Chapitre 3

Variables et instructions élémentaires

L'objectif de ce chapitre est de décrire des algorithmes en langage naturel ou dans un langage de programmation ; d'en réaliser quelques uns à l'aide d'un programme simple écrit dans un langage de programmation textuel et d'interpréter, compléter ou modifier des algorithmes plus complexes.

Le saviez-vous ?

L'origine du mot algorithme vient du nom d'un mathématicien persan al-Khuwarizmi(780-850) dont le traité d'algèbre décrit des procédés de calcul à suivre étape par étape pour résoudre des problèmes qui se ramènent souvent à la résolution d'équation.
Dès l'Antiquité, des algorithmes sont connus comme par exemple celui d'Euclide($\approx$ 300 av. J.-C.) qui permet le calcul du PGCD de deux entiers, ou bien celui d'Archimède pour l'approximation du nombre π.

1. Mise en route

a) Sans ordinateur, ni calculatrice

1. Sur une ligne, écrire 3 nombres.
2. Comparer les deux nombres de gauche, si celui le plus à gauche est le plus grand, ne rien faire, sinon échanger leur position.
3. Comparer les deux nombres de droite, si celui le plus à gauche est le plus grand, ne rien faire, sinon échanger leur position.
4. Comparer les deux nombres de gauche, si celui le plus à gauche est le plus grand, ne rien faire, sinon échanger leur position.
5. Choisir 3 autres nombres et appliquer les mêmes consignes.
6. Que remarque-t-on à la fin de l'algorithme ?

b) Avec ordinateur ou calculatrice

On propose trois algorithmes et trois programmes.

Labo 1.

Algorithme 1

$a \leftarrow 3$
$b \leftarrow 1$
$a \leftarrow 2 \times a - 2 \times b$
$b \leftarrow 4 \times a - 2$
Afficher le message $b =$
Afficher b

Algorithme 2

$a \leftarrow 3$
$b \leftarrow 5 \times a + 2$
Afficher b

Algorithme 3

$a \leftarrow 3$
$b \leftarrow 5 \times a + 2$
Afficher le message b

Prog. 1

```
a = 3
b = 5*a+2
print(b)
```

Prog. 2

```
a = 3
b = 5*a+2
print("b")
```

Prog. 3

```
a = 3
b = 1
a = 2*a-2*b
b = 4*a-2
print("b = ",b)
```

1. À chaque algorithme associez un programme.
2. Écrire et exécuter les programmes 1 et 2.
3. Quel est le résultat affiché par chacun des programmes. Sont-ils identiques ?
4. Que permettent les guillemets ?
5. Lire le programme 3 et annoncer ce qu'il va afficher. Le saisir à la calculatrice puis vérifier le résultat.

c) Mais qu'est ce que c'est ?


Labo 2.

On propose les trois programmes suivants :

Prog. 1

```
a = 3
b = a*2
c = a*2.0
d = a/3*6
print("b = ",b," c = ",c," d = ",d)
```

Prog. 2

```
a = "3"
b = a*2
print(b)
```

Prog. 3

```
a = 4/3
b = int(a)
c = float(a)
print("b = ",b," c = ",c)
```

1. Écrire et exécuter le programme 1.
   a) Quels sont les résultats affichés ? Sont-ils identiques ? Pourquoi ?
   b) Saisir l'instruction **type(b)** puis valider. Quelle information obtient-on en retour ?
   c) En est-il de même pour les variables c et d ? Pourquoi ?
2. Écrire et exécuter le programme 2.
   a) La variable b a-t-elle la même valeur que dans le précédent programme ?
   b) Si l'on modifie la première instruction par **a="z"**, qu'obtient-on pour b ?
3. Écrire et exécuter le programme 3.
   a) Comment expliquer la différence de résultat entre b et c ?
   b) Modifier la valeur de a en prenant 5/3. Les résultats sont-ils différents ?

d) Et si ...


Labo 3.

