Rappels d'algorithmique – Partie 1

Variable, affectation

Définitions :

— Lors de l'exécution d'un algorithme, on va avoir besoin de stocker des données, voire des résultats. Pour cela, on utilise des variables.
On attribue un nom à chaque variable.

— Affecter une valeur à une variable, c'est remplacer le contenu de la variable par cette valeur (qui peut être le résultat d'un calcul).

Remarques :

1. Une variable est comme une boîte, repérée par un nom, qui va contenir une information. Pour utiliser le contenu de cette boîte, il suffit de l'appeler par son nom.
2. Dans l'écriture d'un algorithme, l'affectation d'une variable est désigné par le symbole $\longleftarrow$.
En Python, l'affectation d'une variable se fait grâce au signe $=$.
3. En Python, l'instruction `print(<nom de variable>)` permet d'afficher à l'écran la valeur d'une variable.
4. Il est possible de demander à l'utilisateur de l'algorithme de choisir la valeur d'une variable. On dit que cette variable est saisie. L'instruction Python permettant la saisie est `input()`.
Elle s'utilise de la façon suivante :

$$\underbrace{\texttt{a}}_{\text{nom de la variable}} = \texttt{input}\left(\underbrace{\texttt{"entrer la valeur de a : "}}_{\text{texte à afficher à l'écran}}\right)$$

Différents types de variables

Définitions : Dans un algorithme ou un programme Python, les variables ont un type qui définit la nature des valeurs que cette variable peut contenir.
Les trois principaux types de variables sont :

— Les entiers, lorsque les valeurs prises par la variable ne sont que des nombres entiers (éventuellement relatifs) ;

— Les flottants, lorsque la valeurs prises par la variable sont des nombres réels ;

— Les chaînes de caractères, lorsque les valeurs prises par la variable sont des mots ou des phrases.

Remarques :

1. En Python :
   — le type entier est noté `int` (pour *integer*) ;
   — le type flottant est noté `float` (pour *floating-point*) ;
   — le type chaîne de caractères est noté `str` (pour *string*).
2. Les commandes `int()`, `float()`, `str()` permettent de changer le type d'une variable. Ceci peut être utile lors de la saisie d'une variable, car l'instruction `input` renvoie une variable de type *chaîne de caractères*. La commande `a=float(input("saisir une valeur : "))` permet de s'assurer que la variable a sera bien un flottant.

Fonctions

Définition : Pour diverses raisons (de lisibilité ou pour éviter la répétitions d'instructions par exemple), il peut être utile d'écrire un bloc d'instruction sous la forme d'une fonction. Ce bloc d'instruction ne sera exécuté que s'il est appelé par la suite, en utilisant le nom de la fonction.
Une fonction possède généralement des paramètres, et renvoie généralement une valeur de retour.
Dans un algorithme, cette structure est codée de la façon suivante :

```
Fonction nom(param1,param2,...)
    tâche 1
    tâche 2
    ...
    Retourner variable
```

Remarque : En Python, une fonction se code de la façon suivante :

```
def nom(param1,param2,...) :
    tâche 1
    tâche 2
    ...
    return variable
```

Exemples :

L'algorithme 1 permet de calculer l'aire d'un triangle, en utilisant une fonction.
Sa traduction en Python est donné dans l'algorithme 2.

Algorithme 1 Aire d'un triangle

```
Fonction aire(base,hauteur)
    a ⟵ (base×hauteur)/2
    Retourner a
b ⟵valeur saisie
h ⟵valeur saisie
A ⟵ aire(b,h)
Afficher A
```

Algorithme 2 Programme Python : aire d'un triangle

```
def aire(base,hauteur) :
    a = (base×hauteur)/2
    return a
b = float(input("Entrer la longueur de la base : ")
h = float(input("Entrer la longueur de la hauteur : ")
A = aire(b,h)
print("L'aire du triangle est ",A)
```

L'instruction conditionnelle

Définition : La résolution des certains problèmes nécessite la mise en place d'un test pour savoir si l'on doit effectuer une tâche.
Si la condition est remplie alors on effectue la (ou les) tâches, sinon on effectue (éventuellement) une autre (ou des autres) tâches.
Dans un algorithme, on code la structure du « Si... Alors.. Sinon » sous la forme suivante :

```
Si condition Alors
    Tâche 1
    Tâche 2
    ...
Sinon
    Tâche 1bis
    Tâche 2bis
    ...
Fin Si
```

Remarques : 1. Il est important de respecter les espaces laissés au début de chaque ligne, appelés indentations, car ils permettent de savoir quel bloc d'instructions fait partie du test..

2. Le « Sinon » n'est pas obligatoire. S'il n'est pas présent, aucune tâche ne sera effectué si la condition n'est pas remplie.

3. En Python, une instruction conditionnelle se code de la façon suivante :

```
if condition :
    Tâche 1
    Tâche 2
    ...
else :
    Tâche 1bis
    Tâche 2bis
    ...
```

L'indentation en début de ligne est obtenue grâce à la touche *Tabulation* du clavier. Il ne faut pas oublier les : après la condition du *if* et après le *else*.

Exemples :

L'algorithme 3 et le programme Python associé (algorithme 4) permet de déterminer si un triangle ABC est rectangle en C.

