

from math import*
somme=0
for i in range(1,51)
  somme=somme+sqrt(i)
print(somme)
	from math import*
	somme=0
	for i in range(1,51):
	  somme=somme+sqrt(i)
	print(somme)

Script 3	Script 4
from math import* somme=0 for i in range(1,50): somme=somme+sqrt(i) print(somme)	from math import* somme=0 for i in range(1,51): somme=somme+sqrt(i) print(somme)

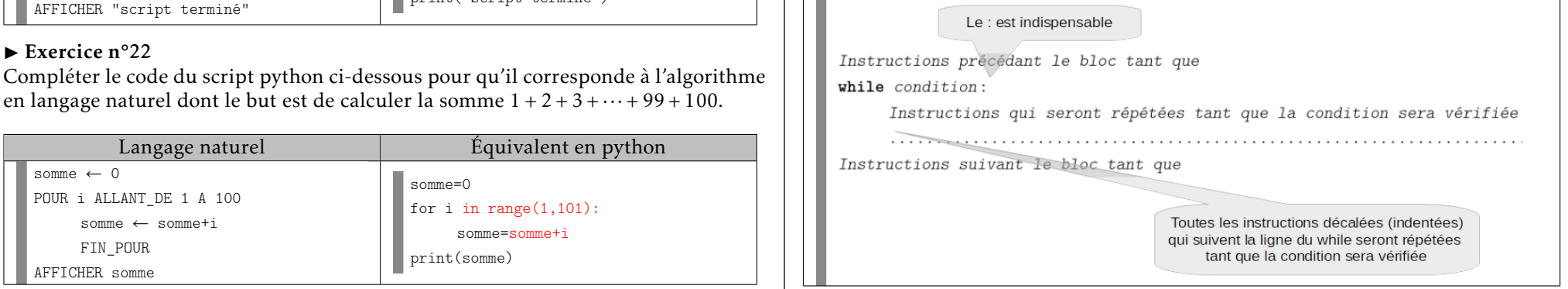
Boucles TANT QUE

• Rappel : ces boucles permettent de répéter des instructions tant qu'une certaine condition est vérifiée. La condition en question s'exprime en python de la même façon qu'avec un if.

Langage naturel

```
Instructions précédant le bloc tant que
TANT_QUE condition FAIRE
    Instructions qui seront répétées tant que la condition sera vérifiée
    .....
    FIN_TANT_QUE
Instructions suivant le bloc tant que
```

Équivalent en python



► **Exemple :**

Script permettant de déterminer le premier entier dont le cube dépasse 5000.

Langage naturel	Équivalent en python

n ← 0
TANT_QUE $n^3 < 5000$ FAIRE
  n ← n+1
  FIN_TANT_QUE
AFFICHER n
	n=0
	while n**3<5000:
	  n=n+1
	print(n)

► Exercice n°24

Compléter le code du script python ci-dessous pour qu'il corresponde à l'algorithme en langage naturel dont le but est de déterminer le premier entier n tel que le produit $1 \times 2 \times 3 \times \cdots \times n$ dépasse 10000 .

Langage naturel	Équivalent en python

n ← 1
produit ← 1
TANT_QUE produit<10000 FAIRE
  n ← n+1
  produit ← produit × n
  FIN_TANT_QUE
AFFICHER n
	n=1
	produit=1
	while produit<10000:
	  n=n+1
	  produit=produit*n
	print(n)

► Exercice n°25

Parmi les quatre scripts python ci-dessous, déterminer le seul valide indiquant le plus petit entier n tel que $0,9^n \leqslant 0.001$. script 4

Script 1	Script 2
n=0 while 0.9**n<=0.001: n=n+1 print(n)	n=0 while 0.9**n>0.001: n=n+1 print(n)
Script 3	**Script 4**

n=0
while 0.9**n<=0.001
  n=n+1
print(n)
	n=0
	while 0.9**n>0.001:
	  n=n+1
	print(n)

Les fonctions en programmation

• Rappel : une fonction en programmation est un bloc d'instructions qui ne sera exécuté que quand on l'appelle. Une fonction dépend en général d'un ou plusieurs paramètres et retourne une valeur dépendant de ces paramètres. L'utilisation d'une fonction n'a en général d'intérêt que si on l'appelle plusieurs fois.

Langage naturel

FONCTION mafonction(*paramètres séparés par des virgules*)
DEBUT_FONCTION
  Instructions
  RETOURNER...
FIN_FONCTION

Équivalent en python

Le : est indispensable
def mafonction(*paramètres séparés par des virgules*):
  Instructions
  return...
Toutes les instructions définissant la fonction doivent être décalées (indentées)

► Exemple :

Un cycliste et un piéton se déplacent à une vitesse respective de 18 $km \cdot h^{-1}$ et 6 $km \cdot h^{-1}$. Ce script donne le nombre de kilomètres parcourus par le cycliste et le piéton durant un même temps t (en heures).

Langage naturel	Équivalent en python

FONCTION distance(vitesse,temps)
DEBUT_FONCTION
  RETOURNER vitesse × temps
FIN_FONCTION
 	
DEBUT_ALGORITHME
LIRE t
AFFICHER distance(42,t)
AFFICHER distance(6,t)
FIN_ALGORITHME
	def distance(vitesse,temps):
	  return vitesse*temps
	
	t=float(input("t?"))
	print(distance(42,t))
	print(distance(6,t))

► Exercice n°26

La formule de conversion entre la valeur d'une température en degrés farenheit et sa valeur en degrés celsius est la suivante :

$$\text{degrés Celsius} = \frac{5 \times (\text{degrés Fahrenheit}) - 160}{9}$$

Compléter le code du script python ci-dessous pour qu'il donne la valeur en degrés celsius d'une température de 100 et de 200 degrés farenheit.

Script python
def celsius(farenheit):   return (5*farenheit-160)/9 print(celsius(100)) print(celsius(200))

3) Application

► Exercice n°27
Compléter le code du script python ci-dessous pour qu'il corresponde à l'algorithme en langage naturel.

Langage naturel	Équivalent en python
POUR i ALLANT_DE 1 A 100 AFFICHER i^3 FIN_POUR	for i in range(1,101): print(i**3)

► Exercice n°28
Que fait ce script python ?

Script python
```
from math import*
i=0
while i*sqrt(i)<500:
      i=i+1
print(i)
```

Réponse : Il indique le plus petit entier positif i tel que $i*\sqrt{i} \geqslant 500$.
.....

► Exercice n°29
La valeur d'une voiture était de 20 000 euros en 2018. On suppose qu'elle est multipliée par 0,9 chaque année. Compléter le script python ci-dessous pour qu'il indique la première année à partir de laquelle la valeur de la voiture sera inférieure ou égale à 12 000 euros.

Script python
```
annee=2018
valeur=20000
while valeur>12000:
       valeur=valeur*0.9
       annee=annee+1
print(annee)
```

► Exercice n°30
Un ticket de tramway coûte 1 euro sans abonnement. Avec un abonnement annuel de 30 euros, le ticket ne coûte plus que 0,75 euros. Déterminer parmi les quatre scripts python ci-dessous le seul valide qui affiche, après avoir entré le nombre de tickets achetés n, si prendre un abonnement est rentable : script 1

Script 1
```
n=int(input("n?"))
if 30+0.75*n<n:
       print("abonnement rentable")
else:
       print("abonnement pas rentable")
```

Script 2
```
n=int(input("n?"))
if 30+0.75*n<n:
       print("abonnement rentable")
else
       print("abonnement pas rentable")
```

Script 3
```
n=int(input("n?"))
if 30+0.75*n<n:
       print("abonnement rentable")
       else:
       print("abonnement pas rentable")
```

Script 4
```
n=int(input("n?"))
if 30+0.75*n>n:
       print("abonnement rentable")
else:
       print("abonnement pas rentable")
```

► Exercice n°31
Durant une année un individu a acheté n DVD valant 15 euros pièce et p livres valant 22 euros pièce. Compléter le script python ci-dessous pour qu'il indique si cet individu a dépensé plus de 300 euros en dvd et livres ou non :

Script python
```
def depense_totale(n,p):
       return 15*n+22*p

n=int(input("nombre de dvd?"))
p=int(input("nombre de livres?"))
if depense_totale(n,p)>300:
       print("plus de 300 euros")
else:
       print("pas plus de 300 euros")
```

► Exercice n°32
Un automobiliste roule à 90 $km \cdot h^{-1}$ pendant 2 heures, puis roule à 120 $km \cdot h^{-1}$. Compléter le script python ci-dessous pour qu'il indique le nombre de kilomètres parcourus en fonction du temps de parcours exprimé en heures :

Script python
```
def distance(temps):
       if temps<=2:
            return 90*temps
       else:
            return 180+120*(temps-2)

t=float(input("temps?"))
print (distance(t))
```

4) Autres fonctionnalités utiles de python

Chaines de caractères

On peut affecter à une variable une chaine de caractères en l'encadrant simplement par des guillemets. Exemple : `a="blabla"`

Situation (a et b : chaines)	Syntaxe python
mettre b au bout de a	a+b
connaitre la longueur de a	len(a)
récupérer le $(i+1)^{\text{ième}}$ caractère de a	a[i]

Exemple :

Code python	Résultat
a="bon" b="jour" print(a+b) print(len(a)) print(a[0]) print(b[3])	bonjour 3 b r

Les listes

On peut affecter à une variable une liste de nombres (ou de chaines de caractères) en les encadrant par des crochets et en les séparant par des virgules. Exemple : `a=[1,3,5,7]`

► **Remarque :** en python, on commence à compter les éléments d'une liste à partir de la position 0. Par exemple, si `a=[1,3,5,7]` :

- le premier élément de la liste est `a[0]` (position 0) qui vaut donc 1 ;
- le deuxième élément de la liste est `a[1]` (position 1) qui vaut donc 3 ;
- etc.
- le dernier terme de la liste est `a[3]` (position 3) qui vaut donc 7 ;

Donc le terme à la position `i` d'une liste correspond à son $(i+1)^{\text{ième}}$ terme

Situation (a : liste)	Syntaxe python
connaitre le nombre d'éléments de a	len(a)
récupérer le terme de a à la position i	a[i]
ajouter l'élément e à la liste a	a.append(e)
supprimer le terme de a à la position i	del a[i]
insère l'élément e à la position i	a.insert(i,e)

► **Exemple :**

Code python	Résultat
a=[1,3,5,7] print(len(a)) print(a[3]) a.append(9) print(a) del a[1] print(a) a.insert(1,12) print(a)	4 7 [1, 3, 5, 7, 9] [1, 5, 7, 9] [1, 12, 5, 7, 9]

Nombres pseudo aléatoires avec python

En ajoutant au début d'un script l'instruction suivante, on indique à python de charger des outils qui permettent de simuler le hasard :

```
from random import*
```

L'instruction `randint(a,b)` (avec `a` et `b` entiers) permet alors de tirer au hasard un entier compris entre `a` et `b` (`a` et `b` inclus).

5) Tester des scripts python en ligne

Il est possible d'exécuter des scripts python depuis un navigateur sans aucune installation, ni inscription, ni connexion préalable à l'adresse suivante :
https://www.xm1math.net/SNT/python_en_ligne.html

2

Représentation numérique et traitement des données

1. Le système binaire

• Pour compter, les êtres humains utilisent habituellement depuis quelques siècles le *système décimal* (ou *système de base 10*) qui comporte 10 chiffres que l'on compose pour pouvoir écrire n'importe quel entier.
Ainsi :

- Le nombre *sept cent cinquante-trois* s'écrit 753 en système décimal car $\boxed{7}\times 10^2+\boxed{5}\times 10+\boxed{3}$ donne bien sept cent cinquante-trois.
- Le nombre *huit mille quatre cent cinquante-six* s'écrit 8456 en système décimal car $\boxed{8}\times 10^3+\boxed{4}\times 10^2+\boxed{5}\times 10+\boxed{6}$ donne bien huit mille quatre cent cinquante-six.

• L'informatique utilise des courants électriques, des aimantations, des rayons lumineux... Chacun de ces phénomènes physiques ne met en jeu deux états possibles : tension nulle ou tension non nulle pour les circuits électroniques ; aimantation dans un sens ou dans l'autre pour les disques durs ; lumière ou absence de lumière pour la lecture optique des CD, DVD...Il suffit donc de deux chiffres pour traduire ces deux états. La représentation des données en informatique se base donc depuis le début sur un *système de base 2*, appelé *système binaire*, qui n'utilise que les chiffres 0 et 1.
Les chiffres (0 ou 1) qui composent un nombre écrit en système binaire s'appellent des *bits*.
Ainsi :

- Le nombre binaire 101 (comportant donc 3 bits) correspond au nombre décimal 5 car $\boxed{1}\times 2^2+\boxed{0}\times 2+\boxed{1}=5$
- Le nombre binaire 110101 (comportant donc 6 bits) correspond au nombre décimal 53 car $\boxed{1}\times 2^5+\boxed{1}\times 2^4+\boxed{0}\times 2^3+\boxed{1}\times 2^2+\boxed{0}\times 2+\boxed{1}=53$

▶ Exercice n°33

1. Donner l'écriture en système décimal du nombre binaire 110.
   Réponse : 6
2. Donner l'écriture en système décimal du nombre binaire 1110001.
   Réponse : 113
3. Donner l'écriture en système binaire du nombre 3.
   Réponse : 11
4. Donner l'écriture en système binaire du nombre 18.
   Réponse : 10010
5. Quel est le plus grand nombre, en système décimal, que l'on peut obtenir à partir d'un nombre binaire formé de 8 bits ?
   Réponse : 11111111 -> 255

. . . . . .

▶ Exercice n°34

1. Donner l'écriture en système décimal du nombre binaire 11.
   Réponse : 3
2. Donner l'écriture en système décimal du nombre binaire 111.
   Réponse : 7

3. Donner le résultat, en système décimal, de la somme et du produit des deux nombres précédents, puis convertir les résultats en écriture binaire :

   Somme en système décimal : 10

   Somme convertie en écriture binaire : 1010

   Produit en système décimal : 21

   Produit converti en écriture binaire : 10101

2. Les unités en informatique

L'octet

En informatique, l'unité d'information de base est un nombre binaire composé de 8 bits et appelé *octet* (*byte* en anglais).

A partir des octets s'est construit historiquement et de façon un peu confuse un double système d'unités basé l'un sur les traditionnelles puissances de 10 et l'autre sur des puissances de 2. Cela a conduit à introduire deux notations différentes pour les multiples d'octets (mais la confusion demeure souvent présente) :

Avec des puissances de 10

1 ko (kilooctet) : 1 000 octets (10^3)
1 Mo (mégaoctet) : 1 million d'octets (10^6)
1 Go (gigaoctet) : 1 milliard d'octets (10^9)

Avec des puissances de 2

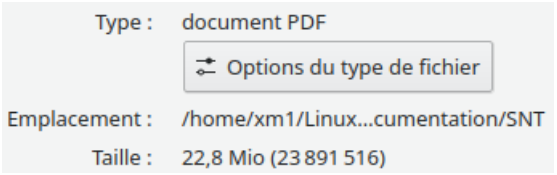
1 kio (kibioctet) : 1 024 octets (2^{10})
1 Mio (mébioctet) : 1 048 576 octets (2^{20})
1 Gio (gibioctet) : 1 073 741 824 octets (2^{30})

▶ Exercice n°35

1. Un fabricant annonce une capacité de stockage de 128 Go sur un smartphone. Combien de Gio cela représente-t-il ?

   Réponse : $128 \times 10^9/2^{30} = 119,2$

2. D'après la capture d'écran des propriétés d'un fichier ci-dessous, vérifier que la taille du fichier donnée en Mio est exacte et donner la taille en Mo.



Réponse : $23891516/2^{20} \approx 22,8$ Mio et $23891516/10^6 \approx 23,9$ Mo

▶ Exercice n°36

Compléter le script python ci-dessous pour que la fonction `conversion` retourne la taille en Kio et préciser ce qu'affichera alors le script une fois exécuté.

Script python

```
def conversion(tailleko):
    return tailleko/1.024

print(conversion(250)) ->244.140625
```

Adresse IP - Protocole TCP-IP

- Dès qu'un appareil se connecte à un réseau (local ou directement à Internet), il se voit attribuer un numéro unique appelé *adresse IP* qui permet de l'identifier de façon unique sur le réseau et de permettre ainsi les échanges de données entre tous les systèmes connectés au réseau (ordinateurs, serveurs, téléphones portables etc...)
  - L'adresse IP ne dépend pas du matériel : c'est un simple identifiant de réseau qui n'est pas forcément fixe. Sur Internet, elle permet cependant à n'importe quel site de connaitre le fournisseur d'accès (*FAI*), la région et le pays d'origine d'un internaute. Par contre, dans les pays démocratiques, il faut une décision de justice ou administrative pour qu'un fournisseur d'accès à internet dévoile le nom et les coordonnées d'un internaute à partir de son adresse IP. Un *VPN* (« virtual private network ») permet de masquer son adresse IP et sa localisation, mais cela ralentit la connexion internet et nécessite un prestataire qui fournisse un service de qualité et de confiance (souvent payant).
- Pour que deux systèmes connectés (ordinateurs, serveurs, etc.) sur internet puissent échanger des données et communiquer entre eux, ils doivent respecter un certain « langage » de communication appelé *TCP/IP* qui se composent :
  - du protocole *TCP* (pour « Transmission Control Protocol ») qui se charge de récupérer les données à envoyer, de les fractionner en paquets et de trouver le moyen de les envoyer au bon destinataire ;
  - du protocole *IP* (pour « Internet Protocol ») qui s'occupe de la localisation du destinataire à partir de son adresse IP lors de l'envoi des paquets TCP.

▶ Exercice n°37

1. Les adresses IP classiques (norme IPv4) sont composées de quatre nombres compris entre 0 et 255 séparés par un point. Exemple : 204.35.129.3.
   Combien d'octets cela représente-t-il ? 4 octets
   Combien d'adresses IP différentes cette norme IPv4 permet-elle d'utiliser ? 256^4 environ 4 milliards

2. Pour faire face à la pénurie d'adresses IPv4, une nouvelle norme, appelée *IPv6* a été établie afin de remplacer progressivement les anciennes adresses IPv4. Les adresses IPv6 sont basées sur 128 bits.
   Combien d'octets cela représente-t-il ? 16 octets
   Combien d'adresses IP différentes cette norme IPv6 permet-elle d'utiliser ? $256^{16} \approx 3 \times 10^{38}$

Adresse MAC

Les matériels connectés possèdent aussi un autre identifiant, *l'adresse MAC* (pour « Media Acces Control »), qui à la différence de l'adresse IP, sert à identifier de façon unique le matériel (sa carte réseau) et non pas une connexion. Cet identifiant est fixe et existe que le matériel soit connecté ou non. Il permet, entre autres, d'interdire ou d'autoriser l'accès de certains matériels au réseau.

▶ Exercice n°38

Une adresse MAC est codée sur l'équivalent de 6 octets. Combien d'adresses MAC différentes sont possibles ? $256^6 \approx 3 \times 10^{14}$

▶ Exercice n°39

À son domicile, le téléphone portable d'un individu est connecté à internet via le wifi de sa box internet. Quand il sort de son domicile et que la connexion à internet utilise le réseau 4G de son opérateur, quelle adresse change ? L'adresse IP ou l'adresse MAC ? l'adresse IP

▶ Exercice n°40

Les débits de transfert de données sont souvent exprimés en MégaBits (1 million de bits) par seconde. Un individu souhaite transférer un morceau de musique de 7,5 Mio via une connexion bluetooth au débit de 2 MégaBits par seconde. Combien de temps cela prendra-t-il ?

Réponse : 7,5 Mio -> $7,5 \times 2^20 \times 8 = 62914560 bits \approx 63$ MégaBits. Il faut environ 31,5 s

3. Représentation numérique des caractères et des textes

Pour représenter un caractère, on lui attribue un nombre. Un texte simple devient alors une liste de nombres que les systèmes numériques peuvent alors traiter en données binaire. Il existe plusieurs manières de coder numériquement un texte : c'est ce qu'on appelle définir l' *encodage* du texte.

Le code ASCII

Un des plus anciens et classiques encodage de texte est l'*ASCII* (« American Standard Code for Information Interchange ») qui permet de coder jusqu'à 128 caractères standards (ne nécessitant donc que 7 bits, même si dans les faits on utilise un octet complet pour des raisons pratiques). Extrait du code *ASCII* :

Caractères		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/	0	1	2
Code	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50

Caractères	3	4	5	6	7	8	9	:	;	¡	=	¿	?	@	A	B	C	D	E
Code	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69

Caractères	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Code	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88

Caractères	Y	Z	[	\	]	^	_	`	a	b	c	d	e	f	g	h	i	j	k
Code	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107

Caractères	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	—	}	~
Code	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126

Ainsi le caractère A est codé par le nombre 65 qui donne en binaire 1000001.

▶ Exercice n°41

1. Donner la liste des trois nombres permettant de coder en *ASCII* le texte *SNT*.

   Réponse : 83 78 84

2. Donner la représentation en binaire de ces 3 nombres

   Réponse : 1010011

   1001110

   1010100

▶ Exercice n°42

En comptant un octet par caractère, combien de caractères peut contenir un texte pesant 56 kio encodé en ASCII ?

Réponse : $56 \times 1024 = 57344$

► **Remarques :**

- En python, l'instruction `ord('A')` permet de retrouver le code *ASCII* du caractère A et inversement, l'instruction `chr(65)` permet de retrouver le caractère A à partir de son code *ASCII*.
- L'encodage *ASCII* ne permettant pas de coder les lettres accentuées et tous les caractères spéciaux des différentes langues, il a été créé ensuite d'autres encodages adaptés à chaque type de langue (l'encodage *ISO-8859-1* pour le français, par exemple), mais on se retrouvait alors avec une multitude d'encodages que les systèmes numériques devaient pouvoir traiter. Pour éviter cet inconvénient, il a été décidé dans les années 90, de définir un encodage plus universel permettant de contenir les caractères de toutes les langues tout en étant compatible avec l'*ASCII* : l'*unicode* (dont la variante la plus connue et utilisée, notamment sur le web, est l'*UTF-8* et qui a vocation à devenir le standard).

4. Représentation matricielle

1) Généralités

Les matrices

- Une matrice est un tableau de nombres.
  Exemple : $A = \begin{pmatrix} 1 & 2 \\ 3 & -5 \\ 4 & 1 \end{pmatrix}$
- Si la matrice comporte n lignes et p colonnes, on dit que la matrice est de dimension n x p
- Les nombres du tableau sont appelés coefficients de la matrice.

