



Variables informatiques

1. Algorithme et programmation

Définition 1.1

Un **algorithme** est une suite finie d'instructions à appliquer dans un ordre déterminé à un nombre fini de données pour arriver, en un nombre fini d'étapes, à un certain résultat.

Un algorithme peut être écrit dans un langage compréhensible par un humain : on parle de **langage naturel**.

Exemple 1.1 — On donne l'algorithme suivant :

Choisir un nombre X
Multiplier ce nombre par 2
Ajouter 3 au résultat

Quel est le nombre obtenu en sortie si le nombre choisi au départ est 7 ?

→ À rédiger

2. Notion de variable

Définition 1.2

Une **variable** est un symbole qui est associé à une valeur. Les variables sont utilisées pour stocker des valeurs.

Une variable est repérée par un nom ou une lettre. Dans un ordinateur ou une calculatrice, elle correspond à un emplacement de la mémoire.

Exemple 1.2 — Dans l'algorithme suivant, quelles sont les variables ?

Choisir un nombre X
Ajouter 3 à X
Choisir un nombre Y
Multiplier X et Y

→ À rédiger

3. Affectation

Définition 1.3

Le fait de donner une valeur à une variable s'appelle une **instruction d'affectation**.
En langage naturel, on écrit cela : « X prend la valeur ... » ou « X reçoit ... » ou « $X \leftarrow \dots$ ».

Exemple 1.3 — On donne l'algorithme suivant :

$X \leftarrow 4$
 $X \leftarrow X + 2$
 $X \leftarrow X * 5$

Quelle est la valeur de la variable X à la fin de cet algorithme ?

→ À rédiger

Exemple 1.4 — On choisit un nombre. On le multiplie par 2 puis on ajoute 7. On multiplie le résultat par 3 et on soustrait 6. Ecrire une suite d'instructions qui décrivent ces étapes lorsque le nombre choisi au départ est 3. → À rédiger

Exemple 1.5 — Dans un repère orthonormé, on donne $A(-2; 4)$ et $B(3; 5)$. Ecrire un algorithme qui permette de calculer la distance AB . On utilisera les noms de variables X_A , Y_A , X_B , Y_B et d . → À rédiger

4. Types de variables

Définition 1.4

On dit qu'une variable est :

de type **entier** si la valeur qu'elle contient est un nombre entier

de type **flottant** si la valeur qu'elle contient est un nombre à virgule

de type **chaîne de caractères** si la valeur qu'elle contient est une suite de caractères

de type **booléen** si la valeur qu'elle contient est « Vrai » ou « Faux »

Exemple 1.6 — Donner le type de chacune des variables des algorithmes suivants :

- | | |
|---|--|
| 1. $X \leftarrow 3$
$Y \leftarrow 5,2$
$Y \leftarrow X + Y$ | 2. $Z \leftarrow \text{"positif"}$
$S \leftarrow 2,5$
$T \leftarrow (S < 1)$ |
|---|--|

→ À rédiger

5. Programmer en Python

Définition 1.5

Programmer un algorithme, c'est le traduire dans un langage compréhensible par un ordinateur. Nous utiliserons le langage de programmation Python.

Correspondance entre le langage naturel et Python :

Langage naturel	Python
$X \leftarrow 2$ Afficher X Saisir un nombre X de type entier Saisir un nombre X de type flottant Saisir un nombre X de type chaîne de caractères	$X = 2$ <code>print(X)</code> <code>X = int(input())</code> <code>X = float(input())</code> <code>X = str(input())</code>

Exemple 1.7 — Traduire l'algorithme suivant en Python. A quoi sert cet algorithme ?

→ À rédiger

Afficher "Choisir un nombre"
Saisir un nombre X de type flottant
 $X \leftarrow X * X$
Afficher X

Voici quelques opérations possibles en Python :

Langage naturel	Python
Ajouter a et b Soustraire b de a Multiplier a par b Diviser a par b Calculer le quotient entier de a par a Calculer le reste de la division de a par b Elever a à la puissance n Calculer 4×10^3	<code>a + b</code> <code>a - b</code> <code>a * b</code> <code>a / b</code> <code>a // b</code> <code>a % b</code> <code>a ** n</code> <code>4e3</code>

Pour utiliser d'autres opérations mathématiques, il faut importer le module `math`. Il s'agit d'une bibliothèque contenant des fonctions supplémentaires comme la fonction racine carrée `sqrt`.

Python
`import math`
`a = math.sqrt(2)`
`print(a)`

Exemple 1.8 — Que va afficher le programme suivant :

→ À rédiger

Python
`import math`
`a = 64`
`b = math.sqrt(a)`
`c = a % 10`
`d = a // 5`
`print(b, c, d)`

1. Qu'est-ce qu'une instruction conditionnelle ?

Définition II.1

Une **instruction conditionnelle** est un type d'instruction qui permet d'exécuter une instruction ou une autre en fonction d'une certaine condition.