Soit l'algorithme et les programmes suivants :

Saisir a, b, c
si $a<b$ **alors**
  d=a
  a=b
  b=d
fin
si $b<c$ **alors**
  d=b
  b=c
  c=d
fin
si $a<b$ **alors**
  d=a
  a=b
  b=d
fin
Afficher a, b, c

Prog. 1

```
a=float(input("saisir un nombre"))
b=float(input("saisir un nombre"))
c=float(input("saisir un nombre"))
if a<b :
    a,b=b,a
if b<c :
    b,c=c,b
if a<b :
    a,b=b,a
print(a,b,c)
```

Prog. 2

```
def Tri(a,b):
    if a<b :
        a,b=b,a
    return(a,b)

a=float(input("saisir un nombre"))
b=float(input("saisir un nombre"))
c=float(input("saisir un nombre"))
a,b=Tri(a,b)
b,c=Tri(b,c)
a,b=Tri(a,b)
print(a,b,c)
```

1. Que permet de réaliser l'algorithme ?
2. Au sujet du programme 1.
   a) Ce programme correspond il à l'algorithme ?
   b) Écrire et exécuter le programme. Que réalise-t-il ?
   c) Comment est programmé l'échange de valeurs ?
   d) Suivant les valeurs saisies pour a, b et c ; toutes les lignes du programme sont-elles exécutées ? Pourquoi ?
3. Écrire et exécuter le programme 2.
   a) Que réalise-t-il ?
   b) En quoi est-il différent du précédent ? Quel intérêt ?
   c) Si l'on désire trier les valeurs dans l'ordre croissant, lequel des deux programmes est-il le plus simple de modifier ? Réaliser la modification et tester le nouveau programme.

2. Algorithmes

Définition

Un algorithme est une suite d'instructions élémentaires s'appliquant dans un ordre déterminé à des données et fournissant en un nombre fini d'étapes des résultats.

Exemple : choisir un nombre, puis multiplier ce nombre par 5, enfin ajouter 2 au résultat. Soit 9, puis $9 \times 5 = 45$, *enfin* $45 + 2 = 47$.

Pour stocker ce nombre puis ses évolutions dans la mémoire d'un ordinateur, on utilise une variable.

3. Variables

Définition

Dans un programme, une variable est repérée par un nom et possède une valeur qui évolue au cours de l'exécution du programme. On peut la schématiser par une boîte (un emplacement de la mémoire d'un ordinateur) qui porte une étiquette et son contenu.
On utilise différents types de variables :

- nombre entier ;
- nombre flottant (nombre à virgule) ;
- chaîne de caractères ;
- liste ;
- booléen.

Exemple :

- *nombre entier : 2 ;*
- *nombre flottant : 4,333 ;*
- *chaîne de caractères : Bonjour ;*
- *liste : [1, 3, 5, 7, 9] ;*
- *booléen : Vrai (True) ou Faux (False).*

Remarques :

— un nombre flottant est un nombre décimal, pour les rationnels et réels une approximation sera faite ;
— une chaîne de caractères est une suite ordonnée de caractères qui peuvent être un chiffre, une lettre, un symbole, ... Dans l'exemple précédent, la chaîne de caractères est de longueur 7 ; le 4^{e} caractère est la lettre **j**. Les chaînes de caractères peuvent être concaténées c'est à dire les chaînes **Bonjour** et **Zoé** concaténées donnent **BonjourZoé**.
— une liste est une suite ordonnée d'objets du langage ;
— un booléen n'a que deux valeurs possibles : 1 ou 0, Vrai ou Faux. C'est souvent le résultat d'un test.

En programmation Python, les différents type de variables sont désignés par :

- **int** (integer) pour les nombres entiers ;
- **float** (floating-point) pour les nombres flottants ;
- **str** (string) pour les chaînes de caractères, elles sont délimitées par des apostrophes ou par des guillemets ;
- pour les listes, l'ensemble des éléments séparés par des virgules sont enfermés dans des crochets.

Des opérateurs et des fonctions du langage permettent de travailler avec chaque type de variables, mais pour cela il faut affecter (attribuer) une valeur à une variable.

4. L'affectation

Lorsque l'on affecte (donne) une valeur, par exemple 37, à une variable par exemple A :

. on écrit l'instruction : $A \leftarrow 37$.
. on lit : « A reçoit 37 » ou « A prend la valeur 37 » ou encore « la variable A est affectée de la valeur 37 ».

La nouvelle valeur remplace la valeur précédente.