► **Remarques :**

- Une matrice ne comportant qu'une seule ligne est appelée matrice ligne
- Une matrice ne comportant qu'une seule colonne est appelée matrice colonne
- Une matrice comportant autant de lignes que de colonnes est appelée matrice carrée

2) Somme, différence et multiplication par un nombre

- Pour ajouter deux matrices de même dimension, il suffit d'ajouter les coefficients. Exemples :
  · si $A = \begin{pmatrix} 1 & 4 \\ 2 & -1 \\ -3 & 2 \end{pmatrix}$ et $B = \begin{pmatrix} 5 & -2 \\ 4 & 5 \\ 1 & -3 \end{pmatrix}$ alors $A + B = \begin{pmatrix} 6 & 2 \\ 6 & 4 \\ -2 & -1 \end{pmatrix}$
  · si $A = \begin{pmatrix} 1 & -2 & 4 \\ -7 & 8 & 2 \end{pmatrix}$ et $B = \begin{pmatrix} 5 & 4 & -3 \\ 1 & -2 & 5 \end{pmatrix}$ alors $A + B = \begin{pmatrix} 6 & 2 & 1 \\ -6 & 6 & 7 \end{pmatrix}$
- Pour soustraire deux matrices de même dimension, il suffit de soustraire les coefficients. Exemples :
  · si $A = \begin{pmatrix} 5 & 1 \\ -2 & 4 \end{pmatrix}$ et $B = \begin{pmatrix} 2 & 6 \\ -1 & -3 \end{pmatrix}$ alors $A - B = \begin{pmatrix} 3 & -5 \\ -1 & 7 \end{pmatrix}$
  · si $A = \begin{pmatrix} 10 & 2 & -5 \\ 8 & 4 & 1 \end{pmatrix}$ et $B = \begin{pmatrix} 6 & -3 & 1 \\ 0 & 1 & 10 \end{pmatrix}$ alors $A - B = \begin{pmatrix} 4 & 5 & -6 \\ 8 & 3 & -9 \end{pmatrix}$
- Pour multiplier une matrice par un nombre, il suffit de multiplier les coefficients de la matrice par ce nombre. Exemple :
  · si $A = \begin{pmatrix} 6 & 14 \\ -5 & 1 \\ 3 & -5 \end{pmatrix}$ alors $3A = \begin{pmatrix} 18 & 42 \\ -15 & 3 \\ 9 & -15 \end{pmatrix}$

3) Multiplication de deux matrices

Multiplication d'une matrice ligne par une matrice colonne

Multiplier une matrice ligne par une matrice colonne ayant le même nombre de coefficients donne un nombre que l'on obtient en appliquant le principe suivant (appelé « mouvement tournant ») :

$$\begin{pmatrix} a & b \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = ax + by \quad ; \quad \begin{pmatrix} a & b & c \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = ax + by + cz$$

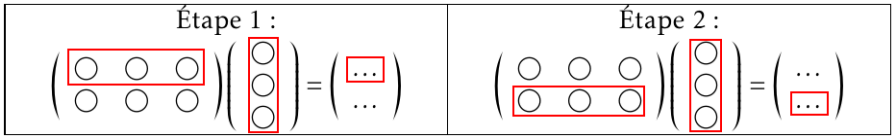
► **Exemples :**

- $\begin{pmatrix} 4 & 5 \end{pmatrix}\begin{pmatrix} 2 \\ 3 \end{pmatrix} = 23$
- $\begin{pmatrix} 3 & 2 & 4 \end{pmatrix}\begin{pmatrix} 5 \\ -1 \\ 2 \end{pmatrix} = 21$

► **Remarque :** la multiplication de deux matrices n'est possible que si les dimensions des matrices permettent le « mouvement tournant ».

Multiplication d'une matrice par une matrice colonne

On multiplie chaque ligne de la première matrice par la matrice colonne comme au paragraphe précédent et on place les résultats les uns sous les autres (dans une matrice colonne).

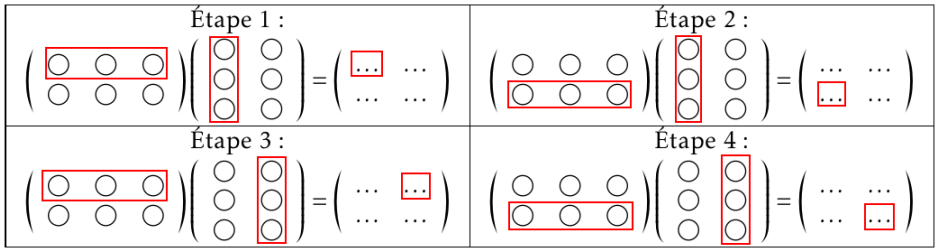


► **Exemples :**

- $\begin{pmatrix} 6 & 5 & 7 \\ 1 & -5 & 2 \end{pmatrix}\begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 55 \\ -5 \end{pmatrix}$
- $\begin{pmatrix} 2 & -3 \\ -1 & -5 \\ 7 & -4 \end{pmatrix}\begin{pmatrix} 2 \\ 5 \end{pmatrix} = \begin{pmatrix} -11 \\ -27 \\ -6 \end{pmatrix}$
- $\begin{pmatrix} -2 & 1 & 7 \\ 8 & -3 & 4 \end{pmatrix}\begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 25 \\ -2 \end{pmatrix}$
- $\begin{pmatrix} 1 & 0 & 2 \\ 3 & -1 & 4 \\ -2 & 1 & 0 \end{pmatrix}\begin{pmatrix} 8 \\ 1 \\ -2 \end{pmatrix} = \begin{pmatrix} 4 \\ 15 \\ -15 \end{pmatrix}$

Cas général

Le produit d'une matrice A par une matrice B lorsqu'elle est possible est la matrice notée AB obtenue en multipliant A par chaque colonne de B.



► **Exemples :**

- $\begin{pmatrix} 1 & 5 & -2 \\ 3 & 2 & 4 \end{pmatrix}\begin{pmatrix} 4 & 1 \\ 7 & 5 \\ 8 & 8 \end{pmatrix} = \begin{pmatrix} 23 & 10 \\ 58 & 45 \end{pmatrix}$
- $\begin{pmatrix} 1 & 4 \\ 2 & 0 \\ -3 & 1 \end{pmatrix}\begin{pmatrix} 5 & 2 \\ 3 & 1 \end{pmatrix} = \begin{pmatrix} 17 & 6 \\ 10 & 4 \\ -12 & -5 \end{pmatrix}$
- $\begin{pmatrix} 2 & -1 \\ -3 & 4 \end{pmatrix}\begin{pmatrix} 3 & 0 & -2 \\ 1 & 3 & 5 \end{pmatrix} = \begin{pmatrix} 5 & -3 & -9 \\ -5 & 12 & 26 \end{pmatrix}$
- $\begin{pmatrix} 1 & 5 \\ 3 & -1 \end{pmatrix}\begin{pmatrix} 3 & 2 \\ 4 & 6 \end{pmatrix} = \begin{pmatrix} 23 & 32 \\ 5 & 0 \end{pmatrix}$
- $\begin{pmatrix} 1 & -2 & 1 \\ 0 & 4 & 1 \\ 2 & 1 & -3 \end{pmatrix}\begin{pmatrix} 2 & 1 & -4 \\ 5 & 8 & 1 \\ -3 & 1 & 0 \end{pmatrix} = \begin{pmatrix} -11 & -14 & -6 \\ 17 & 33 & 4 \\ 18 & 7 & -7 \end{pmatrix}$

► **Remarque :** lorsque les dimensions le permettent, les opérations sur les matrices respectent les règles standards de distributivité sauf qu'en général, $AB \neq BA$.

Puissance d'une matrice carrée

Si A est une matrice carrée, le produit AA est noté A^2, le produit AAA est noté A^3, etc...

► **Exemple :**

- Si $A = \begin{pmatrix} 2 & 0 \\ 1 & -3 \end{pmatrix}$ alors $A^2 = AA = \begin{pmatrix} 2 & 0 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 2 & 0 \\ 1 & -3 \end{pmatrix} = \begin{pmatrix} 4 & 0 \\ -1 & 9 \end{pmatrix}$
- et $A^3 = AA^2 = \begin{pmatrix} 2 & 0 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 4 & 0 \\ -1 & 9 \end{pmatrix} = \begin{pmatrix} 8 & 0 \\ 7 & -27 \end{pmatrix}$

► **Exercice n°43**

Compléter la matrice manquante dans les cas suivants :

a) $\begin{pmatrix} 1 & 2 & 6 \\ 9 & -1 & 2 \end{pmatrix} + \begin{pmatrix} 5 & 6 & -3 \\ 1 & -1 & 7 \end{pmatrix} = \begin{pmatrix} 6 & 8 & 3 \\ 10 & -2 & 9 \end{pmatrix}$

b) $\begin{pmatrix} 9 & 1 & 2 \\ 1 & -4 & 1 \\ 2 & 1 & 7 \end{pmatrix} - \begin{pmatrix} 0 & -1 & 4 \\ 6 & -3 & 1 \\ 1 & -1 & 0 \end{pmatrix} = \begin{pmatrix} 9 & 2 & -2 \\ -5 & -1 & 0 \\ 1 & 2 & 7 \end{pmatrix}$

c) $3\begin{pmatrix} 0 & -4 & 2 \\ 1 & 6 & 0 \\ 5 & -3 & 8 \end{pmatrix} - 2\begin{pmatrix} 1 & -1 & 4 \\ 7 & -5 & -2 \\ 8 & 3 & 7 \end{pmatrix} = \begin{pmatrix} -2 & -10 & -2 \\ -11 & 28 & 4 \\ -1 & -15 & 10 \end{pmatrix}$

d) $\begin{pmatrix} -2 & 1 \\ 2 & 0 \\ -2 & 3 \end{pmatrix} + \begin{pmatrix} 5 & 6 \\ -4 & 1 \\ 7 & -6 \end{pmatrix} = \begin{pmatrix} 3 & 7 \\ -2 & 1 \\ 5 & -3 \end{pmatrix}$

e) $2\begin{pmatrix} 3/2 & 1/2 \\ 0 & 11/2 \end{pmatrix} - \begin{pmatrix} 2 & 4 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & -3 \\ -1 & 9 \end{pmatrix}$

► Exercice n°44

a) Si $A = \begin{pmatrix} 1 & 2 & 0 \\ 0 & -1 & 2 \end{pmatrix}$ et $B = \begin{pmatrix} -2 & 1 \\ 2 & 0 \\ -2 & 3 \end{pmatrix}$ alors $AB = \begin{pmatrix} 2 & 1 \\ -6 & 6 \end{pmatrix}$

b) Si $A = \begin{pmatrix} 2 & 4 \\ 1 & 2 \end{pmatrix}$ et $B = \begin{pmatrix} 2 & -6 \\ -1 & 3 \end{pmatrix}$ alors $AB = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

c) Si $A = \begin{pmatrix} 3 & -4 \\ 2 & 1 \end{pmatrix}$ et $B = \begin{pmatrix} 0 & -4 & 1 \\ 5 & -2 & 3 \end{pmatrix}$ alors $AB = \begin{pmatrix} -20 & -4 & -9 \\ 5 & -10 & 5 \end{pmatrix}$

► Exercice n°45

Soit $A = \begin{pmatrix} 2 & -5 & 1 \\ 1 & -2 & 4 \end{pmatrix}$ et $B = \begin{pmatrix} 1 & 3 \\ -1 & 2 \\ 4 & 0 \end{pmatrix}$.

On a $AB = \begin{pmatrix} 2 & -5 & 1 \\ 1 & -2 & 4 \end{pmatrix}\begin{pmatrix} 1 & 3 \\ -1 & 2 \\ 4 & 0 \end{pmatrix} = \begin{pmatrix} 11 & -4 \\ 19 & -1 \end{pmatrix}$

Et $BA = \begin{pmatrix} 1 & 3 \\ -1 & 2 \\ 4 & 0 \end{pmatrix}\begin{pmatrix} 2 & -5 & 1 \\ 1 & -2 & 4 \end{pmatrix} = \begin{pmatrix} 5 & -11 & 13 \\ 0 & 1 & 7 \\ 8 & -20 & 4 \end{pmatrix}$

► Exercice n°46

Soit $A = \begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix}$, $B = \begin{pmatrix} 5 & -1 \\ -3 & 2 \end{pmatrix}$ et $C = \begin{pmatrix} 3 & 0 \\ -3 & 2 \end{pmatrix}$.

On a $AB = \begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} 5 & -1 \\ -3 & 2 \end{pmatrix} = \begin{pmatrix} -3 & 2 \\ -6 & 4 \end{pmatrix}$

Et $AC = \begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix}\begin{pmatrix} 3 & 0 \\ -3 & 2 \end{pmatrix} = \begin{pmatrix} -3 & 2 \\ -6 & 4 \end{pmatrix}$

► Exercice n°47

Soit $A = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix}$. On a $A^2 = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 5 & 2 & 1 \\ 2 & 2 & 2 \\ 1 & 2 & 5 \end{pmatrix}$

Et $A^3 = AA^2 = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix}\begin{pmatrix} 5 & 2 & 1 \\ 2 & 2 & 2 \\ 1 & 2 & 5 \end{pmatrix} = \begin{pmatrix} 4 & 6 & 12 \\ 6 & 4 & 6 \\ 12 & 6 & 4 \end{pmatrix}$

► Exercice n°48

On considère les matrices $A = \begin{pmatrix} 3 & 7 \\ 2 & 5 \end{pmatrix}$; $B = \begin{pmatrix} 1 & 2 & 3 \\ 4 & -1 & 1 \end{pmatrix}$ et $C = \begin{pmatrix} 3 & 6 \\ 1 & 2 \end{pmatrix}$.

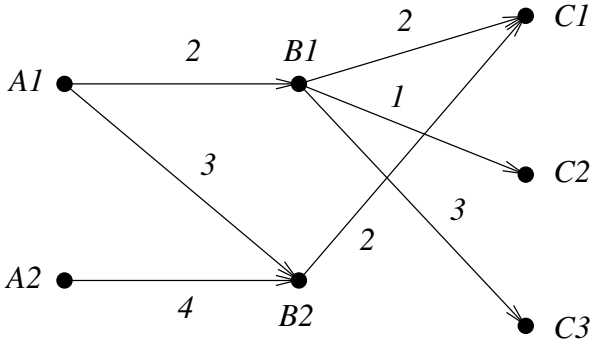
Déterminer si les propositions suivantes sont vraies ou fausses :

1. « On peut calculer $A + B$ » . FAUSSE
2. « On peut calculer $A + C$ » . VRAIE
3. « On peut calculer AB » . VRAIE
4. « On peut calculer BC » . FAUSSE
5. « On a $AB = BA$ » . FAUSSE - BA n'existe pas
6. « On a $B^2 = \begin{pmatrix} 1 & 4 & 9 \\ 16 & 1 & 1 \end{pmatrix}$ » . FAUSSE - B^2 n'existe pas
7. « On a $A^2 = \begin{pmatrix} 23 & 56 \\ 16 & 39 \end{pmatrix}$ ». VRAIE

► Exercice n°49

- Un petit site web A contient 2 pages $A1$ et $A2$;
- Un autre petit site web B contient 2 pages $B1$ et $B2$;
- Un dernier site web C contient 3 pages $C1$, $C2$ et $C3$.

Sur le schéma ci-dessous figure le nombre de liens qui pointent des pages du site A vers celles du site B et le nombre de liens qui pointent des pages du site B vers celles du site C.



1. Indiquer dans les matrices M et N ci-dessous, le nombre de liens qui pointent des pages du site A vers celles du site B et le nombre de liens qui pointent des pages du site B vers celles du site C.

$$M = \begin{matrix} \\ A1 \\ A2 \end{matrix}\begin{matrix} \begin{matrix} B1 & B2 \end{matrix} \\ \begin{pmatrix} 2 & 3 \\ 0 & 4 \end{pmatrix} \end{matrix} \quad \text{et } N = \begin{matrix} \\ B1 \\ B2 \end{matrix}\begin{matrix} \begin{matrix} C1 & C2 & C3 \end{matrix} \\ \begin{pmatrix} 2 & 1 & 3 \\ 2 & 0 & 0 \end{pmatrix} \end{matrix}$$

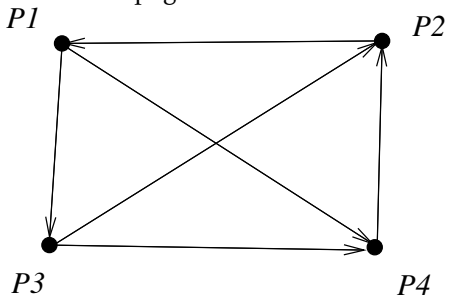
2. Calculer la matrice MN. Que permet-elle de retrouver comme information ?

$$MN = \begin{pmatrix} 10 & 2 & 6 \\ 8 & 0 & 0 \end{pmatrix}$$

On retrouve le nombre de manières possibles de passer des pages de A (lignes) à celles de C (colonnes)

► Exercice n°50

Les flèches dans le schéma ci-dessous indique l'existence d'un lien direct (en un clic) entre les 4 pages d'un site web.



1. Peut-on partir de la page $P1$ et y revenir en cliquant sur deux liens ? NON
2. De combien de façons peut-on aller de la page $P1$ à la page $P2$ en cliquant sur deux liens ? 2
3. De combien de façons peut-on aller de la page $P1$ à la page $P3$ en cliquant sur deux liens ? 0
4. De combien de façons peut-on aller de la page $P1$ à la page $P4$ en cliquant sur deux liens ? 1
5. Construire M, la matrice carrée de dimension 4 de telle façon que le coefficient de la i^e ligne et de la j^e colonne soit égal à 1 s'il y a une flèche de P_i vers P_j et 0 dans le cas contraire.

$$M = \begin{array}{c} \begin{matrix} P1 & P2 & P3 & P4 \end{matrix} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array} \begin{matrix} P1 \\ P2 \\ P3 \\ P4 \end{matrix}$$

6. Calculer M^2. Que retrouve-t-on à la première ligne de M^2 ?

$$M^2 = \begin{pmatrix} 0 & 2 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

On retrouve le nombre de façons possibles d'aller de $P1$ (ligne) aux autres pages (colonnes) en cliquant sur deux liens

► Exercice n°51

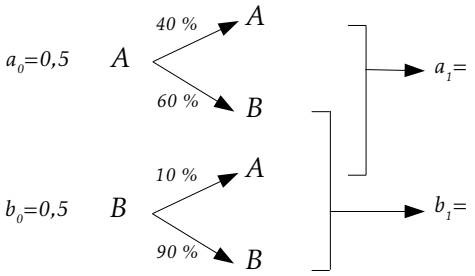
Un journal propose deux formules d'abonnement : une formule « papier » classique notée A et une formule 100% numérique notée B. En 2020 la répartition des abonnés est de 50% pour la formule A et de 50% pour la formule B.
Le journal prévoit que d'une année sur l'autre (à effectif d'abonnés stable) :

- 60% des abonnés à la formule A passeront à la formule B ;
- 10% des abonnés à la formule B passeront à la formule A .

On note :

- a_0 la proportion d'abonnés à la formule A en 2020 ;
- b_0 la proportion d'abonnés à la formule B en 2020 ;
- a_1 la proportion d'abonnés à la formule A en 2021 ;
- b_1 la proportion d'abonnés à la formule B en 2021 ;
- a_2 la proportion d'abonnés à la formule A en 2022 ;
- b_2 la proportion d'abonnés à la formule B en 2022 ;
- ainsi de suite...

a) Indiquer ci-dessous, le calcul de a_1 et b_1.



$a_1 = 0,4 \times a_0 + 0,1 \times b_0 = 0,25$; $b_1 = 0,6 \times a_0 + 0,9 \times b_0 = 0,75$

b) En déduire les coefficients à inclure dans la matrice ci-dessous pour que le calcul soit correct :

$$(a_1\ b_1) = (a_0\ b_0)\begin{pmatrix} 0,4 & 0,6 \\ 0,1 & 0,9 \end{pmatrix}.$$

c) Le mécanisme étant le même d'une année sur l'autre, pour avoir la répartition entre les formules d'abonnement en 2022, il suffit de remultiplier par la même matrice. Compléter le calcul ci-dessous :

$$(a_2\ b_2) = (a_1\ b_1)\begin{pmatrix} 0,4 & 0,6 \\ 0,1 & 0,9 \end{pmatrix} = (0,25\ 0,75)\begin{pmatrix} 0,4 & 0,6 \\ 0,1 & 0,9 \end{pmatrix} = (0,175\ 0,825)$$

d) On cherche à programmer un script python qui donne l'évolution de la répartition jusqu'en 2030. Expliquer pourquoi seul le script 2 ci-dessous pourra répondre au problème.

la valeur de a a changé pour le calcul de b dans le script 1 : calcul qui n'est donc plus valide

Script 1

```
a=0.5
b=0.5
for n in range(1,11):
    a=0.4*a+0.1*b
    b=0.6*a+0.9*b
    print("Année ",2020+n," : a=",a," b=",b)
```

Script 2

```
a=0.5
b=0.5
for n in range(1,11):
    a=0.4*a+0.1*b
    b=1-a
    print("Année ",2020+n," : a=",a," b=",b)
```

5. Représentation numérique des images

Pour décrire de façon numérique une image, il existe deux méthodes :

- la représentation dite *vectorielle* où l'on cherche à décrire l'image par les propriétés mathématiques des objets géométriques qu'elle contient (droites, cercles...). L'avantage de cette méthode est de pouvoir agrandir ou réduire autant que l'on veut l'image sans dégrader sa qualité (l'image étant reconstituée à chaque fois d'après les propriétés des objets géométriques), mais cette méthode ne convient que pour les images que l'on peut décrire uniquement à partir de figures géométriques (icônes, images d'illustration...) et est complètement inadaptée aux images photographiques par exemple.
  Un des formats d'image vectorielle le plus courant est le *svg* (« scalable vector graphic »).
- la représentation dite *matricielle* (ou *bitmap*) obtenue en quadrillant l'image par une grille et en décrivant chaque case, appelée *pixel*, de cette grille. La méthode de description de chaque pixel dépend du type de l'image : en noir et blanc, en nuances de gris ou en couleur.

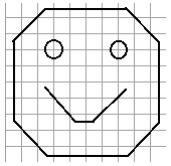
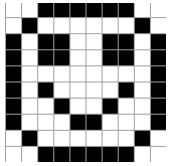


Image matricielle en noir et blanc : on noircit chaque pixel (case du quadrillage) contenant une portion de trait

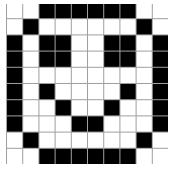


1) Exemple de codage d'une image en noir et blanc

► **Principe :** Une fois que chaque pixel s'est vu attribué la couleur blanche ou noire, on associe à la grille une matrice de même dimension en attribuant le coefficient 1 à un pixel noir et le coefficient 0 à un pixel blanc. Ainsi, un bit suffit pour coder un pixel.

► **Exercice n°52**

a) Compléter la matrice de codage associé à l'image ci-dessous :



$$\begin{pmatrix} 0&0&1&1&1&1&1&1&0&0\\ 0&1&0&0&0&0&0&0&1&0\\ 1&0&1&1&0&0&1&1&0&1\\ 1&0&1&1&0&0&1&1&0&1\\ 1&0&0&0&0&0&0&0&0&1\\ 1&0&1&0&0&0&0&1&0&1\\ 1&0&0&1&0&0&1&0&0&1\\ 1&0&0&0&1&1&0&0&0&1\\ 0&1&0&0&0&0&0&0&1&0\\ 0&0&1&1&1&1&1&1&0&0 \end{pmatrix}$$

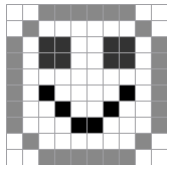
b) Combien de bits sont nécessaires au codage de la matrice de l'image ? 100

► **Remarque :** Ce type de codage est utilisé par le format d'image *pbm*.

2) Exemple de codage d'une image en niveaux de gris

► **Principe :** En plus du noir et du blanc, on peut utiliser (notamment pour les photos en noir et blanc) diverses nuances de gris. A chaque pixel, on associe cette fois dans la matrice un coefficient compris entre 0 et 255 : 0 pour le noir, 255 pour le blanc et les valeurs intermédiaires pour des gris plus ou moins clairs. Ainsi un octet suffit pour coder un pixel.