Une instruction conditionnelle se présente sous la forme suivante :

```
Si Condition alors
| Instruction 1
Sinon
| Instruction 2
```

Exemple II.1 — On donne l'algorithme suivant :

```
Choisir un nombre X
Si  $X^2 > 20$  alors
|  $X \leftarrow X + 3$ 
Sinon
|  $X \leftarrow X - 1$ 
Afficher X
```

Quel est le nombre affiché si le nombre de départ est :

a) 4? b) 7?

→ À rédiger

2. L'instruction conditionnelle en Python

En Python, une instruction conditionnelle commence par le mot-clef « if », suivi d'une condition et se termine par le symbole « : ». Si la condition est vérifiée, le bloc d'instructions à exécuter est écrit à la ligne avec un décalage que l'on appelle indentation. L'instruction alternative est précédée de « else : ».

```
Python
if condition:
    Instruction 1
else:
    Instruction 2
```

Exemple II.2 — On donne un algorithme en Python et son équivalent en langage naturel :

Python

```
X = 3
if X > 5 :
    X = X + 10
else:
    X = X * 2
```

Langage naturel

```
X ← .....
Si ..... alors
| .....
Sinon
| .....
```

1. Compléter l'algorithme en langage naturel.
2. Quelle valeur la variable X contiendra-t-elle à la fin de l'algorithme?

→ À rédiger

3. Tester une condition en Python

Voici quelques façons de tester si une condition est vraie ou fausse en Python :

Langage naturel

```
Si X = 3 alors
Si X ≤ 3 alors
Si X = 1 ou X > 5 alors
Si X ≥ 10 et X < 30 alors
Si X est différent de 5 alors
Si X est un multiple de 3 alors
```

Python

```
if X == 3 :
if X <= 3 :
if X == 1 or X > 5 :
if X >= 10 and X < 30 :
if X != 5 :
if X%3 == 0 :
```

Exemple II.3 — Écrire un programme en Python qui vous demande de saisir votre prénom. Si votre prénom est « Toto » alors le programme affichera « Bonjour Toto ! » ; sinon il affichera « Bonjour cher inconnu ! ».

→ À rédiger

Exemple I.1

17

Exemple I.2

Les variables sont X et Y.

Exemple I.3

30

Exemple I.4

Voici l'algorithme :

```
X ← 3
X ← 2 * X
X ← X + 7
X ← 3 * X
X ← X - 6
```

Exemple I.5

Voici l'algorithme :

```
XA ← -2
YA ← 4
XB ← 3
YA ← 5
D ←  $\sqrt{(X_B - X_A)^2 + (Y_B - Y_A)^2}$ 
Afficher D
```

Exemple I.6

1. X est de type entier, Y est de type flottant.
2. Z est de type chaîne de caractère, S est de type flottant et T est de type booléen.

Exemple I.7

Voici le programme Python :

```
Python
X = float(("Choisir un nombre"))
X = X * X
print(X)
```

Exemple I.8

Ce programme va afficher les valeurs 8, 0, 4 et 12.

Exemple II.1

- a) 3
- b) 10

Exemple II.2

Voici l'algorithme :

```
X ← 3
Si X > 5 alors
| X ← X + 10
Sinon
| X ← X * 2
```

Exemple II.3

Voici le programme Python :

```
Python
prenom = str(input("Quel est votre prénom ? "))
)
if prenom == "Toto":
    print("Bonjour Toto !")
else:
    print("Bonjour cher inconnu !")
```

Algorithmique (1ère partie)

A savoir faire à la fin du chapitre.

- Savoir déterminer le type d'une variable.
- Savoir écrire une instruction d'affectation ou une séquence d'instructions.
- Savoir écrire une formule permettant un calcul combinant des variables.
- Savoir écrire une instruction conditionnelle (« Si »).
- Connaître les bases du langage Python.

Algorithmique (1ère partie)

A savoir faire à la fin du chapitre.

- Savoir déterminer le type d'une variable.
- Savoir écrire une instruction d'affectation ou une séquence d'instructions.
- Savoir écrire une formule permettant un calcul combinant des variables.
- Savoir écrire une instruction conditionnelle (« Si »).
- Connaître les bases du langage Python.

Algorithmique (1ère partie)

A savoir faire à la fin du chapitre.

- Savoir déterminer le type d'une variable.
- Savoir écrire une instruction d'affectation ou une séquence d'instructions.
- Savoir écrire une formule permettant un calcul combinant des variables.
- Savoir écrire une instruction conditionnelle (« Si »).
- Connaître les bases du langage Python.