Exemple : en reprenant l'exemple 1, la traduction de l'algorithme donnerait :

Choisir un nombre	. . .	$A \leftarrow 9$
Multiplier ce nombre par 5	. . .	$A \leftarrow A \times 5$
Ajouter 2 au résultat	. . .	$A \leftarrow A + 2$

En programmation Python, l'affectation est codée par le signe = ; l'algorithme précédent s'écrit :

sans commentaire

```
A = 9
A = A*5
A = A+2
```

avec des commentaires expliquant ce qui est fait

```
A = 9 # Choisir un nombre, ici 9
A = A*5 # Multiplier ce nombre par 5
A = A+2 # Ajouter 2 au résultat
```

Remarques :

— les commentaires ne sont pas pris en considération dans le programme, en langage Python ils sont précédés par le symbole #.

— on peut effectuer de multiples affectations simultanées ; soit la commande a, b = 4, 8.33 permet d'affecter la valeur 4 à la variable a et la valeur 8,33 à la variable b.
— si l'on souhaite connaître le résultat, il faudra demander à afficher le résultat.

5. Les instructions d'entrée-sortie

Les instructions d'entrée-sortie permettent de saisir en entrée et d'afficher en sortie les valeurs des variables.

Algorithme	en Python
Saisir A	A = float(input())
Afficher A	print(A)

Remarques :
— ces instructions permettent également d'afficher un message :
  - n = int(input("Nombre d'essais = ")) ;
  - print("La surface obtenue est : ",S).
— en Python, l'instruction d'entrée précise le type de la variable : int pour les nombres entiers, float pour les nombres à virgules et si rien n'est précisé la variable sera considérée comme une chaîne de caractères.

Exemple : on désire déterminer le volume d'un pavé à partir de sa longueur, sa largeur et sa hauteur.

Algorithme

Saisir L, l, h
$V \leftarrow L \times l \times h$
Afficher V

en Python

```
L = float(input("Longueur ="))
l = float(input("Largeur ="))
h = float(input("Hauteur ="))
V = L*l*h
print("Le volume du pavé est ",V)
```

6. Les instructions conditionnelles

Définition

Dans un algorithme, on est parfois amené à exécuter une ou plusieurs instructions uniquement si une certaine condition est vérifiée : ce sont des instructions conditionnelles. Si la condition n'est pas vérifiée, on peut exécuter un autre bloc d'instruction ou ne rien faire. Dans tous les cas, ensuite, on exécute la suite de l'algorithme.

a) Condition

Une condition est un énoncé qui peut être vrai ou faux ; en conséquence, le résultat d'une condition est de type booléen.
En Python, le signe = est déjà utilisé pour l'affectation ; le test d'égalité entre deux valeurs s'écriera == .
Par exemple pour vérifier si a est égale à 1, on saisira dans le programme **if a == 1**.

Exemple : l'algorithme suivant permet de saisir un nombre entier puis de déterminer si ce nombre est multiplie ou non de 5.

Algorithme

Saisir n
si *n est divisible par 5* **alors**
  $R \leftarrow$ "n est divisible par 5"
fin
si *n n'est pas divisible par 5* **alors**
  $R \leftarrow$ "n n'est pas divisible par 5"
fin
Afficher R

en Python

```
n = int(input("n ="))
if n%5 == 0 :
        R = " est divisible par 5"
if n%5 != 0 :
        R = " n'est pas divisible par 5"
print(n,R)
```

On peut optimiser l'algorithme et le programme en remplaçant les deux instructions conditionnelles par une seule complétée du sinon.

Algorithme

Saisir n
si *n est divisible par 5* **alors**
  $R \leftarrow$ "n est divisible par 5"
sinon
  $R \leftarrow$ "n n'est pas divisible par 5"
fin
Afficher R

en Python

```
n = int(input("n ="))
if n%5 ==0 :
        R = " est divisible par 5"
else :
        R = " n'est pas divisible par 5"
print(n,R)
```

Les langages de programmation permettent de définir des fonctions.

7. Notion de fonction

Définition

Une fonction réalise un bloc d'instructions et renvoie un résultat ; elle ne sera exécutée que si elle est appelée dans le programme et ceux-ci éventuellement plusieurs fois. Elle possède généralement des paramètres qui prendront pour valeur les arguments donnés à la fonction.