► **Exemple :**



$$\begin{pmatrix} 255&255&136&136&136&136&136&136&255&255\\ 255&136&255&255&255&255&255&255&136&255\\ 136&255&51&51&255&255&51&51&255&136\\ 136&255&51&51&255&255&51&51&255&136\\ 136&255&255&255&255&255&255&255&255&136\\ 136&255&0&255&255&255&255&0&255&136\\ 136&255&255&0&255&255&0&255&255&136\\ 136&255&255&255&0&0&255&255&255&136\\ 255&136&255&255&255&255&255&255&136&255\\ 255&255&136&136&136&136&136&136&255&255 \end{pmatrix}$$

► **Exercice n°53**

Combien de bits sont nécessaires au codage de la matrice de l'image de l'exemple ci-dessus ? 800

► Exercice n°54

Le codage de la matrice d'une image en niveau de gris nécessite 64 kio.

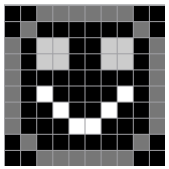
a) De combien de pixels est composée l'image ?
64 kio -> 64×1024=65536 octets -> 65536 pixels

b) Sachant que cette image est carrée, quelle est sa largeur en pixels ?
$\sqrt{65536} = 256$

► Remarque : Ce type de codage est utilisé par le format d'image *pgm*.

► Exercice n°55

Ci-dessous figure le négatif de l'image de l'exemple précédent et une partie de la matrice associée :



$$\begin{pmatrix} 0 & 0 & 119 & 119 & 119 & 119 & 119 & 119 & 0 & 0 \\ 0 & 119 & 0 & 0 & 0 & 0 & 0 & 0 & 119 & 0 \\ 119 & 0 & 204 & 204 & 0 & 0 & 204 & 204 & 0 & 119 \\ 119 & 0 & 204 & 204 & 0 & 0 & 204 & 204 & 0 & 119 \\ 119 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 119 \\ 119 & 0 & 255 & 0 & 0 & 0 & 0 & 255 & 0 & 119 \\ 119 & 0 & 0 & 255 & 0 & 0 & 255 & 0 & 0 & 119 \\ 119 & 0 & 0 & 0 & 255 & 255 & 0 & 0 & 0 & 119 \\ 0 & 119 & 0 & 0 & 0 & 0 & 0 & 0 & 119 & 0 \\ 0 & 0 & 119 & 119 & 119 & 119 & 119 & 119 & 0 & 0 \end{pmatrix}$$

a) Par quelle méthode passe-t'on de la matrice de l'image à celle de son négatif ?
coeff matrice négatif=255-coeff matrice original

b) Compléter la matrice associée au négatif.

3) Exemple de codage d'une image en couleur

► Principe : En mélangeant des teintes rouges, vertes et bleues on peut donner à l'œil humain la sensation de recevoir une lumière de n'importe quelle couleur.
Une des méthodes pour coder une image en couleur est le système *RVB* qui va associer à chaque pixel une intensité en rouge, en vert et en bleu ; ce qui revient à attribuer à chaque image trois matrices : une pour la couleur rouge, une pour la couleur verte et une pour la couleur bleue.
Si l'intensité de chaque couleur est représentée par un coefficient compris entre 0 et 255 (c'est à dire par un octet qui équivaut à 8 bits), on dit que l'on a à faire avec une image « 24 bits » ou image en « couleurs vraies ».

► Remarques :

- Ce type de codage est utilisé par le format d'image *ppm*.
- En photographie numérique, les intensités rouges, vertes et bleues sont obtenues par des capteurs, appelés *photosites*, couplés avec un filtre dit de *Bayer*.

► Exercice n°56

a) Quel est le nombre de couleurs différentes que peut utiliser une image « 24 bits » ?
$256 \times 256 \times 256 = 16777216$

b) Combien de Mio sont nécessaires pour coder les intensités de couleur d'une image « 24 bits » carrée d'une largeur de 1024 pixels ?
Il faut $1024x1024x3$ octets - en divisant par 2^{20}, ça correspond à 3 Mio

Conversion d'une image couleur en niveaux de gris

Une des méthodes possibles est de calculer l'intensité de gris de chaque pixel en faisant la moyenne des intensités rouges, vertes et bleues, ce qui revient à effectuer le calcul matriciel : $G = \frac{1}{3}(R+V+B)$ (où R, V et B représentent les matrices des intensités en couleur rouge, verte et bleue et G représente la matrice des nuances de gris).

► Exercice n°57

Retrouver les valeurs de A, B, C, D et E dans les matrices ci-dessous, pour que cela corresponde à la conversion d'une image couleur carrée de 3 pixels en niveau de gris.

$$R = \begin{pmatrix} 128 & 32 & 3 \\ 0 & 64 & 200 \\ \boxed{D} & 85 & 56 \end{pmatrix}; V = \begin{pmatrix} 42 & 0 & 42 \\ 180 & 11 & 81 \\ 24 & 65 & 0 \end{pmatrix}; B = \begin{pmatrix} 70 & 7 & \boxed{E} \\ 21 & 0 & 55 \\ 40 & 54 & 136 \end{pmatrix}$$

$$G = \begin{pmatrix} \boxed{A} & 13 & 59 \\ 67 & \boxed{B} & 112 \\ 91 & \boxed{C} & 64 \end{pmatrix}$$

$A = (128+42+70)/3 = 80$
$B = (64+11+0)/3 = 25$
$C = (85+65+54)/3 = 68$
$D + 24 + 40 = 3 \times 91 \Leftrightarrow D = 209$
$3 + 42 + E = 3 \times 59 \Leftrightarrow E = 132$

4) Définition, résolution, profondeur d'une image

Vocabulaire sur les images numériques

- La *définition* d'une image décrit le nombre de pixels qui la composent et s'écrit de la façon suivante : *(nombre de pixels en largeur)* × *(nombre de pixels en hauteur)*

- La *résolution* d'une image décrit le nombre de pixels par unité de longueur et permet de mesurer son degré de détail. Elle s'exprime en *dpi* pour « dots per inch » car l'unité de longueur utilisée dans les systèmes numériques est le *pouce* anglo-saxon qui vaut 2,54 cm.
- La *profondeur de couleur* d'une image est définie par le nombre de bits utilisés pour représenter chaque pixel.

► Exercice n°58

a) Quelle est la résolution d'une photographie numérisée de définition 2100×2100 imprimée sur une feuille carrée de côté 17,78 cm ?
17,78 cm -> 7 pouces - 2100/7=300 dpi
.....
.....

b) Quelle est la taille en pixels d'une photographie numérisée à la résolution de 400 dpi et imprimée sur une feuille carrée de côté 21,59 cm ?
21,59 cm -> 8,5 pouces et 8,5 x 400 =3400 - la photo fait 3400 x 3400 pixels
.....
.....

c) Retrouver la largeur et la hauteur en cm d'un écran d'un téléphone portable sachant que la notice indique que cet écran a une résolution de 523 dpi pour une définition de 1440 x 2960 pixels.
Quelle est la longueur en pouces de la diagonale de cet écran ?
1440/523=2,753 pouces -> 7cm ; 2960/523=5,660 pouces -> 14,4 cm
diagonale : $\sqrt{2,753^2+5,660^2} \approx 6,3$ pouces

d) À quelle résolution maximale peut-on prendre en photo dans un musée un tableau de 5 mètres par 5 mètres avec un appareil numérique possédant un capteur de 16 millions de pixels ?
5m -> 500/2,54=196,8 pouces
16 millions de pixels -> 4000 pixels en largeur
dpi=4000/196,8 = 20,3 dpi

e) Quelle est la profondeur de couleur d'une image noire et blanc ? 1

f) Quelle est la profondeur de couleur d'une image en 256 nuances de gris ? 8

► Remarque :
La valeur standard de la résolution d'un écran d'ordinateur est de 72 dpi et on estime que pour scanner ou imprimer un document graphique avec une qualité standard, il faut une résolution de 300 dpi.

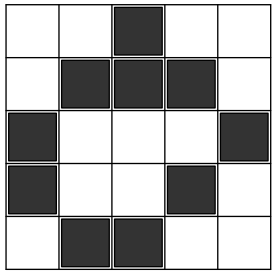
5) Compression

Les méthodes de codage d'images vues précédemment impliquent de donner une information chiffrée pour chaque pixel, ce qui génère des fichiers d'images dont la taille en octets devient vite importante. Des méthodes de *compression* permettent de réduire cette taille en permettant aux fichiers de prendre moins de place sur un périphérique de stockage et de faciliter son transfert sur Internet. On distingue deux types de compression :

- la compression *sans perte* qui permet de retrouver l'intégralité des données d'origine ;
- la compression *avec perte* qui réduit la taille du fichier en supprimant des informations non indispensables (par exemple des nuances que l'oeil humain peut à peine remarquer).

Une des méthodes les plus simples de compression sans perte pour des images en noir et blanc est le système de codage *RLE* (« Run Length Encoding ») qui consiste à compter le nombre de pixels blancs et noirs qui se suivent plutôt que d'attribuer un 0 ou un 1 à chaque pixel.

► Exemple avec l'image ci-dessous :



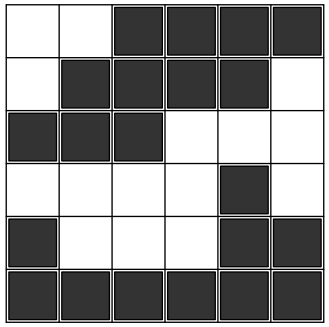
2,1,2
1,3,1
0,1,3,1
0,1,2,1,1
1,2,2

- La 1re ligne contient 2 pixels blancs, 1 pixel noir et 2 pixels blancs. Elle est donc représentée en système RLE par la suite « 2,1,2 » plutôt que par « 0,0,1,1,00 » (il y a donc réduction du nombre d'informations chiffrées nécessaires) ;
- La 2e ligne contient 1 pixels blanc, 3 pixels noirs et 1 pixel blanc. Elle est donc représentée en système RLE par la suite « 1,3,1 » ;
- La 3e ligne commence par 1 pixel noir suivi de 3 pixels blancs et d'un pixel blanc. Mais comme il faut toujours commencer le codage par le nombre de pixels blancs, cette ligne est représentée en système RLE par la suite « 0,1,3,1 ».

► Exercice n°59
Donner le code RLE des deux dernières lignes de l'image de l'exemple.

► Exercice n°60

1. Reconstituer l'image en noir et blanc ci-dessous d'après le code RLE de chaque ligne qui est indiqué :



2,4

1,4,1

0,3,3

4,1,1

0,1,3,2

0,6

2. Combien de nombres sont nécessaires pour coder l'image avec le système RLE ?
17 au lieu de 36

6) Formats, métadonnées

Les formats d'image pbm, pgm et ppm vus auparavant sont parmi les plus simples et plus faciles à lire, mais ne sont pas optimisés en taille de fichier généré et ne sont pas les plus courants.
D'autres formats d'images matricielles sont disponibles et sont beaucoup plus utilisés : ils se distinguent par la manière dont les données sont rangées et sont compressées (la compression pouvant se faire avec ou sans perte d'information).
Formats d'images usuels :

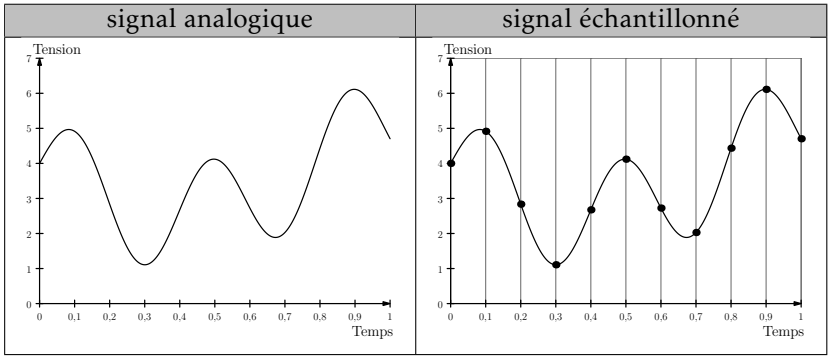
format	compression	profondeur de couleurs	utilité
RAW	oui sans perte	24 bits	photo
JPEG	oui avec perte	24 bits	photo
GIF	oui sans perte	8 bits	web
PNG	oui sans perte	jusqu'à 24 bits	web
TIFF	légère sans perte	jusqu'à 24 bits	numérisation
AVIF	oui avec perte	jusqu'à 24 bits	web

Par ailleurs, pour les photographies numériques, l'appareil enregistre en plus des informations matricielles sur les pixels, d'autres données supplémentaires appelées *métadonnées* qui peuvent contenir la date et l'heure de la prise de vues, le nom du fabricant de l'appareil, les réglages et les informations géographiques du lieu où a été prise la photographie.
Ces métadonnées peuvent être consultées (et modifiées) via les propriétés de l'image fournies par le système et/ou les logiciels graphiques. Le format le plus utilisé pour ces métadonnées est l'*EXIF*

6. De l'analogique vers le numérique - systèmes embarqués

1) Généralités

De nombreux systèmes informatiques ne comportent ni clavier, ni écran : on parle alors de *systèmes embarqués*. Ces unités informatiques reçoivent en entrée des données fournies par des capteurs qui mesurent des grandeurs physiques (intensité lumineuse, intensité sonore,etc.) converties en général sous la forme d'une tension électrique qui varie de façon continue en fonction du temps : c'est ce qu'on appelle un signal analogique
Or pour que ces données puissent être traités par un processeur informatique, elles doivent être numérisées, c'est à dire converties en une liste finie de nombres exprimés en binaire.
Une des méthodes pour faire cela consiste à ne lire la valeur du signal continu qu'à certains instants : c'est ce qu'on appelle échantillonner un signal.



2) Exemple : numérisation d'un son

Un son est une variation de la pression de l'air au cours du temps que l'on peut capter et convertir en signal électrique continu. Pour numériser un tel signal, une carte son du type « 44 kHz - 16 bits » va mesurer et quantifier la valeur du signal, avec un nombre sur 16 bits, 44 000 fois par seconde.

► Exercice n°61
Une carte son « 44 kHz - 16 bits » enregistre un son pendant 10 minutes. Quelle serait la taille, en Mio, du fichier généré s'il n'était pas compressé ?
Nombre de bits nécessaires : $10 \times 60 \times 44000 \times 16 = 422400000$

422400000 bits /8 -> 52800000 octets ; 52800000 octets / 2^{20} -> 50,35 Mio

► Remarque : les formats de fichiers musicaux contiennent souvent aussi, comme les formats de fichiers pour les images, des métadonnées qui indiquent les auteurs, le titre du morceau, l'album, etc...

7. La représentation de textes enrichis

Les textes en ASCII ou en unicode sont des textes bruts sans aucune mise en forme, ni structure. Ils servent notamment aux programmeurs qui les écrivent et les modifient avec des éditeurs de texte.
En dehors la programmation, des textes structurés avec une mise en forme dotée de titres, de paragraphes, d'extraits en gras ou en italiques, etc. sont souvent nécessaires. Ces textes enrichis peuvent être mis au point en bureautique avec des traitements de texte mais sont aussi présents dans les pages web.

1) Exemple avec le langage du web : le HTML

Le langage qui permet créer les pages web s'apelle le *HTML* (pour « HyperText Markup Language ») et est basé sur l'utilisation de *balises* : le principe consiste à encadrer le texte entre deux balises : `<balise>...</balise>` (une balise « ouvrante » et une balise « fermante » avec un /). Quand une page web est chargée par le navigateur, celui-ci interprète le code html de la page et affiche alors la page avec tous les enrichissements de forme indiqués par les balises.

Quelques balises de base

`<b>...</b>`	mettre en gras
`<i>...</i>`	mettre en italique
`<h1>...</h1>`	titre principal
`<h2>...</h2>`	sous-titre
`<p>...</p>`	un paragraphe

Des balises spéciales

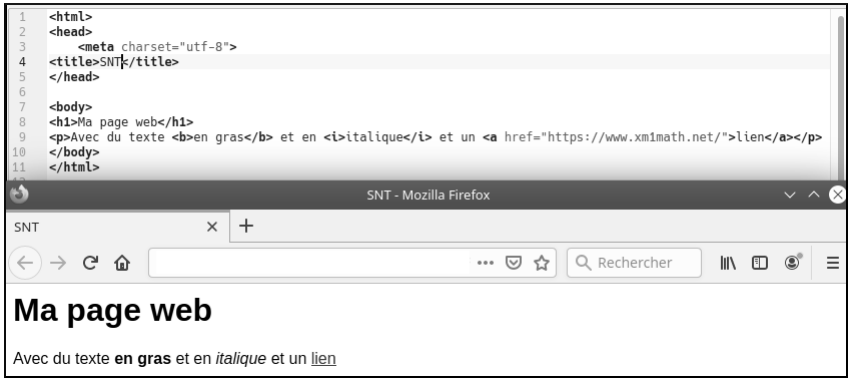
- Lien hypertexte (quand on clique, le navigateur affiche la page web dont l'adresse est spécifiée) :
  `<a href="http://lycee-palissy-agen.fr">lycée Palissy</a>`
- Une exception au principe des balises ouvrantes et fermantes est le code pour insérer une image . Il n'y a qu'une seule balise où l'on doit indiquer quelle est l'image à faire apparaitre avec le paramètre `src` :
  `<img src="monimage.jpg">`

Structure minimale d'une page html

Code html	Explication

`<html>`
`<head>`
`  <meta charset="utf-8">`
`  <title>Titre de la page</title>`
`</head>`
 	
`<body>`
 	
`</body>`
`</html>`
	• Entre les balises <head></head> se trouve l'en-tête de la page qui n'est pas affiché mais qui sert à paramétrer l'affichage par le navigateur.
	• <meta charset="utf-8"> sert à indiquer l'encodage utilisé pour éditer le code html (indispensable)
	• Entre les balises <title></title>, se trouve le titre de la page qui sera indiqué par le navigateur.
	• C'est entre les balises <body></body> que l'on indique tout le texte enrichi par les balises de base qui sera affiché par le navigateur.

Exemple minimaliste de rendu d'un code html par un navigateur



► **Exercice n°62**
Le code html suivant contient une erreur :

```
il y a <b>une erreur</b> quelque <i>part<i>.
```

Proposer un rectificatif : `<i>part</i>`

► Exercice n°63
Donner le code html pour que le texte `Sciences numériques` soit affiché par le navigateur avec `Sciences` en gras et `numériques` en italique :
`<b>Sciences</b> <i>numériques</i>`

► Exercice n°64
Vers quel site pointe le lien dans le code html suivant :

```
Cliquer <a href="http://escroc.com">ici</a> pour mettre à jour vos
coordonnées bancaires
```

Réponse : http ://escroc.com

► Exercice n°65
Quel est le nom du fichier de l'image que va afficher le code html suivant :

```
Une photo de vacances : <img src="agen_plage.jpg">
```

Réponse : agen_plage.jpg

► Remarque : Il est possible de changer le style (police, couleur, etc.) de chaque élément d'une page web via une *feuille de style* au format *css* (« Cascading Style Sheet »). Via un fichier css, on peut modifier de façon cohérente le style de l'ensemble de toutes les pages d'un site avec une gestion séparée du contenu (les pages html) et de la présentation (le fichier css).

Si l'on veut, par exemple, que tous les titres d'un site web soient affichés en rouge, il suffira d'indiquer le code suivant dans la feuille de style css du site web :

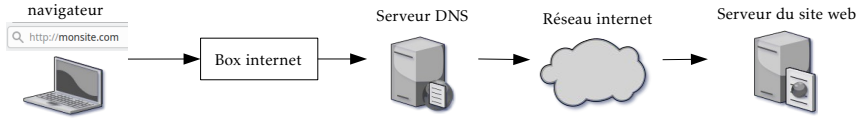
```
h1 {
  color: red;
}
```

À propos du web

Internet est un vaste ensemble de réseaux mondiaux interconnectés qui ne permet pas seulement d'avoirs accès aux sites web. Il permet aussi l'échange de fichiers, de mails, la transmission de conversations téléphonique, la diffusion de vidéos, de musique etc....
Le web n'est donc qu'un service parmi d'autres que permet d'utiliser internet.

Quelques spécificités du web

- Les accès à un site web se fait à l'aide des protocoles *http* (« hypertext transfert protocol ») et *https* (une version sécurisée où les communications sont chiffrées).
- Un site web est hébergé sur un serveur et est identifié par une adresse IP et peut-être atteint par un navigateur en tapant comme adresse : `http://xxx.xxx.xxx.xxx` où `xxx.xxx.xxx.xxx` serait l'adresse IP du site web.
  Mais comme ce n'est ni facile à utiliser ou à retenir comme adresse, le propriétaire d'un site web peut associer un nom à son site en achetant un *nom de domaine*. Dans ce cas là, l'adresse du site devient par exemple : `http://monsite.com`
  L'adresse du site est alors, ce qu'on appelle une *url* (« Uniform Ressource Locator » qui précise le protocole utilisé (http) et où se trouve le site à joindre.
  Pour faire la correspondance entre le nom de domaine et l'adresse IP, qui elle seule permet de joindre le site, des serveurs spéciaux appelés *DNS* (« Domain Name System ») sont appelés par le fournisseur d'accès à internet dès que l'on entre une adresse dans un navigateur. Ces serveurs *DNS* sont des immenses annuaires qui permettent de dire que tel nom de domaine correspond à telle adresse IP sur tel serveur.
- Schéma résumant (de façon simplifiée) ce qui se passe en tapant une adresse dans un navigateur :



8. Trier des données numériques

Pour pouvoir exploiter des données numériques, une des opérations standards à maitriser est le tri de ces données. Nous allons voir le principe d'une des méthodes les plus classiques pour cela, qui s'appelle le *tri par insertion*.

1) 1re étape : comment insérer un nouvel élément dans une liste triée ?

► **Exemple :** on considère la liste de nombres déjà triés dans l'ordre croissant ci-dessous que l'on appelle `liste_triee`.

Dessous la liste on a fait figurer la position des éléments dans la liste en commençant par 0 comme souvent en informatique.

a) Si l'on veut insérer comme nouvel élément le nombre 4 dans la liste en respectant l'ordre croissant, à quelle position faudrait-il l'insérer ? 2

b) Si l'on veut insérer comme nouvel élément le nombre 8 dans la liste en respectant l'ordre croissant, à quelle position faudrait-il l'insérer ? 4

c) On cherche maintenant un moyen algorithmique (donc automatique) pour savoir à quelle position placer n'importe quel nouveau élément. Pour cela on se place au début de la liste et on va chercher un principe qui répond au problème sous la forme suivante :
`tant qu'on est pas à la fin de la liste et que le terme de la liste à la position courante est inférieur au nouvel élément que l'on veut insérer, on passe à la position suivante`
Compléter la phrase pour que la méthode soit valable.

d) On va maintenant pouvoir en déduire une fonction python qui prend en paramètres une liste de nombres déjà triée dans l'ordre croissant et un nouvel élément et qui va retourner une nouvelle liste toujours triée mais à laquelle on a inséré à la bonne place le nouvel élément :

```
def insere(liste_triee,nouvel_element):
    position=0
    while (position<len(liste_triee) and liste_triee[position]<nouvel_element):
        position=position+1
    liste_triee.insert(position,nouvel_element)
    return liste_triee
```

Compléter l'inégalité dans la condition de la boucle `while` pour que le code corresponde à la méthode vue à la question précédente.

2) Le principe du tri par insertion

► **Explication du tri par insertion avec un exemple :**
Pour trier cette liste :

- On initialise une nouvelle liste (`liste_triee`) avec comme premier élément celui de la liste à trier :

- Puis on insère au fur et à mesure dans `liste_triee`, avec la méthode vue au paragraphe précédent, tous les autres termes de la liste à trier. Ainsi, on va insérer, en respectant l'ordre croissant, à `liste_triee` le 3, puis le 7, etc.

Cela donne en python (en affichant toutes les étapes) :

Code python

```
def insere(liste_triee,nouvel_element):
    position=0
    while (position<len(liste_triee) and
liste_triee[position]<nouvel_element):
        position=position+1
    liste_triee.insert(position,nouvel_element)
    return liste_triee

liste_non_triee=[12,3,7,1,9,24,8]
liste_triee=[liste_non_triee[0]]

for position in range(1,len(liste_non_triee)):
    insere(liste_triee,liste_non_triee[position])
    print(liste_triee)
```

Résultat après exécution

```
[3, 12]
[3, 7, 12]
[1, 3, 7, 12]
[1, 3, 7, 9, 12]
[1, 3, 7, 9, 12, 24]
[1, 3, 7, 8, 9, 12, 24]
```

► **Exercice n°66**
Quelle modification au code faudrait-il apporter pour n'afficher que le résultat final ?
Désindenter le print(liste_triee) pour qu'il sorte de la boucle .....
.....

► **Remarque :** Le principe du tri par insertion est celui qu'on utilise naturellement quand on range des cartes au fur et à mesure qu'elles sont distribuées.

9. Organisation des données

On vient de voir comment trier une donnée simple (une liste d'entiers), mais beaucoup de données en informatiques sont en fait composées de plusieurs éléments liés entre eux (nombres, textes, images, sons, etc.) qu'il faut organiser de façon à pouvoir les exploiter et éventuellement les croiser.

Info sur les données structurées

Pour illustrer le vocabulaire, on va utiliser les données météo suivantes :

Heure	Température °C	Vent km/h
00h	22.4	4
02h	19.9	11
04h	19.2	9

- Les données fournies dans ce tableau constituent ce qu'on appelle en informatique une table de données.
- Elle contient trois types de données (dites données élémentaires) classées par colonnes.
  On dit en informatique que l'heure, la température et la vitesse du vent sont des champs ou descripteurs de la table.
- Il existe plusieurs formats de fichiers pour enregistrer ce type de table de données. L'un des plus simples (et sommaires) est le format *csv* (« Comma-separated values ») qui consiste en un simple fichier texte dont la première ligne contient les intitulés des champs de la table séparés par des virgules et où ensuite chaque ligne contient les valeurs séparées elles aussi par une virgule.

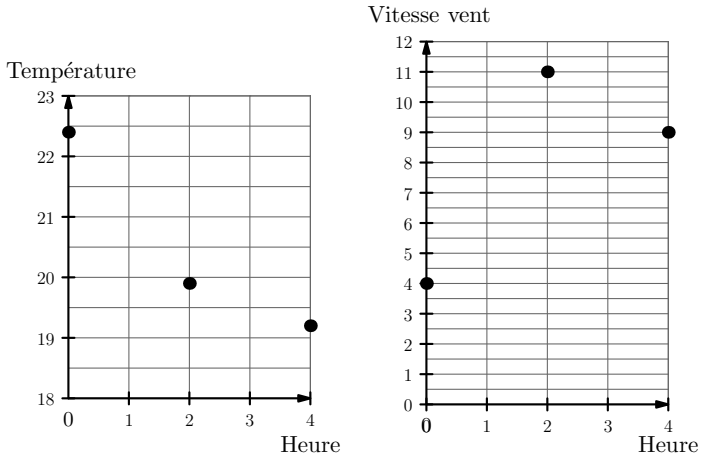
- Ainsi, le fichier *csv* correspondant à l'exemple serait constitué de la façon suivante :

```
Heure,Température °C,Vent km/h
00h,22.4,4
02h,19.9,11
04h,19.2,9
```

  Compléter la dernière ligne du fichier.
- Les algorithmes de tri permettent alors de trier la table de données selon le *champ* que l'on veut.
  Ci-dessous, figure la même table que l'exemple mais trié selon un autre champ. Lequel ? vitesse du vent (dans l'ordre croissant)

Heure	Température °C	Vent km/h
00h	22.4	4
04h	19.2	9
02h	19.9	11

- On peut aussi extraire les données d'une table pour créer des graphiques. Ci-dessous, figurent le graphique complet pour les températures et le repère pour celui de la vitesse du vent :



Rajouter les points pour la vitesse du vent.

10. Le problème des données personnelles

Les réseaux sociaux, les objets connectés (télévisions, voitures, ...) et l'internet des objets (téléphones, tablettes numériques ...) posent des problèmes nouveaux concernant la captation et l'exploitation de données insignifiantes en elles-mêmes mais "« *susceptibles de contribuer à un profilage très fin des individus et de produire à leur propos un «savoir» (probabilistique plutôt que de certitude) de leurs propensions personnelles et intimes, de leurs croyances religieuses, de leurs opinions politiques, de leur orientation sexuelle, de leur mode de vie et de bien d'autres aspects de leur vie personnelle et intime* »" (Commission Nationale Informatique et Libertés).
C'est pour cela qu'au niveau de l'Union Européenne un règlement a été mis en place auquel doit se soumettre toute organisation publique et privée qui traite des données personnelles (c'est à dire toute information se rapportant à une personne physique identifiée ou identifiable).
Les principales notions de ce règlement (appelé RGPD) sont :

- La pertinence : une organisation traitant des données personnelles ne doit collecter que celles qui sont nécessaires au fonctionnement du service qu'elle propose.
- Le consentement : un utilisateur doit donner son consentement pour qu'une organisation collecte une donnée personnelle le concernant.
- La transparence : un utilisateur doit être informé de tous les aspects concernant le traitement de ses données personnelles.

▶ Exercice n°67
Donner des exemples de données personnelles :
un nom, une photo, une empreinte, une adresse postale, une adresse mail, un numéro de téléphone, un numéro de sécurité sociale, une adresse IP, un identifiant de connexion informatique, un enregistrement vocal .....
.....

Info sur les « cookies »

Un cookie est un fichier texte généré par le serveur du site web que l'on visite ou par le serveur d'une application tierce (régie publicitaire, logiciel d'analyse du trafic internet, etc.) et qui est déposé par le navigateur sur l'appareil utilisé (ordinateur, téléphone, etc.).
Les cookies peuvent avoir un rôle technique indispensable en permettant de reconnaître et retrouver les préférences d'un internaute lorsqu'il revient sur un site web, mais ils peuvent aussi servir à « pister » les goûts et les comportements d'un internaute afin de lui proposer ensuite des publicités ciblées. Ainsi, par exemple, lorsqu'on navigue sur plusieurs sites vendant des chaussures, on risque d'être confronté à des publicités sur le thème des chaussures par la suite en raison des données enregistrées dans certains cookies de pistage.

3

Les graphes au service des systèmes numériques

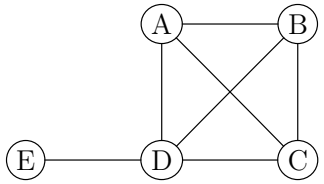
De nombreuses situations dans le monde numérique peuvent être schématisées à l'aide de graphes.

1. Les graphes non orientés

1) Généralités

Un graphe non orienté est un schéma composé de *sommets* dont certains sont reliés par des *arêtes*.

► **Exemple :** Dans le graphe ci-dessous, les sommets A, B, C, D et E représentent des membres d'un réseau social et la présence d'une arête entre deux sommets signifie que les deux membres sont amis sur ce réseau social.



Vocabulaire

- Le nombre total de sommets est appelé *ordre* du graphe.
  Avec l'exemple : l'ordre du graphe est égal à 5
- Deux sommets sont dits *adjacents* s'ils sont reliés par une arête.
  Dans le contexte de l'exemple, dire que deux sommets sont adjacents revient à dire que les deux membres sont amis.
  Citer dans l'exemple, deux sommets qui ne sont pas adjacents : A et E

- On appelle *degré* d'un sommet le nombre d'arêtes dont le sommet est une extrémité.
  Dans le contexte de l'exemple, que signifie le degré d'un sommet ? nombre d'amis du membre.
  Avec l'exemple, compléter le tableau suivant :

Sommet	*A*	*B*	*C*	*D*	*E*
Degré	*3*	*3*	*3*	*4*	*1*

  Quelle est la somme des degrés du graphe ? 14
  Quel est le nombre d'arêtes du graphe ? 7

Propriété

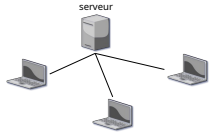
La somme des degrés d'un graphe est égal à deux fois le nombre d'arêtes. Cette somme ne peut donc être qu'un entier pair.

Vocabulaire

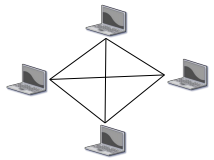
Un graphe est dit *complet* si chaque sommet est relié à tous les autres par une arête.
Le graphe de l'exemple est-il complet ? NON

Architectures classiques d'échanges sur un réseau

- Le cas le plus courant : le modèle *client/serveur* :
  Le *client* (ordinateur, téléphone, etc...) demande une ressource à un *serveur* (où sont, par exemple, stockés des sites web) qui lui répond en envoyant les données demandées.



- Le modèle *pair-à-pair* (« P2P : peer to peer » an anglais) :
  Chaque participant est à la fois client et serveur : chacun met ses ressources matérielles et logicielles au service de tous. Cela peut servir à échanger des fichiers ou faire du calcul distribué.



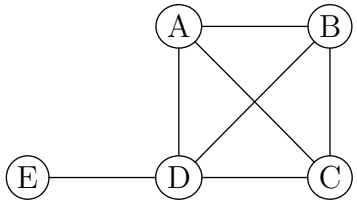
► Exercice n°68

Parmi les deux modèles d'échanges décrits ci-dessus, lequel peut être représenté par un graphe complet ? le modèle pair-à-pair

Vocabulaire

La *matrice d'adjacence* d'un graphe est une matrice carrée de même ordre que celui du graphe où un coefficient 1 indique l'existence d'une arête entre le sommet de la ligne et celui de la colonne (le coefficient est nul dans le cas contraire).

Compléter la matrice d'adjacence du graphe de l'exemple ci-dessous :

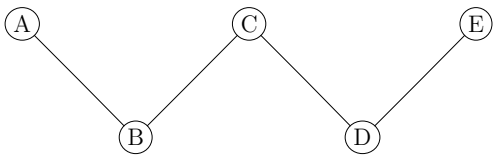


	A	*B*	*C*	*D*	*E*	
	0	1	1	1	0	*A*
	1	0	1	1	0	*B*
	1	1	0	1	0	*C*
	1	1	1	0	1	*D*
	0	0	0	1	0	*E*

Que retrouve-t-on en effectuant la somme des coefficients de chaque ligne ?
le degré du sommet de la ligne

► Exercice n°69

a) Compléter la matrice d'adjacence du graphe ci-dessous :



	A	B	C	D	E	
	0	1	0	0	0	A
	1	0	1	0	0	B
	0	1	0	1	0	C
	0	0	1	0	1	D
	0	0	0	1	0	E

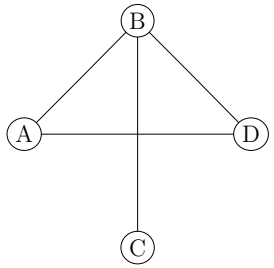
b) Donner l'ordre du graphe : 5

c) Compléter le tableau suivant :

Sommet	A	B	C	D	E
Degré	1	2	2	2	1

► Exercice n°70

Compléter le graphe ci-dessous pour qu'il corresponde à la matrice d'adjacence donnée.



	A	B	C	D	
	0	1	0	1	A
	1	0	1	1	B
	0	1	0	0	C
	1	1	0	0	D

► Exercice n°71

Un graphe possède quatre sommets A, B, C et D et comporte 11 arêtes.
A est de degré 3, B est de degré 4 et C est de degré 5.
Quel est le degré du sommet D ?
11 arêtes -> somme des degrés=22 -> degré du sommet D=22-12=10

► Exercice n°72

Un graphe complet est d'ordre 10.

1. Quel est le degré de chacun des sommets ? 9
2. Combien d'arêtes le graphe comporte-t-il ? $(10 \times 9)/2 = 45$

► Exercice n°73

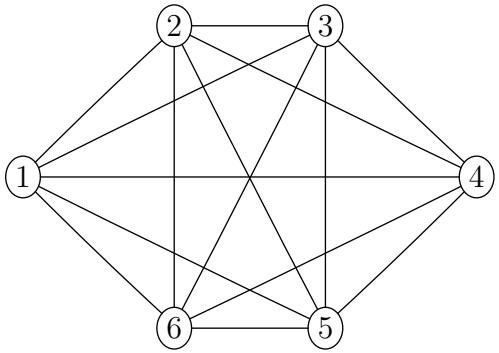
Peut-il exister un graphe d'ordre 6 dont les degrés des sommets sont donnés ci-dessous ? NON : somme des degrés impaire

Sommet	A	B	C	D	E	F
Degré	2	4	5	3	4	3

► Exercice n°74

Un tournoi d'e-sport réunit 6 joueurs numérotés de 1 à 6 qui doivent s'affronter en duel. Chaque joueur rencontre une fois et une seule les cinq autres.

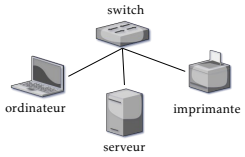
1. Compléter le graphe ci-dessous dans lequel les sommets représentent les joueurs de façon à ce que les arêtes représentent une rencontre.



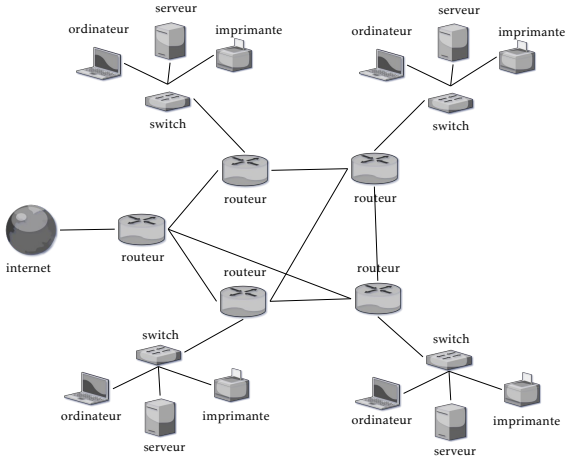
2. Quel est le degré de chacun des sommets ?5
   Quel est le nombre total d'arêtes ?15
   Le graphe est-il complet ?OUI
3. Sachant que trois assauts peuvent se dérouler en même temps, déterminer un calendrier possible du tournoi. *(on pourra s'aider en marquant au fur et à mesure les arêtes correspondantes à une série de trois duels)*
   1/2 ; 3/4 ; 5/6
   1/3 ; 2/5 ; 4/6
   1/4 ; 2/6 ; 3/5
   1/5 ; 2/4 ; 3/6
   1/6 ; 2/3 ; 4/5

Info sur les réseaux informatiques

- Pour faire communiquer entre eux localement des ordinateurs, des imprimantes, des serveurs, etc. on se sert d'un *switch* qui utilise directement l'identifiant de chaque carte réseau (*l'adresse mac*) pour transmettre les données (sans passer par les adresses IP).



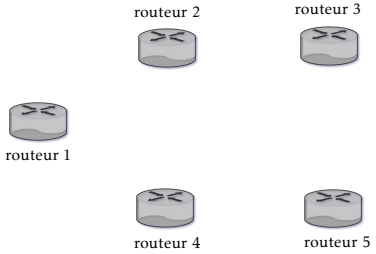
- Alors qu'un *switch* connecte plusieurs périphériques pour créer un réseau local, un *routeur* connecte plusieurs *switchs* et leurs réseaux respectifs pour former un plus grand réseau et sert alors de répartiteur : il dirige le trafic en utilisant les protocoles TCP et IP et choisit l'itinéraire le plus efficace (avec ce qu'on appelle une *table de routage*) pour que les informations, sous forme de paquets de données, arrivent à la bonne destination.
  Outre la connexion de plusieurs réseaux entre eux, le routeur permet aussi l'accès à internet.



- Les box des fournisseurs d'accès à internet sont à la fois un switch et un routeur avec un modem pour la connexion internet.

► Exercice n°75

On dispose de 5 routeurs. Est-il possible de relier chacun d'entre eux à trois autres routeurs parmi les 5 ?



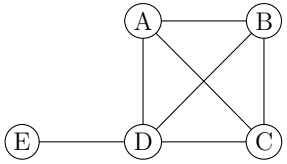
non la somme des degrés serait impaire (15) : impossible

. . . . . .

. . . . . .

2) Chaines

Pour illustrer les définitions, on reprend notre exemple des 5 membres d'un réseau social :

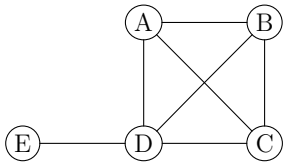


Vocabulaire

- Dans un graphe, une *chaine* est une liste ordonnée d'arêtes permettant de décrire un chemin reliant deux sommets et on appelle *longueur* de la chaine le nombre d'arêtes qui la composent.
  Dans l'exemple, E-D-B-C-A-D est une chaine de longueur 5.
- La *distance* entre deux sommets est le plus petit nombre d'arêtes qu'il faut emprunter pour les joindre.
  Avec l'exemple : la distance entre E et B est égale à 2.
  Que cela signifie-t-il dans le contexte de l'exemple :
  un seul ami intermédiaire suffit pour mettre en contact E et B

  . . . . . .

- L'*excentricité* d'un sommet est la plus grande distance entre ce sommet et les autres sommets.
  Avec l'exemple (rappelé ci-dessous), compléter le tableau ci-dessous pour qu'il indique la distance entre deux sommets et l'excentricité de chaque sommet



Distance	A	B	C	D	E	excentricité
A	0	1	1	1	2	2
B	1	0	1	1	2	2
C	1	1	0	1	2	2
D	1	1	1	0	1	1
E	2	2	2	1	0	2

- Le *diamètre* d'un graphe est la plus grande distance entre deux de ses sommets.
  Pour l'exemple, le diamètre est égal à 2
- Le *rayon* d'un graphe est la plus petite excentricité de ses sommets.
  Pour l'exemple, le rayon est égal à 1
- On appelle *centre* d'un graphe tout sommet dont l'excentricité est égale au rayon.
  Pour l'exemple, le(s) centre(s) du graphe est(sont) D
  Que cela signifie-t-il dans le contexte de l'exemple :
  c'est le membre le plus « proche » des autres

  . . . . . .

► Exercice n°76

Avec l'exemple,

a) De combien de façons peut-on passer de D à A,B et C avec une chaine de longueur 2 ? 2

b) De combien de façons peut-on passer de D à D avec une chaine de longueur 2 ? 4

c) De combien de façons peut-on passer de D à E avec une chaine de longueur 2 ? 0

d) Le carré de la matrice d'adjacence du graphe donne :

$$\begin{array}{c} \begin{matrix} A & B & C & D & E \end{matrix} \\ \begin{pmatrix} 3 & 2 & 2 & 2 & 1 \\ 2 & 3 & 2 & 2 & 1 \\ 2 & 2 & 3 & 2 & 1 \\ 2 & 2 & 2 & 4 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array} \begin{matrix} A \\ B \\ C \\ D \\ E \end{matrix}$$

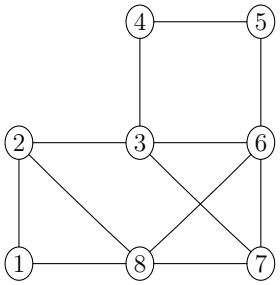
Que retrouve-t-on à la ligne de D ? le nombre de chaines de longueur 2 vers les autres sommets

Propriété

Si M est la matrice d'adjacence d'un graphe simple, alors les coefficients de M^n donne le nombre de chaines de longueur n entre le sommet de la ligne et le sommet de la colonne.

► Exercice n°77

Huit routeurs sont reliés selon le graphe ci-dessous :



a) Quelle est la distance entre les sommets 1 et 5 ? 3
En déduire le nombre minimum de routeurs par lequel doit passer un paquet envoyé du routeur 1 pour arriver au routeur 5 : 2

b) On note M la matrice associée à ce graphe et on admet que la cinquième ligne de M^3 donne : $(1, 3, 1, 4, 0, 6, 3, 1)$. En déduire le nombre de chaines de longueur 3 entre les sommets 5 et 6 : 6
Parmi ces chaines, lesquelles permettent-elles d'envoyer un paquet de données du routeur 5 au routeur 6 en passant par deux autres routeurs ?
5-4-3-6

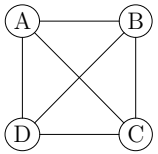
c) On souhaite que deux routeurs reliés par une simple arête puisse être identifié sur un plan du réseau par une couleur différente. Justifier qu'il faudra au moins trois couleurs différentes.
1,2 et 8 par exemple nécessiteront des couleurs différentes

3) Sous-graphe complet

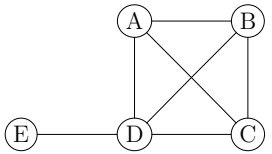
Propriété

Un sous-graphe complet d'un graphe est un extrait du graphe (sommets et arêtes les joignant) dont chaque sommet est relié par une arête à tous les autres.

► Exemple :



est un sous-graphe complet de

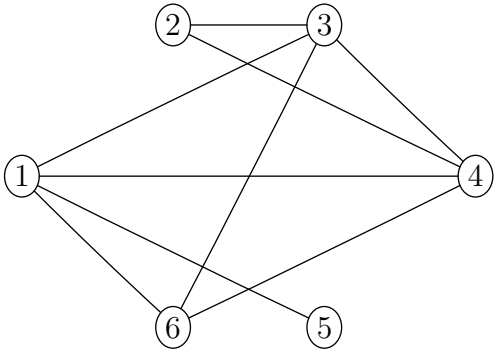


► Exercice n°78

Un élève souhaiterait inviter à une soirée six de ses camarades de classe (numérotés de 1 à 6), mais uniquement ceux qui sont amis entre eux sur un réseau social afin d'éviter une mauvaise ambiance. Le tableau ci-dessous indique ceux qui ne sont **pas amis** entre eux :

camarade de classe	1	2	3	4	5	6
n'est pas ami avec	2	1,5,6	5	5	2,3,4,6	2,5

a) Représenter cette situation dans le graphe ci-dessous en reliant par une arête les camarades de classe qui **sont amis** entre eux.



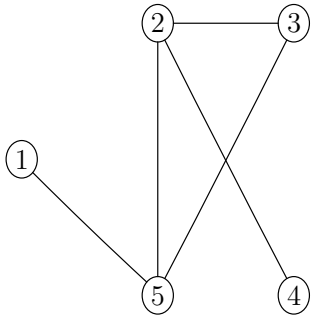
b) Combien de camarades au maximum l'élève peut-il inviter sans risque de mauvaise ambiance ?
cela revient à chercher le sous-graphe complet ayant le plus grand nombre de sommets :
Il peut en inviter 4 : 1,3,4 et 6

► Exercice n°79

Une entreprise dispose de 5 gros serveurs numérotés de 1 à 5 qui nécessitent une bonne climatisation. Pour éviter des problèmes de consommation électrique, ils ne peuvent être utilisés ensemble qu'avec les contraintes suivantes :

- Le serveur 1 peut être utilisé en même temps que le serveur 5 ;
- Le serveur 2 peut être utilisé en même temps que les serveurs 3, 4 et 5 ;
- Le serveur 3 peut être utilisé en même temps que le serveur 5 ;
- Les autres combinaisons ne sont pas possibles.

a) Représenter cette situation par le graphe ci-dessous dans lequel on relie les serveurs pouvant être utilisés ensemble par une arête.