L'intérêt des fonctions est de faciliter l'écriture des programmes notamment quand un bloc d'instructions est souvent répété mais aussi de structurer et de rendre plus lisible les programmes.

Exemple : si l'on souhaite définir une fonction VolPave calculant le volume d'un pavé, en langage Python on écrira :

en Python

```
def VolPave(L,l,h):
        V = L*l*h
        return V
```

Exemple : pour s'en servir, il suffira d'écrire par exemple VolPave(7,3,5) pour calculer le volume d'un pavé de longueur 7, de largeur 3 et de hauteur 5 et obtenir 105 comme résultat.
Dans cet exemple les paramètres sont L, l et h ; les arguments sont 7, 3 et 5.

8. Caisse à outils

Déterminer le type d'une variable $\Longrightarrow$ on identifie les variables qui ne prennent pas de valeur numériques, car elles sont de type chaîne de caractères. Pour les autres on identifie si ces valeurs sont entières ou non , dans ce cas elles seront de type flottant.

Application : dans un établissement scolaire le nom des classes est une couleur donnée en fonction du niveau. On souhaite écrire un programme prenant en compte le nom de classes, le nombre d'élèves les constituant et la moyenne trimestrielle.
Déterminer le type de chacune de ces variables.

Déterminer les valeurs prises par les variables d'un algorithme $\Longrightarrow$ on construit un tableau avec pour en-tête de colonnes le nom des variables utilisées dans le programme. La première ligne correspond à l'initialisation des variables, c'est à dire la première valeur affectée à chacune d'entre-t-elle. Puis chaque nouvelle affectation est inscrite dans la ligne suivante.

Application : à partir de l'algorithme ci-contre ; quelles sont les valeurs des variables a, b et c en fin d'exécution ?

$a \leftarrow 3$
$b \leftarrow 5$
$a \leftarrow a * 2 - 5$
$c \leftarrow a + 3 * b$
$b \leftarrow 2 * a - b + c - 1$

Comprendre et écrire une instruction conditionnelle $\Longrightarrow$ on identifie la condition à tester et les instructions à exécuter si elle est vérifiée. On identifie les instructions à exécuter sinon, c'est à dire quand la condition n'est pas vérifiée.

Application : sur un site marchand, à moins de 90 € d'achats les frais de livraison s'élèvent à 8,50 €. Sinon ils sont offerts. On souhaite écrire un algorithme que l'on programmera ensuite, qui permettent de calculer le montant à facturer.

Écrire et utiliser une fonction simple $\Longrightarrow$ on identifie les paramètres de la fonction puis la valeur retournée et comment celle-ci est obtenue. Ensuite on écrit la fonction que l'on appellera dans le programme.

Application : on souhaite définir une fonction qui détermine le volume d'un cône ; pour rappel $V_{cone} = \frac{1}{3} \times B \times h$ *où* B *est la surface du disque et* h *la hauteur du cône.*
Tester la fonction avec les valeurs $h = 6$ *et* $R = 2$.

9. Vers d'autres horizons

Python Numworks	Maths et tiques Y. Monka	Capytale L. Fonquergne	Python Sabatier L. Fonquergne

10. Évaluations

Devoir en temps libre n° 3 : Variables et instructions élémentaires

Il est rappelé que la qualité de la rédaction, la clarté et la précision des raisonnements entreront pour une part importante dans l'apprécitaion des copies. Le barème est donné à titre indicatif. Le sujet sera rendu avec la copie.

Exercice n°1 : ***Aux bons conducteurs . . .***

Dans le tableau ci-dessous, on présente les consommations moyennes annuelles de conducteurs d'une société.

Consommations	Amandine	Boris	Charlotte	Denis	Elisée
Année $n-1$	4,9	4,4	3,8	4,1	4,2
Année n	4,3	4,2	3,9	3,9	4,1

Le responsable de la société décide d'octroyer une prime aux conducteurs ayant une consommation inférieure à 4 L/100 km ou ayant diminuer au moins de 5 % leur consommation.
À minima, il est attendu une feuille de calculs sous tableur déterminant quel employé a droit à la prime avec une mise en forme conditionnelle. La version experte serait l'écriture de fonctions renvoyant si oui ou non l'employé à droit à la prime. Pensez à présenter l'algorithme programmé.