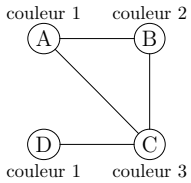
b) Quel est le nombre maximal de serveurs pouvant être utilisés ensemble ? Citer ces serveurs.

cela revient à chercher le sous-graphe complet ayant le plus grand nombre de sommets - 3 serveurs peuvent être utilisés en même temps : 2, 3 et 5

4) Coloration d'un graphe

Vocabulaire

- La *coloration* d'un graphe consiste à affecter une « couleur » à chacun des sommets de sorte que deux sommets reliés par une arête ne portent pas la même couleur. C'est un moyen de différencier les sommets d'un graphe de façon à ce que deux sommets adjacents ne soient pas dans la même catégorie. *Exemple :*



- Le plus petit nombre de couleurs permettant la coloration d'un graphe est appelé *nombre chromatique*

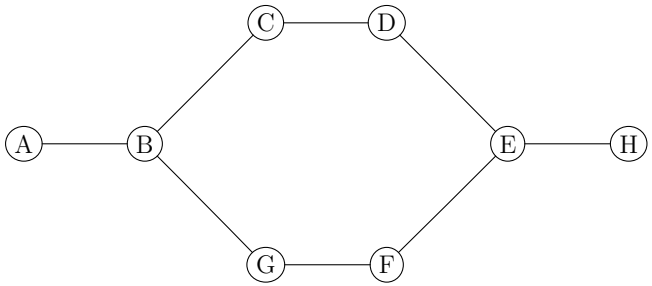
Propriété

- Si le graphe est complet le nombre chromatique est égal au nombre de sommets
- Si le graphe n'est pas complet alors le nombre chromatique est supérieur ou égal au nombre de sommets nécessaire pour former le sous-graphe complet d'ordre maximal.

Un algorithme de coloration

- On classe dans un tableau les sommets dans l'ordre décroissant de leurs degrés ;
- On attribue une nouvelle couleur (par un numéro par exemple) au premier sommet non encore coloré du tableau et la même couleur à chaque sommet non adjacent à un sommet de cette couleur (dans l'ordre du tableau) ;
- On réitère l'étape précédente jusqu'à ce que tous les sommets se sont vu attribués une couleur.

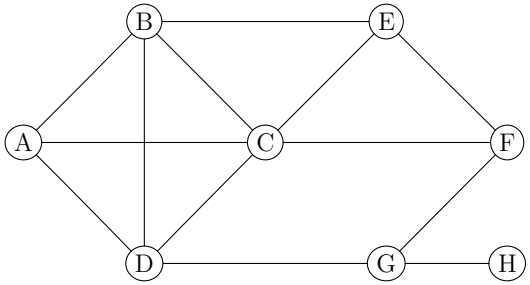
► Exemple :



Sommet	B	E	C	D	F	G	A	H
Degré	3	3	2	2	2	2	1	1
Couleur	1	1	2	3	2	3	2	2

► Exercice n°80

Dans le graphe suivant, les sommets représentent 8 émetteurs de télécommunication pour la 4G et une arête entre deux sommets signifie que l'on ne peut pas allouer la même fréquence aux deux émetteurs correspondants pour éviter des interférences entre eux à cause de leur proximité. On cherche à minimiser le nombre de fréquences à utiliser.



a) Justifier qu'il faudra au moins 4 fréquences différentes.
A,B,C et D forment un sous-graphe complet d'ordre 4

b) Proposer une coloration du graphe en s'aidant du tableau suivant

Sommet	C	B	D	A	E	F	G	H
Degré	5	4	4	3	3	3	3	1
Couleur	1	2	3	4	3	2	1	2

c) Indiquer alors une répartition des émetteurs pouvant utiliser la même fréquence :
fréquence 1 : C,G
fréquence 2 : B,F,H
fréquence 3 : D,E
fréquence 4 : A

Info sur le « Big Data »

On désigne par le terme « Big Data » les énormes quantités de données numériques générées par l'immensité des individus et objets connectés et les nouveaux moyens qu'il a fallu créer pour y accéder et les analyser. Le « Big Data » peut notamment être utilisé pour développer des technologies d'*intelligence artificielle* (IA) qui nécessitent de gros moyens informatiques.

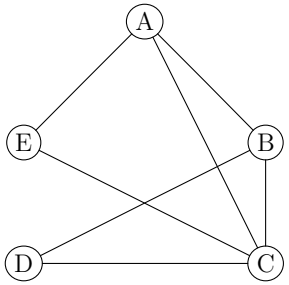
► Exercice n°81

Dans un laboratoire d'intelligence artificielle comportant 5 machines numérotées de 1 à 5, un analyste doit effectuer cinq tâches notées A,B,C,D et E :

- la tâche A nécessite l'utilisation des machines 1, 3 et 5 ;
- la tâche B nécessite l'utilisation des machines 1 et 2 ;
- la tâche C nécessite l'utilisation des machines 2, 3 et 5 ;
- la tâche D nécessite l'utilisation des machines 2 et 4 ;
- la tâche E ne nécessite que l'utilisation de la machine 5.

Chaque tâche requiert le même temps d'exécution d'une heure et une même machine ne peut être utilisée que pour une seule tâche à la fois.

1. Représenter la situation par le graphe ci-dessous ayant pour sommets les cinq tâches et en reliant les tâches ne pouvant pas être exécutées en même temps.



2. Justifier qu'il faudra au moins 3 couleurs différentes pour colorer ce graphe.
B,C et D forment un sous-graphe complet d'ordre maximal

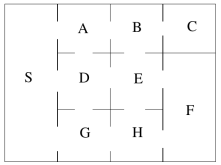
3. Proposer une coloration du graphe en s'aidant du tableau suivant

Sommet	C	A	B	D	E
Degré	4	3	3	2	2
Couleur	1	2	3	2	3

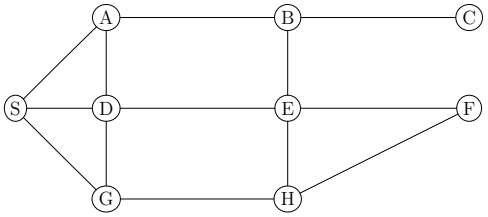
4. En déduire une organisation du travail qui permet de minimiser le temps total nécessaire à l'exécution des cinq tâches.
tache C / taches A et D / taches B et E : total 3 heures

► Exercice n°82

Dans la figure ci-dessous, les lettres S, A, B ... représentent les salles d'un data center et une ouverture représente une porte de communication entre deux salles.



1. Compléter le graphe ci-dessous en traçant une arête pour chaque porte de communication entre deux salles



a) Ce graphe est-il complet ?
non car, par exemple, les sommets C et F ne sont pas reliés.

b) Quel est l'ordre maximal des sous-graphes complets du graphe ? 3

c) Quel est le diamètre du graphe ? 4

2. On note M la matrice associée au graphe (avec l'ordre des sommets suivant : S, A, B, C, D, E, F, G et H).

On donne la matrice M^4 : $\begin{pmatrix} 18 & 12 & 11 & 2 & 20 & 12 & 6 & 12 & 12 \\ 12 & 20 & 3 & 6 & 11 & 20 & 5 & 18 & 5 \\ 11 & 3 & 16 & 0 & 19 & 3 & 8 & 4 & 12 \\ 2 & 6 & 0 & 3 & 1 & 7 & 1 & 4 & 1 \\ 20 & 11 & 19 & 1 & 31 & 9 & 11 & 12 & 19 \\ 12 & 20 & 3 & 7 & 9 & 28 & 9 & 20 & 9 \\ 6 & 5 & 8 & 1 & 11 & 9 & 9 & 8 & 9 \\ 12 & 18 & 4 & 4 & 12 & 20 & 8 & 20 & 6 \\ 12 & 5 & 12 & 1 & 19 & 9 & 9 & 6 & 17 \end{pmatrix}$

a) Combien y-a-t-il de chemins qui, en quatre étapes, partent de D et reviennent à D ? 31

b) Combien y-a-t-il de chemins qui, en quatre étapes, partent de S et arrivent à C ? Citer ces chemins.
2
S-D-E-B-C et S-D-A-B-C

c) Est-il toujours possible de joindre en quatre étapes deux salles quelconques ?
non, car un des coefficients de M^4 est nul.

3. Le responsable du data center souhaite différencier chaque salle de sa ou ses salles voisines (accessible par une porte) par la couleur de la moquette posée au sol. À l'aide du tableau ci-dessous, proposer une répartition de couleurs qui réponde au problème.

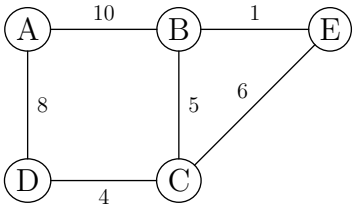
Sommet	D	E	A	B	G	H	S	F	C
Degré	4	4	3	3	3	3	3	2	1
Couleur	1	2	2	1	2	1	3	3	2

5) Graphes pondérés

Vocabulaire

Un *graphe pondéré* est un graphe sur lequel on affecte à chaque arête un coefficient (qui peut représenter une distance, un temps de parcours, un coût etc.). On appelle alors *poids* d'une chaine la somme des coefficients sur les arêtes qui la composent.

▶ Exemple de graphe pondéré :

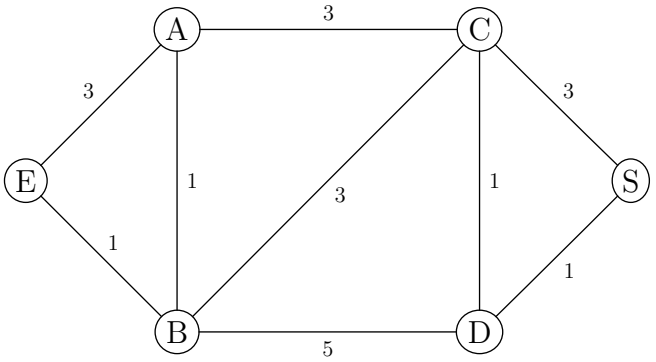


Le poids de la chaine A-B-C-E est égal à 21
Le poids de la chaine A-D-C-E est égal à 18

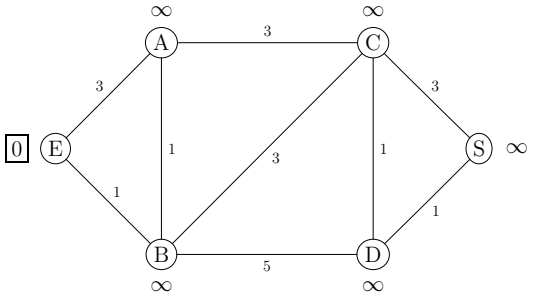
6) Chaine de poids minimal - Algorithme de Dijsktra

Dans de nombreuses situations modélisées par un graphe pondéré, on est amené à chercher la chaine reliant deux sommets dont le poids est minimal (recherche de distance minimale, de plus faible coût, etc.). Pour cela une méthode efficace consiste à utiliser l'algorithme de Dijsktra que nous allons détailler avec l'exemple ci-dessous :

▶ Exemple : recherche de la chaine de poids minimal du sommet E au sommet S.

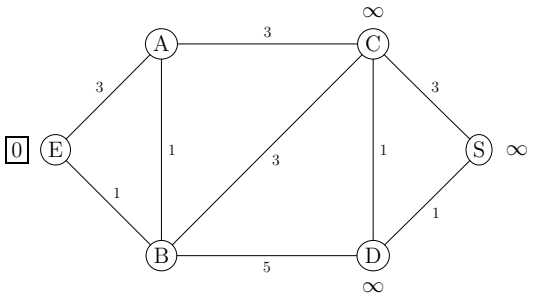


- *Étape 1* :
  - On associe le coefficient 0 au sommet de départ E ;
  - On associe le coefficient ∞ aux autres sommets ;
  - On sélectionne le sommet de départ E en l'encadrant.



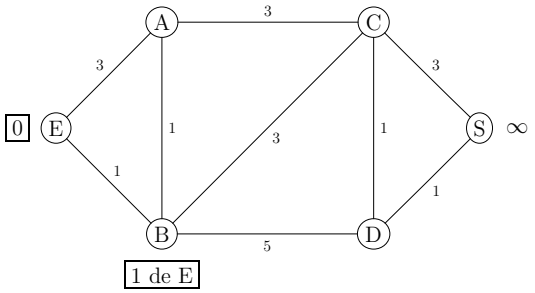
- *Étape 2* :
  - Pour chaque arête sortant de E et qui se termine par un sommet non encore sélectionné, on ajoute le coefficient affecté à E (c'est à dire 0) au poids de l'arête.
    Si le résultat est inférieur au coefficient actuel du sommet auquel aboutit l'arête, on affecte à ce sommet ce résultat comme nouveau coefficient (en précisant de quel sommet il provient).
  - On sélectionne alors le sommet dont le coefficient est le plus bas parmi les sommets non encore sélectionnés.

  Résultat :
  - A passe de « ∞ » à « 3 de E » car $0+3<\infty$
  - B passe de « ∞ » à « 1 de E » car $0+1<\infty$
  - On sélectionne le sommet B en l'encadrant.



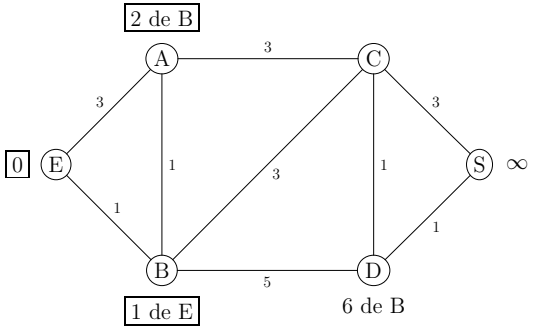
- *Étape 3* :
  - Pour chaque arête sortant de B et qui se termine par un sommet non encore sélectionné, on ajoute le coefficient affecté à B (c'est à dire 1) au poids de l'arête.
    Si le résultat est inférieur au coefficient actuel du sommet auquel aboutit l'arête, on affecte à ce sommet ce résultat comme nouveau coefficient (en précisant de quel sommet il provient).
  - On sélectionne alors le sommet dont le coefficient est le plus bas parmi les sommets non encore sélectionnés.

  Résultat :
  - A passe de « 3 de E » à « 2 de B » car $1+1<3$
  - C passe de « ∞ » à « 4 de B » car $1+3<\infty$
  - D passe de « ∞ » à « 6 de B » car $1+5<\infty$
  - On sélectionne le sommet A en l'encadrant (car c'est le sommet non encore sélectionné qui a le coefficient le plus bas).



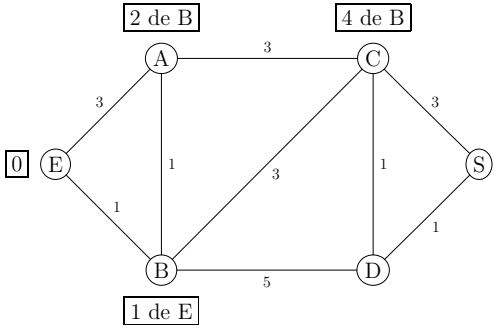
- *Étape 4* :
  - Pour chaque arête sortant de A et qui se termine par un sommet non encore sélectionné, on ajoute le coefficient affecté à A (c'est à dire 2) au poids de l'arête.
    Si le résultat est inférieur au coefficient actuel du sommet auquel aboutit l'arête, on affecte à ce sommet ce résultat comme nouveau coefficient (en précisant de quel sommet il provient).
  - On sélectionne alors le sommet dont le coefficient est le plus bas parmi les sommets non encore sélectionnés.

  Résultat :
  - C reste à « 4 de B » car $4<2+3$
  - On sélectionne le sommet C en l'encadrant (car c'est le sommet non encore sélectionné qui a le coefficient le plus bas).



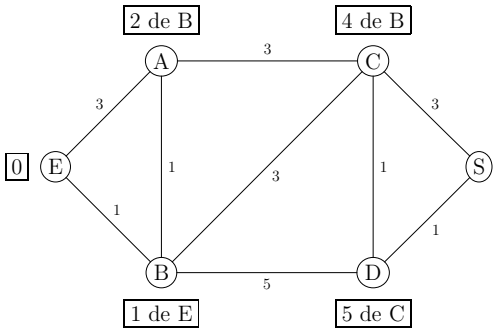
- *Étape 5* :
  - Pour chaque arête sortant de C et qui se termine par un sommet non encore sélectionné, on ajoute le coefficient affecté à C (c'est à dire 4) au poids de l'arête.
    Si le résultat est inférieur au coefficient actuel du sommet auquel aboutit l'arête, on affecte à ce sommet ce résultat comme nouveau coefficient (en précisant de quel sommet il provient).
  - On sélectionne alors le sommet dont le coefficient est le plus bas parmi les sommets non encore sélectionnés.

  Résultat :
  - D passe de « 6 de B » à « 5 de C » car $4+1<6$
  - S passe de « ∞ » à « 7 de C » car $4+3<\infty$
  - On sélectionne le sommet D en l'encadrant (car c'est le sommet non encore sélectionné qui a le coefficient le plus bas).



- *Étape 6* :
  - Pour chaque arête sortant de D et qui se termine par un sommet non encore sélectionné, on ajoute le coefficient affecté à D (c'est à dire 5) au poids de l'arête.
    Si le résultat est inférieur au coefficient actuel du sommet auquel aboutit l'arête, on affecte à ce sommet ce résultat comme nouveau coefficient (en précisant de quel sommet il provient).
  - On sélectionne alors le sommet dont le coefficient est le plus bas parmi les sommets non encore sélectionnés.

  Résultat :
  - S passe de « 7 de C » à « 6 de D » car 5+1<7
  - On sélectionne le sommet S en l'encadrant.
  - Tous les sommets sont ont été sélectionnés : l'algorithme est terminé.



- On détermine alors la chaine de poids minimal en « remontant » le parcours à partir de S :
  - S vient de D ;
  - D vient de C ;
  - C vient de B ;
  - B vient de E ;

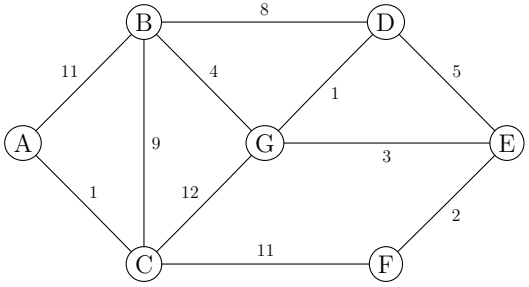
  La chaine de poids minimal de E à S est donc : E-B-C-D-S.
  Son poids total est égal à 6 (1+3+1+1).

Algorithme de Dijkstra

- **Initialisation :** on affecte 0 au sommet de départ et ∞ aux autres sommets et on sélectionne le sommet de départ.
- **Á chaque étape :**
  Pour chaque arête sortant du dernier sommet sélectionné et se terminant par un sommet non encore sélectionné, on ajoute le coefficient du sommet sélectionné au poids de l'arête.
  Si le résultat est inférieur au coefficient actuel du sommet auquel aboutit l'arête, on affecte à ce sommet ce résultat comme nouveau coefficient (en précisant de quel sommet il provient).
  On sélectionne alors le sommet dont le coefficient est le plus bas parmi les sommets non encore sélectionnés.
- On répète le processus jusqu'à ce que tous les sommets soient sélectionnés.

► Exercice n°83

Déterminer la chaine de poids minimal entre les sommets D et A dans le graphe ci-dessous (compléter au fur et à mesure les étapes dessous le graphe) :

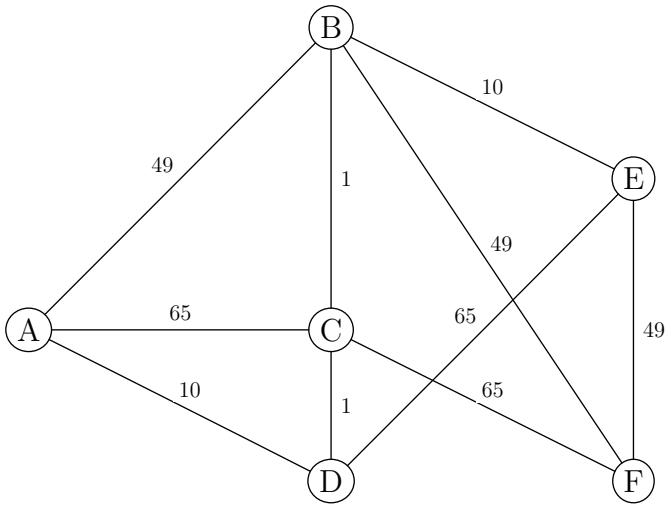


- Étape 1 : sélection de D
- Étape 2 : B :8 de D ; E :5 de D ; G :1 de D ; sélection de G
- Étape 3 : B :5 de G ; C :13 de G ; E :4 de G ; sélection de E
- Étape 4 : F :6 de E ; sélection de B
- Étape 5 : A :16 de B ; sélection de F
- Étape 6 : sélection de C
- Étape 7 : A :14 de C ; sélection de A
- Chaîne de poids minimal entre D et A : D-G-C-A

► Exercice n°84

Pour acheminer des paquets de données, les algorithmes de routage peuvent prendre en compte la bande passante de chaque liaison entre les routeurs. On attribue alors un *coût* à chaque liaison inversement proportionnel à la bande passante : ainsi plus le *coût* est faible, plus la liaison est rapide (un coût de 1 correspond à une liaison en fibre optique).

Dans le graphe ci-dessous, les sommets représentent des routeurs et le poids de chaque arête indique le *coût* de la liaison. Déterminer la chaine qui correspond à la liaison la plus rapide entre les routeurs A et E (compléter au fur et à mesure les étapes dessous le graphe) :

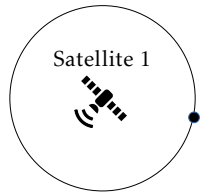


- Étape 1 : sélection de A
- Étape 2 : B :49 de A ; C :65 de A ; D :10 de A ; sélection de D
- Étape 3 : C :11 de D ; E :75 de D ; sélection de C
- Étape 4 : B :12 de C ; F :76 de C ; sélection de B
- Étape 5 : E :22 de B ; F :61 de B ; sélection de E
- Étape 6 : sélection de F
- Liaison la plus rapide entre A et E : A-D-C-B-E

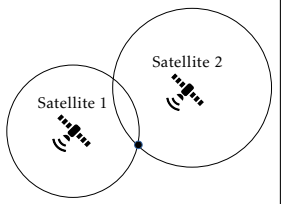
Info sur la géolocalisation par satellite

Principe pour déterminer les coordonnées précises d'un appareil via un système de géolocalisation par satellite (GPS,Galileo,..) :

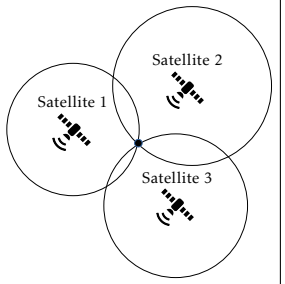
L'appareil envoie un signal vers des satellites. Grâce à une mesure très précise du temps mis entre la réception et l'émission du signal, un premier satellite peut déterminer à quelle distance précise de lui se trouve l'appareil et donc sur quelle sphère centrée en lui se trouve l'appareil. Mais cela ne suffit pas à déterminer où sur la sphère se trouve l'appareil. (ci-contre : vision simplifiée de la sphère - le point représente l'appareil)



Un deuxième satellite capte aussi le signal et détermine sur quelle sphère centrée en lui se trouve l'appareil. L'appareil doit donc se trouver à l'intersection des deux sphères.



Un troisième satellite captant aussi le signal permet finalement de déterminer les coordonnées précise de l'appareil à l'intersection des trois sphères.

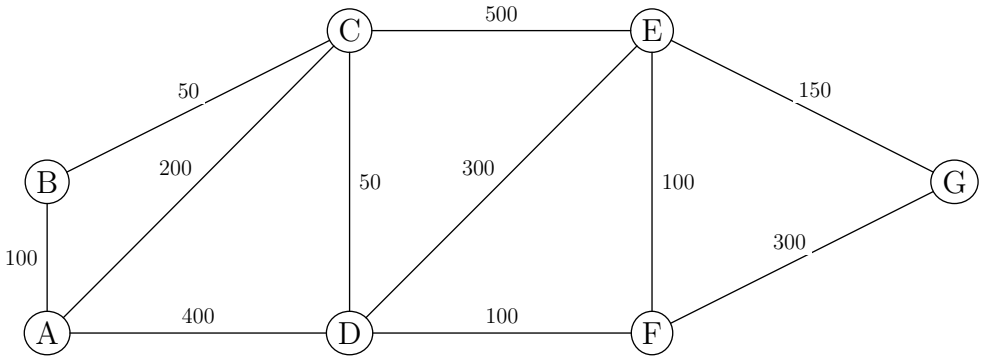


Remarques :

- Il faut une mesure très précise du temps pour que cela fonctionne. C'est pour cela que les satellites embarquent des horloges atomiques ultra-précises.
- Il faut aussi que les trois satellites et l'appareil dont on veut la position soit parfaitement synchronisée. C'est pour cela que dans les faits, il faut un quatrième satellite pour synchroniser l'horloge de l'appareil avec celui des satellites.
- Ce procédé s'appelle la trilatération et permet aux systèmes de navigation de fournir de nombreux services.

► Exercice n°85

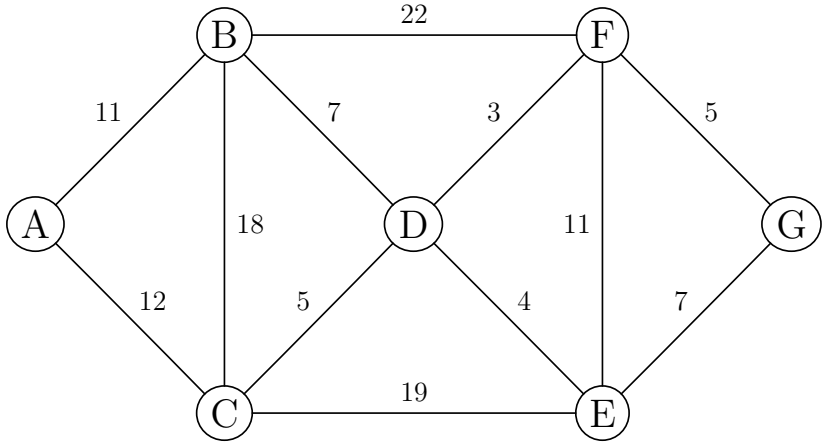
Le graphe ci-dessous représente les itinéraires possibles à un piéton souhaitant aller du point A au point G. Les arêtes indiquent la distance en mètres entre deux points.
Le piéton demande au service de navigation de son téléphone le chemin le plus court pour aller de A à G.
Déterminer quelle va être la proposition du système de navigation (compléter au fur et à mesure les étapes dessous le graphe) :



- Étape 1 : sélection de A
- Étape 2 : B :100 de A ; C :200 de A ; D :400 de A ; sélection de B
- Étape 3 : C :150 de B ; sélection de C
- Étape 4 : D :200 de C ; E :650 de C ; sélection de D
- Étape 5 : E :500 de D ; F :300 de D ; sélection de F
- Étape 6 : E :400 de F ; G :600 de F ; sélection de E
- Étape 7 : G :550 de E ; sélection de G
- Itinéraire le plus court : A-B-C-D-F-E-G

► Exercice n°86

Dans un jeu vidéo, un joueur circule dans des catacombes infestées de monstres qu'il doit combattre.
On a représenté ci-dessous le graphe modélisant ces catacombes. Les sommets représentent les salles et les arêtes représentent les couloirs. Les nombres sur les arêtes correspondent au nombre de monstres présents dans chaque couloir.



1. Donner un chemin qui permette à un joueur en A d'y revenir après avoir parcouru tous les couloirs une et une seule fois.
   A-B-F-G-E-F-D-E-C-D-B-C-A

2. Un joueur situé en G souhaite parvenir en A en affrontant le minimum de monstres. Déterminer à l'aide de l'algorithme de Dijkstra un chemin qui réponde au problème.

   - Étape 1 : sélection de G
   - Étape 2 : E :7 de G ; F :5 de G ; sélection de F
   - Étape 3 : B :27 de F ; D :8 de F ; sélection de E
   - Étape 4 : C :26 de E ; sélection de D
   - Étape 5 : B :15 de D ; C :13 de D ; sélection de C
   - Étape 6 : A :25 de C ; sélection de B
   - Étape 7 : sélection de A

   Chemin avec le minimum de monstres : G-F-D-C-A

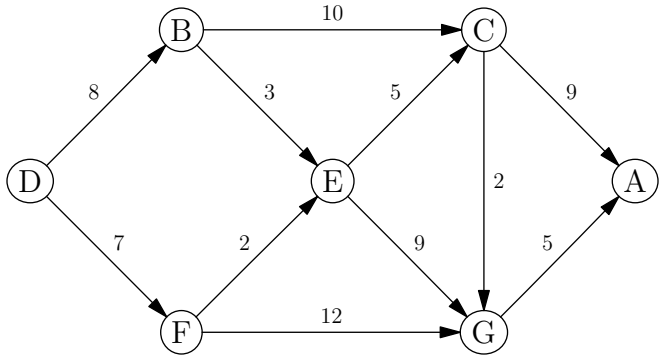
3. À l'aide de l'algorithme du cours, proposer une coloration du plan des catacombes de telle façon que deux salles reliées par un couloir n'aient pas la même couleur :

Sommet	B	C	D	E	F	A	G
Degré	4	4	4	4	4	2	2
Couleur	1	2	3	1	2	3	3

► Exercice n°87

Le graphe ci-dessous représente un réseau de cours d'eau que les joueurs d'un jeu en ligne peuvent parcourir à l'aide d'une barque :

- Le sommet D représente la ligne de départ et le sommet A, la ligne d'arrivée.
- Une flèche indique le sens d'écoulement de l'eau ;
- Le nombre sur chaque arête indique le nombre d'épreuves à passer entre deux jonctions.



1. Compléter ci-dessous la matrice d'adjacence de ce graphe.

$$\begin{array}{c} \begin{matrix} A & B & C & D & E & F & G \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{matrix} A \\ B \\ C \\ D \\ E \\ F \\ G \end{matrix} \end{array}$$

2. Citer tous les parcours permettant d'aller de D en A en ne parcourant que 3 cours d'eau.
   D-B-C-A
   D-F-G-A
3. Déterminer à l'aide de l'algorithme de Dijkstra le circuit qui permet d'aller de D en A en affrontant le moins d'épreuves possible :
   - Étape 1 : sélection de D
   - Étape 2 : B :8 de D ; F :7 de D ; sélection de F
   - Étape 3 : E :9 de F ; G :19 de F ; sélection de B
   - Étape 4 : C :18 de B ; sélection de E
   - Étape 5 : C :14 de E ; G :18 de E ; sélection de C
   - Étape 6 : G :16 de C ; A :23 de C ; sélection de G
   - Étape 7 : A :21 de G ; sélection de A

   Circuit à emprunter : D-F-E-C-G-A

2. Les graphes orientés

Dans certains cas les relations symbolisées par les arêtes entre deux sommets d'un graphe ne peuvent avoir lieu que dans un sens. On ajoute alors un sens symbolisé par une flèche aux arêtes et le graphe est dit orienté dans ce cas là (les principales propriétés des graphes rencontrées jusque là restent valides).

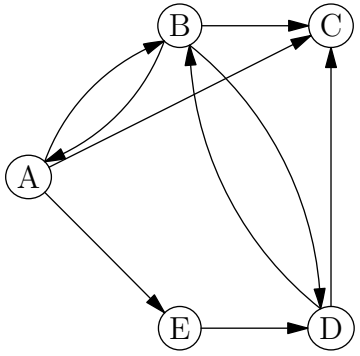
► Exercice n°88

Dans certains réseaux sociaux, les relations ne sont pas centrées autour du concept réciproque d'« amis », mais sur le fait que des membres peuvent « suivre » d'autres membres sans que cela soit réciproque. Les graphes orientés sont alors tout à fait adaptés :

Cinq élèves (notés A, B, C, D et E) sont membres d'un réseau social où :

- A suit ce que publient B, C et E ;
- B suit ce que publient A, C et D ;
- C ne suit aucun autre élève ;
- D suit ce que publient B et C ;
- E suit ce que publie D.

1. Compléter le graphe orienté ci-dessous en ajoutant une arête fléchée quand l'élève de départ suit l'élève d'arrivée (un tel graphe est appelé *sociogramme* par certains).



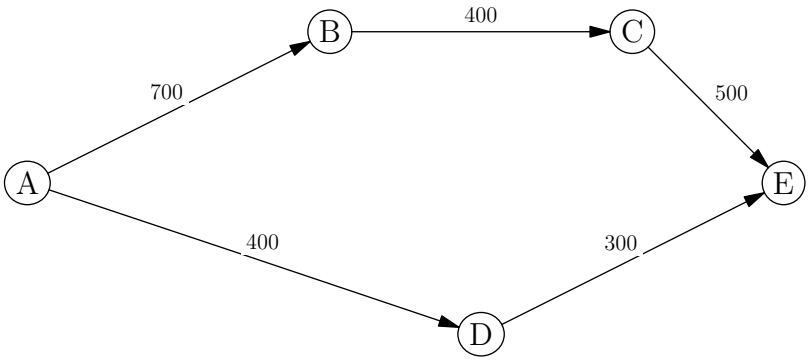
2. Compléter la matrice d'adjacence du graphe ci-dessous *(attention à partir de la ligne pour arriver à la colonne et à bien regarder si le parcours de l'arête est possible pour mettre un 1)*

	A	B	C	D	E	
	0	1	1	0	1	A
	1	0	1	1	0	B
	0	0	0	0	0	C
	0	1	1	0	0	D
	0	0	0	1	0	E

3. Comment peut-on déterminer l'élève qui est le plus suivi par les autres ?
c'est le sommet qui correspond à la colonne dont le total des coefficients est le plus grand : C .....

► Exercice n°89

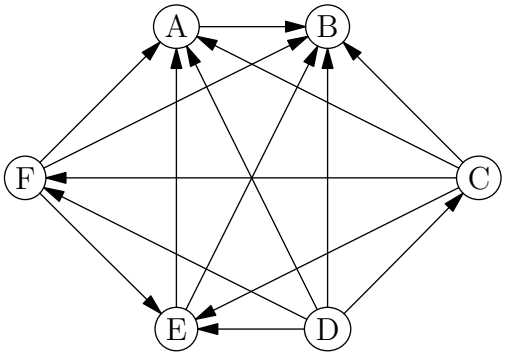
Le graphe orienté ci-dessous schématise les deux routes possibles par lesquelles doit passer un internaute depuis son poste de travail, symbolisé par le sommet A, pour joindre un site hébergé sur le serveur E. Le nombre indiqué sur chaque arête représente le nombre maximum de connexions simultanées que peut supporter le serveur d'arrivée.



Combien de connexions simultanées sont-elles possibles en tout pour le site hébergé sur le serveur E ? 400 (chemin du haut)+300 (chemin du bas)=700

► Exercice n°90

Dans le graphe orienté ci-dessous :



- chaque sommet représente un joueur d'e-sport lors d'une phase éliminatoire ;
- une arête indique que le joueur de départ a battu le joueur d'arrivée.

1. Compléter ci-dessous la matrice d'adjacence de ce graphe.

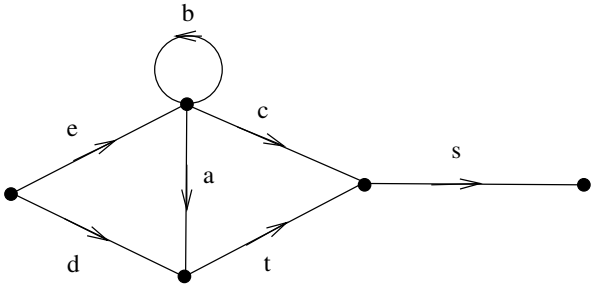
	A	B	C	D	E	F	
	0	1	0	0	0	0	A
	0	0	0	0	0	0	B
	1	1	0	0	1	1	C
	1	1	1	0	1	1	D
	1	1	0	0	0	0	E
	1	1	0	0	1	0	F

2. Que représente la somme des coefficients de chaque ligne de la matrice ?
nombre de duels gagnés par le joueur

3. Que représente la somme des coefficients de chaque colonne de la matrice ?
   nombre de duels perdus par le joueur
4. Une victoire rapporte 2 points et une défaite 0 point. Déterminer quel joueur a le meilleur à l'issue de cette phase éliminatoire. le joueur D avec 10 points

▶ Exercice n°91

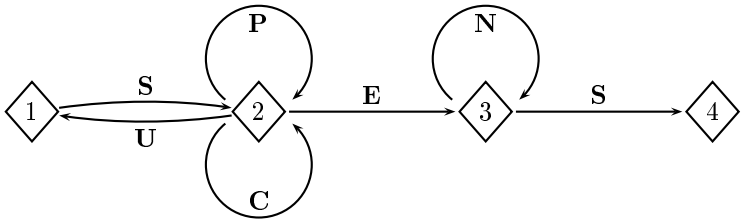
Dans de nombreux sites web nécessitant une inscription, il est souvent demander de prouver à travers un petit test si le visiteur n'est pas un « robot » (c'est à dire un programme informatique qui pourrait compromettre le site avec des milliers de fausses inscriptions).
Comme test, un site demande aux visiteurs de rentrer un mot qui ne peut être formé qu'en parcourant un graphe comme dans l'exemple ci-dessous :



Avec cet exemple, les mots « ebcs » , « ebacs » , « dts » , « ebbats » et « eats » seront-ils reconnus comme valides ?
oui ;non ;oui ;oui ;oui

▶ Exercice n°92

Pour accéder à sa messagerie, Antoine a choisi un code qui doit être reconnu par le graphe étiqueté suivant, de sommets 1, 2, 3 et 4 :



Une succession des lettres constitue un code possible si ces lettres se succèdent sur un chemin du graphe orienté ci-dessus, en partant du sommet 1 et en sortant au sommet 4.

1. Le code SUCCES est-il reconnu valide par le graphe ? NON
2. Le code SCENES est-il reconnu valide par le graphe ? NON
3. Le code SUSPENS est-il reconnu valide par le graphe ? OUI
4. Compléter la matrice d'adjacence M associée au graphe. On prendra les sommets dans l'ordre 1-2-3-4.

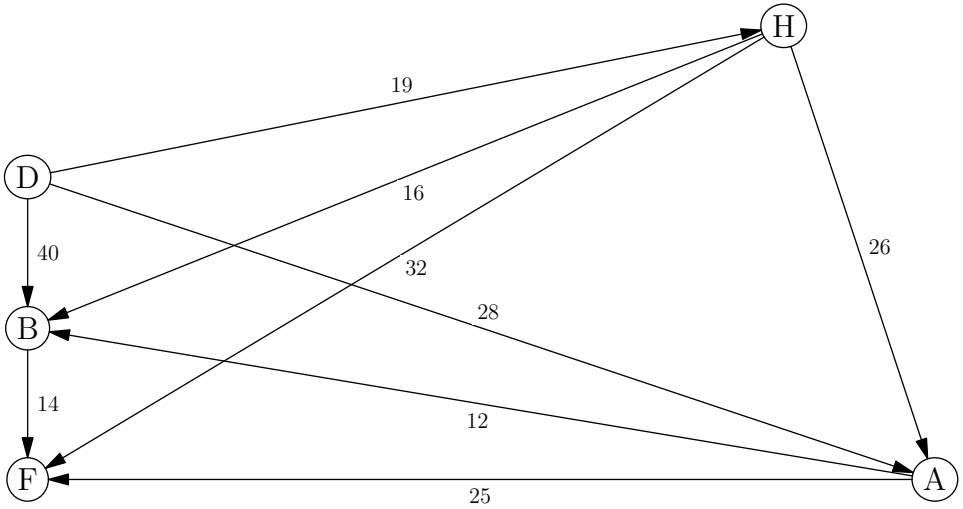
$$M = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

5. On admet que $M^4 = \begin{pmatrix} 5 & 12 & 8 & 3 \\ 12 & 29 & 20 & 8 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$
   En déduire le nombre de codes de 4 lettres reconnus par le graphe : 3
   Quels sont ces codes ?
   SENS , SPES et SCES

▶ Exercice n°93

Un parcours sportif virtuel fourni par une application de fitness est représenté par le graphe orienté ci-dessous où :

- Le sommet D représente le départ ;
- Le sommet F représente la fin du parcours ;
- Les sommets B, H et A représentent des exercices de fitness à effectuer ;
- Les arêtes représentent des temps de course sur tapis roulant à effectuer entre deux sommets.



1. On note M la matrice d'adjacence de ce graphe où les sommets sont rangés dans l'ordre alphabétique. On donne $M^3 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$

   Combien de parcours de 3 arêtes sont-ils possibles de D à F ? 3 Citer ses parcours : D-H-A-F ; D-A-B-F ; D-H-B-F

2. Un utilisateur souhaite courir le moins longtemps possible sur tapis roulant en allant de D à F. Déterminer, à l'aide de l'algorithme de Dijkstra, le trajet pour lequel le temps de course est minimal :

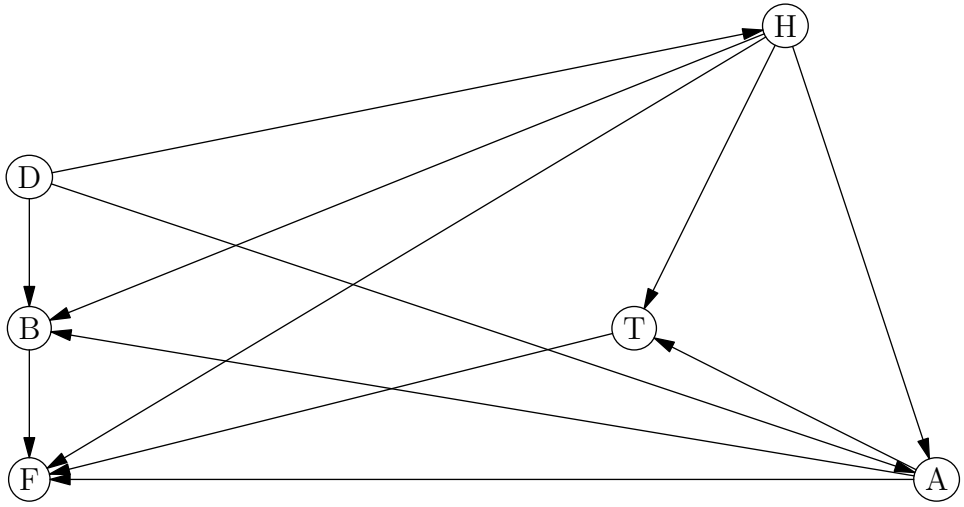
   - Étape 1 : sélection de D
   - Étape 2 : A :28 de D ; B :40 de D ; H :19 de D ; sélection de H
   - Étape 3 : B :35 de H ; F :51 de H ; sélection de A
   - Étape 4 : sélection de B
   - Étape 5 : F :49 de B ; sélection de F

   Trajet à effectuer : D-H-B-F

3. Le concepteur de l'application souhaite ajouter un exercice noté T créant ainsi de nouveaux parcours possibles. La matrice d'adjacence N associée au graphe représentant les nouveaux parcours, dans lequel les sommets sont classés dans l'ordre alphabétique, est

$$N = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

   Compléter le graphe ci-dessous , en ajoutant les arêtes nécessaires au graphe orienté pour qu'il corresponde à la matrice N.



4 Compléments

1. Les moteurs de recherche

1) Principe

Les moteurs de recherche mis au point pour permettre de rechercher des données dans la masse grandissante des ressources disponibles sur le Web.

Les techniques mises en œuvre par les moteurs de recherche :

- Des logiciels robots appelés spiders et crawlers parcourent automatiquement les sites Web en suivant tous les liens qui y figurent et inspectent leurs contenus.
- Les mots-clés et les métadonnées des pages visitées sont stockés dans des bases de données où ils sont associés à l'adresse des pages Web. Cette étape est appelée indexation .
- Un algorithme, propre à chaque moteur de recherche, classe alors les pages associées à un mot-clé selon leur pertinence.

Du côté de l'internaute :

- Lors d'une recherche, les mots-clés envoyés par l'internaute sont comparés à ceux stockés dans les bases de données du moteur de recherche.
- Les pages Web associées par le moteur de recherche à ces mots-clés sont alors classées selon leur pertinence et renvoyées sous la forme d'une liste de liens.
- La plupart des moteurs de recherche ajoutent au début de la liste des résultats des liens dits « sponsorisés » qui sont définis lors de ventes aux enchères de certains mots-clés à des entreprises commerciales. Ce référencement payant est appelé SEA (« Search Engine Advertising »).

Du côté des auteurs de sites Web, il existe des techniques pour faciliter le référencement des sites par les moteurs de recherche. Ces techniques sont appelés SEO (« Search Engine Optimization »).

▶ Exercice n°94

Citer quatre moteurs de recherche. Réponses possibles : google bing duckduckgo yahoo qwant ecosia ...

▶ Exercice n°95

Ci-dessous figure un extrait du code source d'une page d'un site Web.

```
<meta name="description" content="Vos articles à petits prix : culture, high-tech,
mode, jouets, sport, maison et bien plus !">
<meta name="keywords" content="livres, achats en ligne, librairie, magazine, abon-
nement, musique, Cds, DVD, vidéo, électronique, vêtements, accessoires, chaussures,
bijoux, montres, produits de bureau, sports en plein air, articles de sport,...">
```

À quel type de sites Web cela peut-il correspondre ? site de commerce en ligne

▶ Exercice n°96

Le premier résultat de la recherche sur « mairie » sur deux moteurs de recherche différents figure ci-dessous. Comment expliquer la différence ?

Réponse : le premier base aussi ses résultats sur la géolocalisation de l'internaute d'après son adresse IP

2) Détermination du classement des résultats

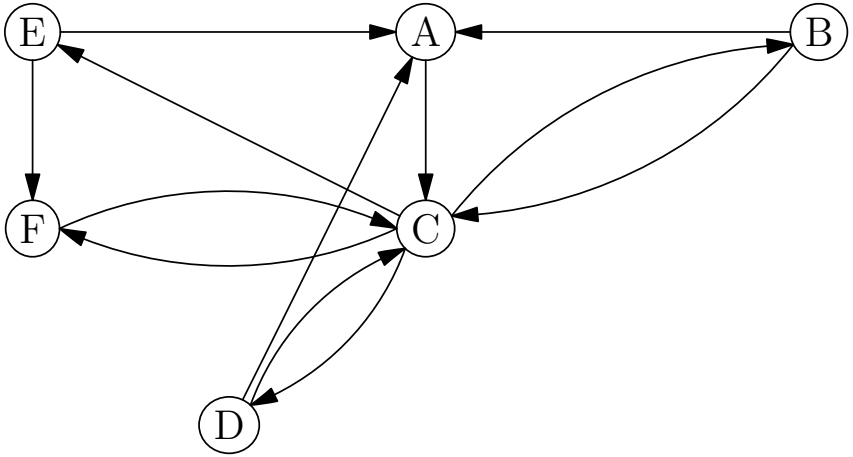
Pour classer les résultats trouvés selon leur pertinence, les moteurs de recherche se basent sur deux types de critère :

- Les critères dits « in page » basés sur le contenu des pages (présence des mots-clés dans le texte, les métadonnées, la description des images ou dans l'adresse même de la page)
- Les critères dits « off page » basés notamment sur la fréquentation, le sérieux, le nombre et la qualité des liens qui figurent sur les sites.

L'algorithme « PageRank » créé par Google a permis de créer un système de notation des pages en se basant à l'origine sur la méthode dite du « surfeur aléatoire » qui attribue à une page une note comprise entre 0 et 1 selon le principe suivant :
Après avoir établi la liste de tous les sites traitant la requête, un robot en choisit un au hasard. Puis il regarde les liens hypertextes du site sur lequel il se trouve vers les autres sites dans la liste. Il en choisit alors un au hasard et répète le processus sans s'arrêter en comptant pour chacun des sites combien de fois il l'a visité. Plus un site est visité de cette façon, plus sa note est grande.
Le début de l'exercice suivant montre comment on peut simuler le fonctionnement de ce principe.

► Exercice n°97

Dans le graphe orienté ci-dessous, les sommets représentent les pages répondant à une requête de recherche et une arête représente un hyperlien entre la page de départ et la page d'arrivée.



1. Compléter les phrases suivantes :

   - La page A contient un lien vers la(les) page(s) C
   - La page B contient un lien vers la(les) page(s) A,C
   - La page C contient un lien vers la(les) page(s) B,D,E,F
   - La page D contient un lien vers la(les) page(s) A,C
   - La page E contient un lien vers la(les) page(s) A,F
   - La page F contient un lien vers la(les) page(s) C

2. À l'aide d'un dé ou du script python ci-après, choisir une page de départ au hasard, puis suivre au hasard 20 fois un lien possible et compléter le tableau suivant :
   Page de départ : dépend de la simulation, comme le tableau

Pages	A	B	C	D	E	F
Nombre de fois où la page est visitée	…	…	…	…	…	…
Fréquence (en divisant par 20)	…	…	…	…	…	…

   Dernière page visitée : dépend de la simulation
   Page la mieux notée (qui figurera en tête de classement) : C

Script python

```
from random import choice
hyperLiens = { "A":["C"] ,"B":["A","C"],"C":["B","D","E","F"]
,"D":["A","C"],"E":["A","F"],"F": ["C"] }
nomsPages=[n for n in hyperLiens.keys()]

def parcours(nbEtapes,pause):
      nbVisites = {n:0 for n in hyperLiens}
      page=choice(nomsPages)
      print("Page de départ :",page)
      for i in range(nbEtapes):
            page=choice(hyperLiens[page])
            nbVisites[page] += 1
            if pause:
                  print("On passe à la page : " , page)
                  input()
      pageRank={n:nbVisites[n]/nbEtapes for n in nbVisites}
      return pageRank

print(parcours(20,True))
```

3. Compléter la matrice d'adjacence M du graphe :

$$M = \begin{array}{c} \begin{matrix} A & B & C & D & E & F \end{matrix} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array} \begin{matrix} A \\ B \\ C \\ D \\ E \\ F \end{matrix}$$

4. On donne ci-dessous M^{20}. En déduire le nombre de possibilités qu'il y avait lors du parcours du surfeur aléatoire à la question 2 pour aller de la première page à la dernière page de la simulation effectuée.
Réponse : dépend de la simulation

$$M^{20} = \begin{pmatrix} 809568 & 592291 & 592291 & 592291 & 1300644 & 862147 \\ 1177875 & 862147 & 862147 & 862147 & 1892935 & 1254772 \\ 736614 & 539712 & 539712 & 539712 & 1184582 & 785250 \\ 1177875 & 862147 & 862147 & 862147 & 1892935 & 1254772 \\ 1776873 & 1300644 & 1300644 & 1300644 & 2856297 & 1892935 \\ 809568 & 592291 & 592291 & 592291 & 1300644 & 862147 \end{pmatrix}$$

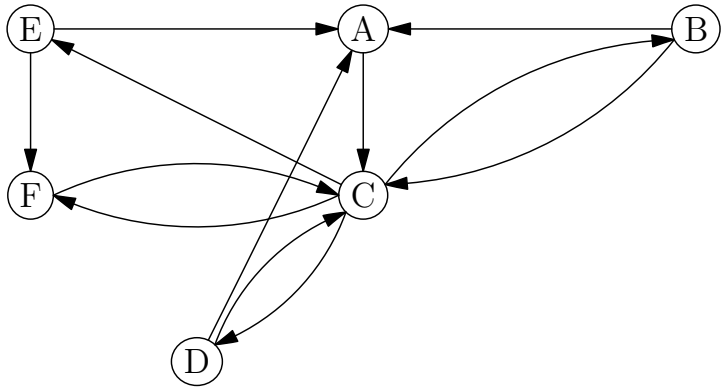
5. On donne ci-dessous M^4. En déduire le nombre de possibilités pour aller de la page E et de finir sur la page C en cliquant sur 4 liens et donner l'ensemble de ses possibilités.
Réponse : 6 (3ème coeff de la 5ème ligne)
E-A-C-B-C ; E-A-C-D-C ; E-A-C-F-C ; E-F-C-F-C ; E-F-C-D-C ; E-F-C-B-C

$$M^4 = \begin{pmatrix} 0 & 3 & 4 & 3 & 3 & 3 \\ 3 & 3 & 7 & 3 & 3 & 4 \\ 9 & 4 & 9 & 4 & 4 & 7 \\ 3 & 3 & 7 & 3 & 3 & 4 \\ 6 & 0 & 6 & 0 & 0 & 2 \\ 0 & 3 & 4 & 3 & 3 & 3 \end{pmatrix}$$

2. Parcours Eulériens

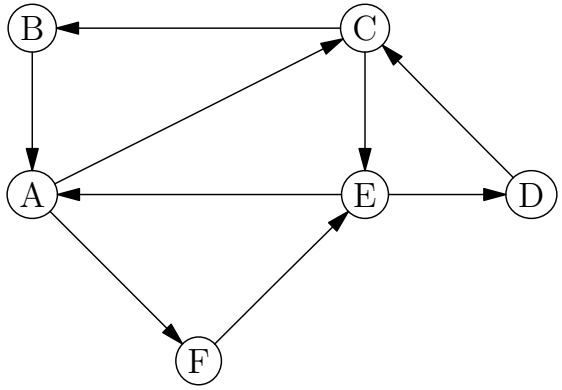
► Exercice n°98

1. On reprend le graphe de l'exercice précédent où les sommets représentent des pages web et une arête représente un hyperlien entre la page de départ et la page d'arrivée.



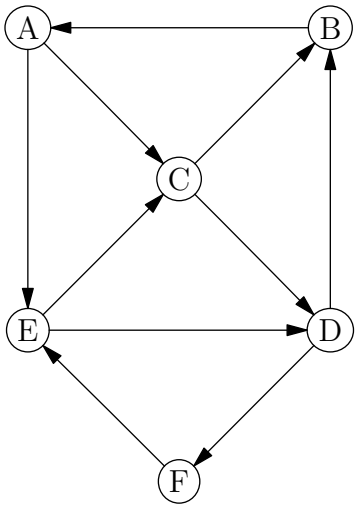
Est-il possible de partir d'une page et d'y revenir en cliquant une fois (et une fois seulement) sur chaque lien disponible ? Si oui, donner un exemple d'un tel parcours (note : un tel parcours est appelé circuit eulérien)
Réponse : impossible
Il faudrait pour cela que chaque sommet comporte autant d'arêtes entrantes que sortantes

2. Répondre à la même question avec la configuration des pages suivantes :



Réponse : oui A-C-E-D-C-B-A-F-E-A

► **Exercice n°99**
On considère le graphe ci-dessous où les sommets représentent des pages web et une arête représente un hyperlien entre la page de départ et la page d'arrivée.



Est-il possible de partir d'une page et d'arriver à une autre en cliquant une fois (et une fois seulement) sur chaque lien disponible ? Si oui, donner un exemple d'un tel parcours (note : un tel parcours est appelé chemin eulérien)
Réponse : oui entre A et B (A admet 2 arêtes sortantes pour une entrante et pour B c'est l'inverse)
A-C-D-F-E-C-B-A-E-D-B

3. Numération hexadécimale

Si le système binaire est à la base de l'informatique, il n'est pas toujours pratique et lisible pour les humains. C'est pour cela, qu'en informatique, on utilise aussi le système hexadécimal qui permet de représenter de façon plus courte les nombres.

- Le système décimal classique nécessite 10 chiffres : 0,1,2,3,4,5,6,7,8,9
- Le système binaire nécessite 2 chiffres : 0,1
- Le système hexadécimal nécessite 16 chiffres : 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

1) Conversion binaire - hexadécimal

Comme un groupe de 4 bits permet de représenter $2^4 = 16$ chiffres, pour passer du binaire à l'hexadécimal, il suffit d'utiliser la table suivante :

Code binaire	0000	0001	0010	0011	0100	0101	0110	0111
Chiffre hexadécimal	0	1	2	3	4	5	6	7

Code binaire	1000	1001	1010	1011	1100	1101	1110	1111
Chiffre hexadécimal	8	9	A	B	C	D	E	F

► **Exercice n°100**

1. a) Convertir en hexadécimal le nombre binaire 10100100. Réponse : A4
   b) Convertir en hexadécimal le nombre binaire 1101100. Réponse : 6C
   c) Convertir en hexadécimal le nombre binaire 1111100110. Réponse : 3E6
2. a) Convertir en binaire le nombre hexadécimal F3.
   Réponse : 11110011
   b) Convertir en binaire le nombre hexadécimal 123.
   Réponse : 100100011
   c) Convertir en binaire le nombre hexadécimal 8A4.
   Réponse : 100010100100

2) Conversion hexadécimal - décimal

Le principe est le même qu'avec la conversion décimale-binaire sauf qu'à la place de puissances de 2, on utilise des puissances de 16.
Ainsi :

- Le nombre hexadécimal BF correspond au nombre décimal 191 car $\boxed{11} \times 16 + \boxed{15} = 191$
- Le nombre hexadécimal 8A3 correspond au nombre décimal 2211 car $\boxed{8} \times 16^2 + \boxed{10} \times 16 + \boxed{3} = 2211$

▶ Exercice n°101

1. Donner l'écriture en système décimal du nombre hexadécimal 321 .

   Réponse : 801 car $\boxed{3}\times 16^2+\boxed{2}\times 16+\boxed{1}=801$

2. Donner l'écriture en système décimal du nombre hexadécimal E0A.

   Réponse : 3594 car $\boxed{14}\times 16^2+\boxed{0}\times 16+\boxed{10}=3594$

3. Quel est le plus grand nombre, en système décimal, que l'on peut obtenir à partir d'un nombre hexadécimal formé de 2 chiffres ?

   Réponse : FF en hexadécimal, ce qui correspond à $15\times 16+15=255$ en décimal

4. Donner l'écriture en système hexadécimal du nombre décimal 156 .

   Réponse : 9C car $9\times 16+12=156$

.....

3) Le codage des couleurs dans les pages Web

Une couleur dans le code html (ou css) d'une page web peut-être définie sous deux formes différentes :

- la forme `rgb(a,b,c)` où a, b et c sont des nombres, allant de 0 à 255, représentant l'intensité de la nuance de rouge, vert et bleu.
  Par exemple,
  — `rgb(255,0,0)` correspond à la couleur rouge
  — `rgb(0,0,255)` correspond à la couleur bleue
  — `rgb(255,255,255)` correspond à la couleur blanche
  — `rgb(0,0,0)` correspond à la couleur noire
  — `rgb(255,255,0)` correspond à la couleur jaune
- la notation hexadécimale `#xxyyzz` où `xx`, `yy` et `zz` représentent les intensités des nuances de rouge, vert et bleu en hexadécimal.
  Par exemple,
  — `#FF0000` correspond à `rgb(255,0,0)`
  — `#00FF00` correspond à `rgb(0,255,0)` et donc à la couleur verte
  — `#FFFFFF` correspond à `rgb(255,255,255)` et donc à la couleur blanche

▶ Exercice n°102

1. Convertir en notation rgb le code de couleur html `#A04C9D` :
   Réponse : `rgb(160,76,157)` car

   A0 correspond en décimal à $10\times 16+0=160$, 4C correspond en décimal à $4\times 16+12=76$ et 9D correspond en décimal à $9\times 16+13=157$

2. Convertir en notation rgb le code de couleur html `#5080DE` :
   Réponse : `rgb(80,128,222)` car

   50 correspond en décimal à $5\times 16+0=80$, 80 correspond en décimal à $8\times 16+0=128$ et DE correspond en décimal à $13\times 16+14=222$

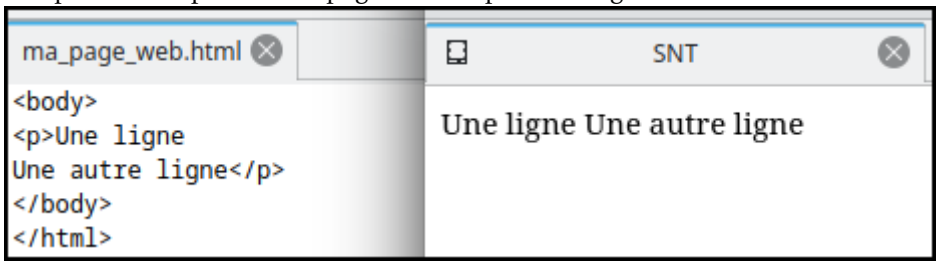
3. Convertir en notation hexadécimale le code de couleur html `rgb(127,10,0)` :
   Réponse : #7F0A00 car :
   La conversion de 127 en hexadécimal donne 7F ($7\times 16+15=127$)
   La conversion de 10 en hexadécimal sur 2 chiffres donne 0A

4. Convertir en notation hexadécimale le code de couleur html `rgb(248,32,32)` :
   Réponse : #F82020 car :
   La conversion de 248 en hexadécimal donne F8 ($15\times 16+8=248$)
   La conversion de 32 en hexadécimal donne 20 ($2\times 16+0=32$)

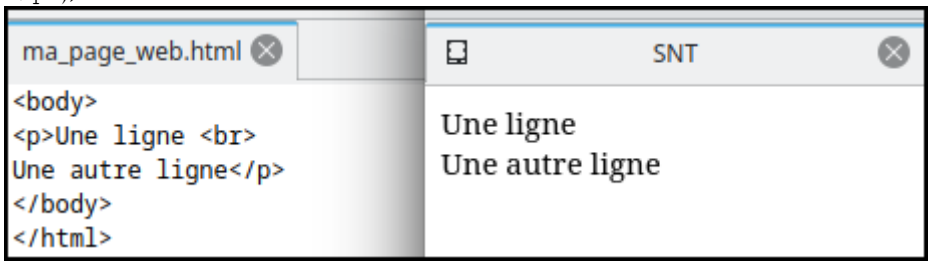
4. Compléments sur le HTML

1) Retour à la ligne en html

Effectuer un retour à la ligne dans le code d'une page html ne suffit pas pour qu'il soit pris en compte dans la page affichée par le navigateur :



Pour qu'un retour à la ligne soit affiché dans un paragraphe (entre les balises `<p>` et `</p>`), il faut utiliser la commande `<br>` :



2) Listes en html

- Pour afficher une **liste non ordonnée**, il faut l'introduire dans le code avec les balises `<ul></ul>`. Chaque élément de la liste doit alors être encadrée par les balises `<li></li>`.

ma_page_web.html

```
<body>
Une liste non ordonnée
<ul>
    <li>Premier élément</li>
    <li>Deuxième élément</li>
</ul>
</body>
```

SNT

Une liste non ordonnée

- Premier élément
- Deuxième élément

- Pour afficher une **liste ordonnée**, il faut l'introduire dans le code avec les balises `<ol></ol>`. Chaque élément de la liste doit alors être encadrée par les balises `<li></li>`.

ma_page_web.html

```
<body>
Une liste ordonnée
<ol>
    <li>Premier élément</li>
    <li>Deuxième élément</li>
</ol>
</body>
```

SNT

Une liste ordonnée

1. Premier élément
2. Deuxième élément

► Exercice n°103

Compléter le code source ci-dessous, pour que le résultat affiché dans un navigateur soit conforme à la copie d'écran ci-contre :

```
<body>
<h1>Ma recette</h1>
<p>Pour cette <b>délicieuse</b> recette , il faut :<br>
(bien noter la liste)
</p>
<ol>
      <li> de la crème <i>fouettée</i> </li>
      <li> de la farine </li>
</ol>
</body>
```

SNT

Ma recette

Pour cette **délicieuse** recette , il faut :
(bien noter la liste)

1. de la crème *fouettée*
2. de la farine

3) Caractères spéciaux

Comme les caractères < et > sont utilisés pour les balises html, on ne peut pas les afficher dans une page web en les incluant directement dans le code html de la page.
Exemple (le code n'est pas rendu correctement dans le navigateur) :

ma_page_web.html

```
<body>
x<y ?
</body>
```

SNT

x

La solution pour afficher les caractères < et > dans une page web est de les remplacer dans le code html par `&lt;` et `&gt;` .
En reprenant l'exemple précédent :

ma_page_web.html

```
<body>
x &lt; y ?
</body>
```

SNT

x < y ?

4) Complément sur les styles de texte en html

Nous avons déjà vu les balises `<b></b>` (pour mettre en gras) et `<i></i>` pour mettre du texte en gras et en italique.
Il existe aussi la balise `<u></u>` pour souligner du texte et la balise `<s></s>` pour du texte barré.
Il est de plus possible de mixer ces effets de texte en imbriquant les balises.

Exemple :

ma_page_web.html

```
<body>
Dans une <u>lointaine</u> <i><b>galaxie</b></i>
</body>
```

SNT

Dans une lointaine ***galaxie***

► Exercice n°104

Compléter le code source ci-dessous, pour que le résultat affiché dans un navigateur soit conforme à la copie d'écran :

```
<body>
<p>
Les <b><i>inégalités</i></b> x &gt; 3 et y &lt; 7 sont-elles vérifiées?<br>
<i>(justifer sa réponse)</i>
</p>
<ol>
<li><u><i>OUI</i></u></li>
<li><s>NON</s></li>
</ol>
</body>
```

SNT

Les ***inégalités*** x > 3 et y < 7 sont-elles vérifiées?
(justifer sa réponse)

1. *OUI*
2. ~~NON~~

► Exercice n°105

Pour chaque question, indiquer la seule affirmation exacte parmi celles proposées :

1. Quelle balise faut-il utiliser pour un titre principal ?

A : `<head></head>`
B : `<h1></h1>`
C : `<title></title>`

2. Quel est le code valable pour créer un lien vers `https://monsite.com` ?

A : `<a src="https://monsite.com">lien</a>`
B : `<a target="https://monsite.com">lien</a>`
C : `<a href="https://monsite.com">lien</a>`

3. Quelles balises faut-il utiliser pour créer une liste ordonnée ?

A : `<ul></ul>` et `<li></li>`
B : `<ol></ol>` et `<li></li>`
C : `<ul></ul>` et `<ol></ol>`

4. HTML est un acronyme pour :

A : HyperText Markup Language
B : HyperText Markup Link
C : HyperText Make Link

5. La notation hexadécimale du code de couleur html `rgb(56,206,10)` est :

A : `#38CEAA`
B : `#36DE0A`
C : `#38CE0A`

6. La conversion en notation `rgb()` du code de couleur hexadécimal `#B0FE23` est :

A : `rgb(176,254,35)`
B : `rgb(B0,FE,23)`
C : `rgb(192,255,37)`

7. Le code valable pour insérer une image est :

A : `<img src="monimage.jpg"></img>`
B : `<img src="monimage.jpg">`
C : `<img href="monimage.jpg"></img>`

8. Quel code faut-il utiliser pour afficher un mot en italique et souligné ?

A : `<i><u>mot</u></i>`
B : `<b><u>mot</u></b>`
C : `<s><i>mot</i></s>`

9. Quel code faut-il utiliser pour que le navigateur affiche `< hello >` ?

A : `< hello >`
B : `> hello <`
C : `< hello >`

10. Le code d'une page web au format html doit-être enregistré avec l'extension

A : `.html`
B : `.web`
C : `.css`

11. Laquelle de ces balises insérera un saut de ligne ?

A : `<newline>`
B : ` `
C : `<line>`

12. Quel code permettra-t'il au navigateur d'afficher `blabla` ?

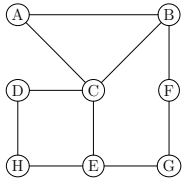
A : `<head><p>blabla</p></head>`
B : `<body><p>blabla</p></body>`
C : `<htm><p>blabla</p></htm>`

5. Exemple de table de routage

Dans un réseau de routeurs, la table de routage indique, pour chaque routeur, comment envoyer des données à un autre routeur.

▶ Exercice n°106

On considère le réseau de routeurs et sa table de routage ci-dessous :



Routeur A		Routeur B		Routeur C		Routeur D	
Pour	Passer par	Pour	Passer par	Pour	Passer par	Pour	Passer par
B	B	A	A	A	A	A	C
C	C	C	C	B	B	B	C
D	C	D	C	D	D	C	C
E	C	E	C	E	E	E	H
F	B	F	F	F	B	F	C
G	C	G	F	G	E	G	H
H	C	H	C	H	E	H	H

Routeur E		Routeur F		Routeur G		Routeur H	
Pour	Passer par	Pour	Passer par	Pour	Passer par	Pour	Passer par
A	C	A	B	A	F	A	D
B	C	B	B	B	F	B	D
C	H	C	G	C	E	C	D
D	H	D	B	D	E	D	D
F	G	E	G	E	E	E	E
G	G	G	G	F	F	F	E
H	H	H	G	H	E	G	E

1. Quel parcours doivent suivre des données pour aller du routeur E au routeur G ?
   Réponse : E-G
2. Quel parcours doivent suivre des données pour aller du routeur E au routeur F ?
   Réponse : E-G-F
3. Quel parcours doivent suivre des données pour aller du routeur A au routeur G ?
   Réponse : A-C-E-G
4. Quel parcours doivent suivre des données pour aller du routeur F au routeur C ?
   Réponse : F-G-E-H-D-C

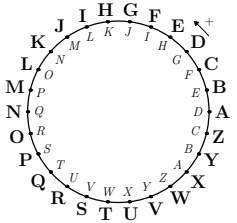
6. Un peu de cryptographie

1) Vocabulaire de base

- L'action de transformer un texte compréhensible par tous en un autre incompréhensible (sauf pour le destinataire autorisé) s'appelle chiffrement ;
- L'action inverse permettant au destinataire autorisé de reconstituer le texte original s'appelle déchiffrement ;
- L'action consistant par une personne non autorisée à retrouver le texte original s'appelle décryptage ou cryptanalyse ;

2) Le chiffre de César

Une des premières méthodes pour chiffrer a consisté à remplacer les caractères du texte par d'autres dans le même alphabet (chiffrement par substitution).
Le principe général de la méthode utilisée par Jules César consiste à remplacer chaque lettre par la lettre située située à un certain rang fixe plus loin dans l'alphabet. Pour un décalage de 3 rangs (utilisé historiquement par Jules César), la situation peut se représenter selon le schéma circulaire suivant (à l'extérieur se trouve la lettre d'origine et à l'intérieur la lettre de substitution) :



▶ Exercice n°107

`DWWDTXHDPLGL` est le texte chiffré par la méthode de César avec un décalage de 3 rangs. Quel était le texte original ?
Réponse : `ATTAQUEAMIDI`

▶ Exercice n°108

1. Pour un alphabet ne comportant que les lettres en majuscule de `A` à `Z`, combien de décalage sont possibles avec la méthode de chiffrement de César ?
   Réponse : 25
2. `YLUMVYAZCPLUKYVUAKLTHPU` est un message chiffré par la méthode de César avec un décalage inconnu. Quel était le texte original ?
   Réponse : `RENFORTSVIENDRONTDEMAIN`

► Codage en python possible pour le (dé)chiffrement de César :

Script python

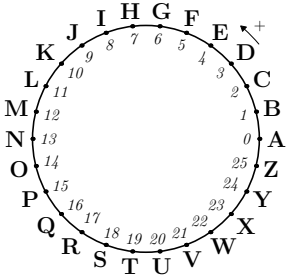
```
def chiffrer(message_en_clair,cle):
      message_chiffre=""
      for i in range(len(message_en_clair)):
            message_chiffre+=chr((ord(message_en_clair[i])-65+cle)%26+65)
      return message_chiffre

def dechiffrer(message_chiffre,cle):
      message_dechiffre=""
      for i in range(len(message_chiffre)):
            message_dechiffre+=chr((ord(message_chiffre[i])-65-cle)%26+65)
      return message_dechiffre
```

3) Le chiffre de Vigenère

La méthode de chiffrement de Vigenère est aussi un chiffrement par substitution mais avec un décalage non fixe.
Principe :

- Le chiffre de Vigenère utilise un mot clé que l'on répète (en dessous du texte à chiffrer) autant de fois que nécessaire afin d'avoir la même longueur que le message à chiffrer. Exemple avec comme message LEMESSAGEACRYPTER et comme clef MACLEF :
  LEMESSAGEACRYPTER
  MACLEFMACLEFMACLE
- La lettre du mot clé figurant dans la deuxième ligne indique le décalage à apporter à la lettre du message située au dessus. Exemple :
  — Si la lettre de la clef est un A, on ne décale pas la lettre correspondante du message ;
  — Si la lettre de la clef est un B, on décale d'un rang dans l'alphabet la lettre correspondante du message : ainsi un A dans le message devient un B, un B devient un C, etc...
  — Si la lettre de la clef est un C, on décale de 2 rangs dans l'alphabet la lettre correspondante du message : ainsi un A dans le message devient un C, un B devient un D, etc...
  — etc...
- Avec notre exemple : la première lettre du message est un L ; la lettre correspondante de la clef est un M qui représente un décalage de 12. La première lettre du message crypté est donc un X (12 lettres plus loin que L dans l'alphabet).

► **Exercice n°109**
On reprend l'exemple précédent :
LEMESSAGEACRYPTER
MACLEFMACLEFMACLE
Les 10 premières lettres du message crypté sont XEOPWXMGGL.
Donner les sept dernières lettres (correspondantes à CRYPTER) :
Réponse : GWKPVPV
On pourra s'aider de l'alphabet en cercle ci-dessous :



► **Exercice n°110**
Pour le déchiffrement dans le procédé de Vigenère, on applique le même principe que pour le chiffrement sauf que le décalage pour le déchiffrement est égal à 26 moins le décalage pour le chiffrement, ce qui revient à tourner dans le sens contraire sur le cercle de l'exercice précédent.
Ci-dessous figure un message crypté avec toujours comme clef MACLEF :
DDXLQNPI
MACLEFMA
Déchiffrer le message. Réponse : RDVAMIDI

► Codage en python possible pour le (dé)chiffrement de Vigenère :

Script python

```
def chiffrer(message_en_clair,cle):
      message_chiffre=""
      for i in range(len(message_en_clair)):
            decalage=ord(cle[i%len(cle)])-65
            message_chiffre+=chr((ord(message_en_clair[i])-65+decalage)%26+65)
      return message_chiffre

def dechiffrer(message_chiffre,cle):
      message_dechiffre=""
      for i in range(len(message_chiffre)):
            decalage=26-(ord(cle[i%len(cle)])-65)
            message_dechiffre+=chr((ord(message_chiffre[i])-65+decalage)%26+65)
      return message_dechiffre
```

7. Graphes probabilistes

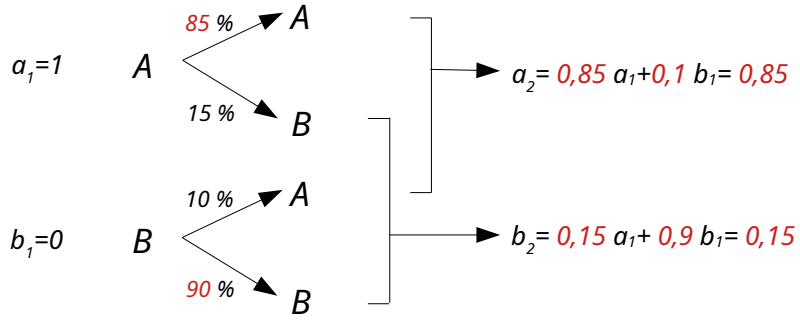
► **Exemple :** Dans un jeu vidéo, une suite d'énigmes est proposée au joueur. Ces énigmes sont classées en deux catégories : les énigmes de catégorie A sont les énigmes faciles ; les énigmes de catégorie B sont les énigmes difficiles.
Le choix des énigmes successives est aléatoire et vérifie les conditions suivantes :

- la première énigme est facile ;
- si une énigme est facile, il y a 15% de chances que la suivante soit difficile ;
- si une énigme est difficile, il y a 10% de chances que la suivante soit facile.

Pour tout entier $n \geqslant 1$, on note :

- a_n la probabilité que l'énigme numéro n soit facile (de catégorie A) ;
- b_n la probabilité que l'énigme numéro n soit difficile (de catégorie B) ;
- P_n la matrice ligne, appelée état probabiliste pour l'énigme numéro n, définie par $P_n = \begin{pmatrix} a_n & b_n \end{pmatrix}$.

1. Compléter le schéma ci-dessous et indiquer la valeur de P_2 :

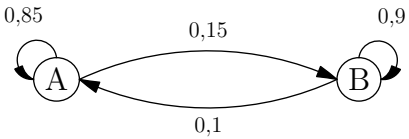


$P_1 = (1\ 0)$ $\qquad$ $P_2 = (0{,}85 \quad 0{,}15)$

2. En déduire les coefficients à inclure dans la matrice ci-dessous pour que le calcul soit correct :
$(a_2\ b_2) = (a_1\ b_1)\begin{pmatrix} 0{,}85 & 0{,}15 \\ 0{,}1 & 0{,}9 \end{pmatrix}$.

3. Le mécanisme étant le même d'une énigme sur l'autre, pour avoir l'état probabiliste pour l'énigme numéro 3, il suffit de remultiplier par la même matrice. Compléter le calcul ci-dessous :
$(a_3\ b_3) = (a_2\ b_2)\begin{pmatrix} 0{,}85 & 0{,}15 \\ 0{,}1 & 0{,}9 \end{pmatrix} = (0{,}85\ 0{,}15)\begin{pmatrix} 0{,}85 & 0{,}15 \\ 0{,}1 & 0{,}9 \end{pmatrix} = (0{,}7375\ 0{,}2625)$

4. Dans le graphe ci-dessous, les sommets représentent les catégories A (faciles) et B (difficile) des énigmes. Indiquer sur les arêtes les probabilités de passer de la catégorie du sommet de départ à celle du sommet d'arrivée.



Note : un tel graphe est appelé graphe probabiliste

5. Compléter la matrice M ci-dessous en insérant les probabilités de passer de la catégorie de la ligne à celle de la colonne :

$$M = \begin{array}{c} \\ A \\ B \end{array} \begin{array}{c} \begin{array}{cc} A & B \end{array} \\ \begin{pmatrix} 0{,}85 & 0{,}15 \\ 0{,}1 & 0{,}9 \end{pmatrix} \end{array}$$

Que retrouve t-on ? la matrice de la question 2
Remarques :

- cette matrice est appelée matrice de transition
- On a donc $P_2 = P_1 M$, $P_3 = P_2 M = P_1 MM = P_1 M^2$

6. Si la matrice de transition M n'admet aucun coefficient nul, la probabilité a_n se rapproche de plus en plus d'une valeur a telle que $(a \quad 1-a) = (a \quad 1-a)M$.
On peut donc savoir vers quelles valeurs les probabilités vont se stabiliser au bout de plusieurs étapes et la matrice $P = (a \quad 1-a)$ est appelée état stable.
Déterminer ci-dessous la valeur de a vers laquelle tend la probabilité a_n que l'énigme soit facile :

$$(a \quad 1-a) = (a \quad 1-a)\begin{pmatrix} 0{,}85 & 0{,}15 \\ 0{,}1 & 0{,}9 \end{pmatrix} \Rightarrow a = 0{,}85a + 0{,}1(1-a)$$

$$\Rightarrow a = 0{,}85a + 0{,}1 - 0{,}1a \Rightarrow a - 0{,}85a + 0{,}1a = 0{,}1 \Rightarrow 0{,}25a = 0{,}1 \Rightarrow a = \frac{0{,}1}{0{,}25} = 0{,}4$$

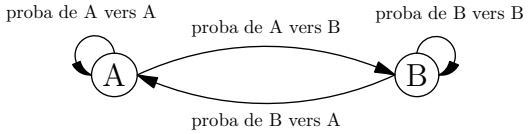
Une revue spécialisée dans les jeux vidéo indique qu'en évoluant dans le jeu, la part des énigmes difficiles devient majoritaire. La revue a-'telle raison ? oui

Vocabulaire

• On appelle **graphe probabiliste**, tout graphe orienté pondéré tel que la somme des poids des arêtes issues de chaque sommet soit égale à 1.
• La **matrice de transition** M d'un graphe probabiliste à n sommets est une matrice carrée de n lignes et de n colonnes où le terme situé à la $i^{\text{ème}}$ ligne et à la $j^{\text{ème}}$ colonne est égal au poids de l'arête orientée allant du sommet i au sommet j si elle existe et à 0 si elle n'existe pas. *(ne pas confondre avec la matrice associée)*

Cas des graphes probabilistes à 2 sommets

• Étant donné une entité ne connaissant que deux états possibles A et B et pouvant passer aléatoirement d'un état à l'autre selon des probabilités connues. On peut modéliser la situation avec un graphe probabiliste où les deux sommets représentent les deux états possibles et où le poids d'une arête représente la probabilité de passage d'un état à un autre :



• Si on note
- M la matrice de transition du graphe probabiliste ;
- a_n la probabilité d'être à l'état A à l'étape numéro n ;
- b_n la probabilité d'être à l'état B à l'étape numéro n ;
- P_n la matrice ligne, appelée état probabiliste à l'étape numéro n, définie par $P_n = \begin{pmatrix} a_n & b_n \end{pmatrix}$.

On a $P_2 = P_1 M$, $P_3 = P_2 M$, etc.

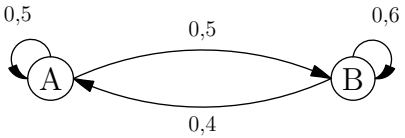
• Si la matrice de transition M n'admet aucun coefficient nul, la probabilité a_n se rapproche de plus en plus d'une valeur a telle que $(a \quad 1-a) = (a \quad 1-a)M$. La matrice $P = (a \quad 1-a)$ est alors appelée l'état stable associé à la situation.

► Exercice n°111

Une étude statistique d'un société de commerce en ligne indique que :
- Si un client effectue une commande une semaine, il y a 50% de chances qu'il en refasse une semaine suivante ;
- Si un client n'effectue pas de commande une semaine, il y a 60% de chances qu'il n'en fasse pas non plus la semaine suivante.

Pour tout entier $n \geqslant 1$, on note :
- a_n la probabilité qu'un client effectue une commande la semaine numéro n (état A) ;
- b_n la probabilité qu'un client n'effectue pas de commande la semaine numéro n (état B) ;
- $P_n = \begin{pmatrix} a_n & b_n \end{pmatrix}$, l'état probabiliste pour la semaine numéro n.

1. Compléter le graphe probabiliste ci-dessous correspondant à la situation :



2. Indiquer ci-dessous la matrice de transition du graphe :

$M = \begin{pmatrix} 0,5 & 0,5 \\ 0,4 & 0,6 \end{pmatrix}$

3. On suppose qu'un client passe une commande la semaine numéro 1 : on a donc $P_1 = (1 \; 0)$.
Calculer ci-dessous P_2 et P_3 :

$P_2 = P_1 M = (1 \; 0)\begin{pmatrix} 0,5 & 0,5 \\ 0,4 & 0,6 \end{pmatrix} = (0,5 \; 0,5)$

$P_3 = P_2 M = (0,5 \; 0,5)\begin{pmatrix} 0,5 & 0,5 \\ 0,4 & 0,6 \end{pmatrix} = (0,45 \; 0,55)$

4. Déterminer ci-dessous l'état stable $P = (a \quad 1-a)$ associé à la situation :

$(a \quad 1-a) = (a \quad 1-a)\begin{pmatrix} 0,5 & 0,5 \\ 0,4 & 0,6 \end{pmatrix} \Rightarrow a = 0,5a + 0,4(1-a)$

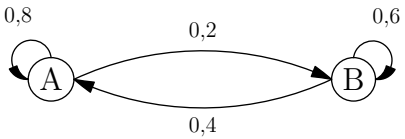
$\Rightarrow a = 0,5a + 0,4 - 0,4a \Rightarrow a - 0,5a + 0,4a = 0,4 \Rightarrow 0,9a = 0,4 \Rightarrow a = \frac{0,4}{0,9} \approx 0,44$

► Exercice n°112

Une étude de qualité sur des serveurs indique que :
- Si un serveur à un taux de satisfaction des requêtes de 100% un jour, il y a 20% de chances que ce taux ne soit pas de 100% le lendemain ;
- Si un serveur n'a pas satisfait à 100% des requêtes un jour, il y a 40% de chances qu'il y satisfasse le lendemain.

Pour tout entier $n \geqslant 1$, on note :
- a_n la probabilité que le serveur réponde à 100% des requêtes le jour numéro n (état A) ;
- b_n la probabilité que serveur ne réponde pas à 100% des requêtes le jour numéro n (état B) ;
- $P_n = \begin{pmatrix} a_n & b_n \end{pmatrix}$, l'état probabiliste pour le jour numéro n.

1. Compléter le graphe probabiliste ci-dessous correspondant à la situation :



2. Indiquer ci-dessous la matrice de transition du graphe :

$M = \begin{pmatrix} 0,8 & 0,2 \\ 0,4 & 0,6 \end{pmatrix}$

3. On suppose qu'un serveur n'a pas satisfait à 100% des requêtes le jour numéro 1 : on a donc $P_1 = (0\ 1)$.
Calculer ci-dessous P_2 et P_3 :

$P_2 = P_1 M = (0\ 1)\begin{pmatrix} 0,8 & 0,2 \\ 0,4 & 0,6 \end{pmatrix} = (0,4\ 0,6)$

$P_3 = P_2 M = (0,4\ 0,6)\begin{pmatrix} 0,8 & 0,2 \\ 0,4 & 0,6 \end{pmatrix} = (0,56\ 0,44)$

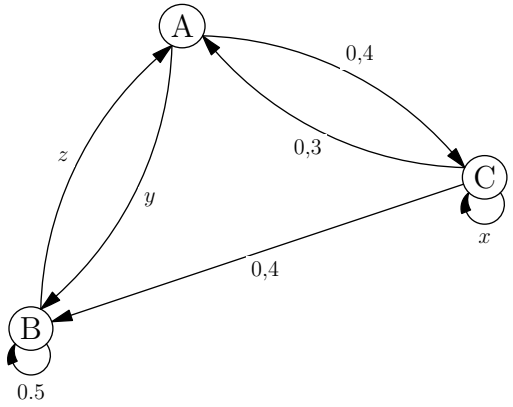
4. Déterminer ci-dessous l'état stable $P = (a \quad 1-a)$ associé à la situation :

$(a \quad 1-a) = (a \quad 1-a)\begin{pmatrix} 0,8 & 0,2 \\ 0,4 & 0,6 \end{pmatrix} \Rightarrow a = a = 0,8a + 0,4(1-a)$

$\Rightarrow a = 0,8a + 0,4 - 0,4a \Rightarrow a - 0,8a + 0,4a = 0,4 \Rightarrow 0,6a = 0,4 \Rightarrow a = \frac{0,4}{0,6} \approx 0,67$

▶ Exercice n°113

1. Déterminer les valeurs de x, y et z pour que le graphe ci-dessous soit un graphe probabiliste.



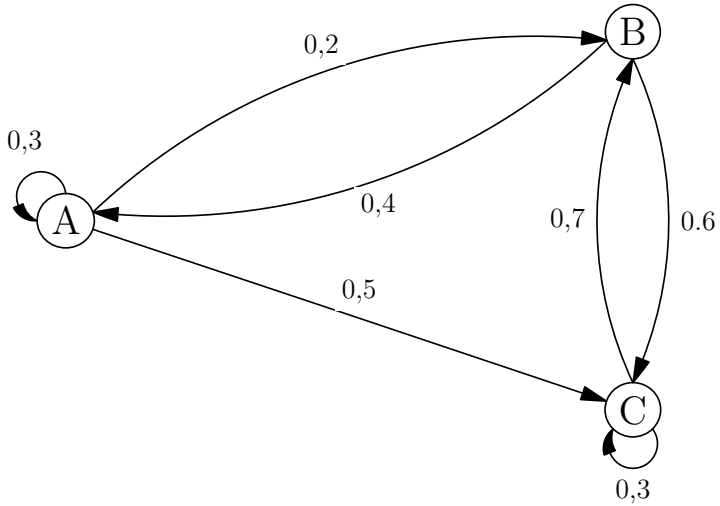
Arêtes issues de A : on doit avoir $0,4 + y = 1 \Leftrightarrow y = 0,6$
Arêtes issues de B : on doit avoir $0,5 + z = 1 \Leftrightarrow z = 0,6$
Arêtes issues de C : on doit avoir $0,3 + 0,4 + x = 1 \Leftrightarrow x = 0,3$

2. Déterminer la matrice de transition du graphe.

$M = \begin{pmatrix} 0 & 0,6 & 0,4 \\ 0,5 & 0,5 & 0 \\ 0,3 & 0,4 & 0,3 \end{pmatrix}$

▶ Exercice n°114

On considère le graphe ci-dessous :



1. Justifier qu'il s'agit d'un graphe probabiliste :

Somme des poids des arêtes issues de A : $0,3 + 0,2 + 0,5 = 1$
Somme des poids des arêtes issues de B : $0,4 + 0 + 0,6 = 1$
Somme des poids des arêtes issues de C : $0 + 0,7 + 0,3 = 1$

2. Déterminer la matrice de transition du graphe.

$M = \begin{pmatrix} 0,3 & 0,2 & 0,5 \\ 0,4 & 0 & 0,6 \\ 0 & 0,7 & 0,3 \end{pmatrix}$

3. On suppose que l'état probabiliste initial est $P_1 = (1\ 0\ 0)$. Calculer les états probabilistes P_2 et P_3.

$P_2 = P_1 M = (1\ 0\ 0)\begin{pmatrix} 0,3 & 0,2 & 0,5 \\ 0,4 & 0 & 0,6 \\ 0 & 0,7 & 0,3 \end{pmatrix} = (0,3\ 0,2\ 0,5)$

$P_3 = P_2 M = (0,3\ 0,2\ 0,5)\begin{pmatrix} 0,3 & 0,2 & 0,5 \\ 0,4 & 0 & 0,6 \\ 0 & 0,7 & 0,3 \end{pmatrix} = (0,17\ 0,41\ 0,42)$

► Exercice n°115

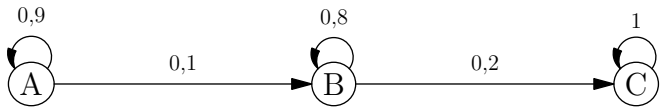
Une société d'autoroute étudie l'évolution de l'état de ses automates de péage en l'absence de maintenance. Un automate peut se trouver dans l'un des états suivants :

- fonctionnel (état A) ;
- en sursis (état B) s'il fonctionne encore, mais montre des signes de faiblesse ;
- défaillant (état C) s'il ne fonctionne plus.

La société a observé que d'un jour sur l'autre :

- concernant les automates fonctionnels, 90 % le restent et 10 % deviennent en sursis ;
- concernant les automates en sursis, 80 % le restent et 20 % deviennent défaillants.

1. Compléter le graphe probabiliste ci-dessous correspondant à la situation :



2. Indiquer ci-dessous la matrice de transition du graphe :

$$M = \begin{pmatrix} 0,9 & 0,1 & 0 \\ 0 & 0,8 & 0,2 \\ 0 & 0 & 1 \end{pmatrix}$$

3. la société observe qu'au début de l'expérience tous ses automates sont fonctionnels : on a donc $P_1 = (1\,0\,0)$. Calculer les états probabilistes P_2 et P_3.

$$P_2 = P_1 M = (1\,0\,0)\begin{pmatrix} 0,9 & 0,1 & 0 \\ 0 & 0,8 & 0,2 \\ 0 & 0 & 1 \end{pmatrix} = (0,9\;0,1\;0)$$

$$P_3 = P_2 M = (0,9\;0,1\;0)\begin{pmatrix} 0,9 & 0,1 & 0 \\ 0 & 0,8 & 0,2 \\ 0 & 0 & 1 \end{pmatrix} = (0,81\;0,17\;0,02)$$

4. Soit $P = (0\,0\,1)$. Calculer PM ci-dessous :

$$PM = (0\,0\,1)\begin{pmatrix} 0,9 & 0,1 & 0 \\ 0 & 0,8 & 0,2 \\ 0 & 0 & 1 \end{pmatrix} = (0\,0\,1)$$

Que peut-on en conclure (on admettra que P est l'unique état stable) :
Réponse : à long terme la quasi-totalité des automates seront défaillants

8. Les métiers du numérique

Famille de métiers	Intitulé des métiers
Programmation et développements	Développeur Ingénieur étude et développement Architecte logiciel
Métiers de l'intelligence artificielle et de la donnée	Administrateur de base de données Data analyst Data scientist
Infrastructures, clouds, réseaux et data centers	Technicien Cloud et réseaux Ingénieur Cloud et réseaux Architecte Cloud et réseaux / Urbaniste
Maintenance, assistance et support pour l'exploitation	Technicien de maintenance, support et services aux utilisateurs en informatique
Interfaces utilisateurs et créations numériques	Web designer Designer d'expérience et d'interface Ergonome Directeur artistique Showrunner / story architect / Transmedia producer
Direction , management et stratégie	Manager de projet / d'équipe Responsable de la stratégie / de la prospective Chargé de relations avec l'écosystème Coach agile, product owner Business analyst Directeur des systèmes d'information Responsable sécurité des système d'information
Communication et marketing	Community manager Social Media Manager Marketeur digital Chargé de référencement Analyste de trafic
Expertise et conseil	Responsable cybersécurité Consultant/expert métier Expert en protection des données Expert en propriété intellectuelle

(Source : groupe de travail VPPEC numérique 2017)