Algorithme 3 Triangle rectangle en C

```
AB ⟵ valeur saisie
AC ⟵ valeur saisie
BC ⟵ valeur saisie
x ⟵ AB²
y ⟵ AC²+BC²
Si x = y Alors
    Afficher « Le triangle ABC est rectangle en C »
Sinon
    Afficher « Le triangle ABC n'est pas rectangle en C »
Fin Si
```

Algorithme 4 Programme Python :Triangle rectangle en C

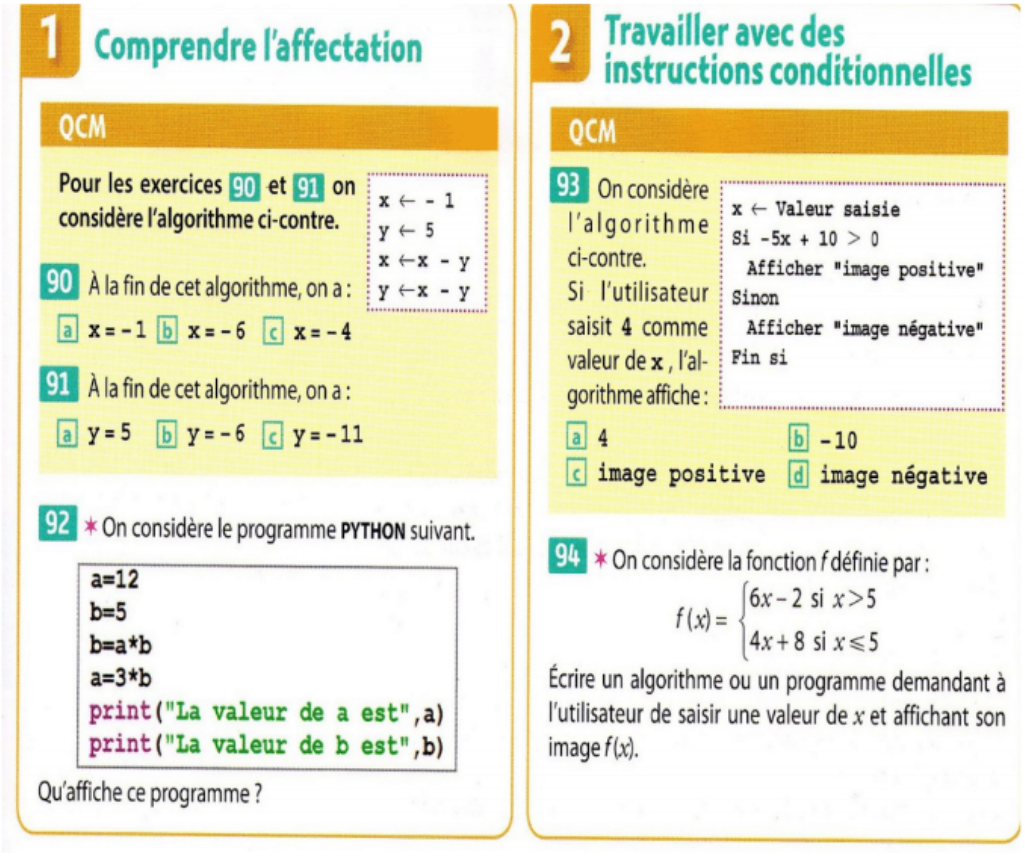
```
from math import *
AB = float(input("Entrer la valeur de AB : "))
AC = float(input("Entrer la valeur de AC : "))
BC = float(input("Entrer la valeur de BC : "))
x = pow(AB,2)
y = pow(AC,2)+pow(BC,2)
if x == y :
    print("Le triangle ABC est rectangle en C")
else :
    print("Le triangle ABC n'est pas rectangle en C")
```

Remarques :

- L'instruction `from math import *` permet de charger la bibliothèque mathématique de Python, qui contient plus de fonctions mathématiques, notamment la mise au carré : `pow(AB,2)` qui signifie AB^2.
- En Python, le test d'égalité se fait en utilisant ==. Le signe = est réservé aux affectations de variables.
- Lors d'un test, on peut aussi utiliser `>` , < , `>=` , `<=` ou bien `!=` (qui signifie n'est pas égal à).

Exercices

Exercice 1



1 Comprendre l'affectation

QCM

Pour les exercices 90 et 91 on considère l'algorithme ci-contre.

```
x ← - 1
y ← 5
x ←x - y
y ←x - y
```

90 À la fin de cet algorithme, on a :

a `x = -1` b `x = -6` c `x = -4`

91 À la fin de cet algorithme, on a :

a `y = 5` b `y = -6` c `y = -11`

92 * On considère le programme **PYTHON** suivant.

```
a=12
b=5
b=a*b
a=3*b
print("La valeur de a est",a)
print("La valeur de b est",b)
```

Qu'affiche ce programme ?

2 Travailler avec des instructions conditionnelles

QCM

93 On considère l'algorithme ci-contre. Si l'utilisateur saisit 4 comme valeur de `x`, l'algorithme affiche :

```
x ← Valeur saisie
Si -5x + 10 > 0
  Afficher "image positive"
Sinon
  Afficher "image négative"
Fin si
```

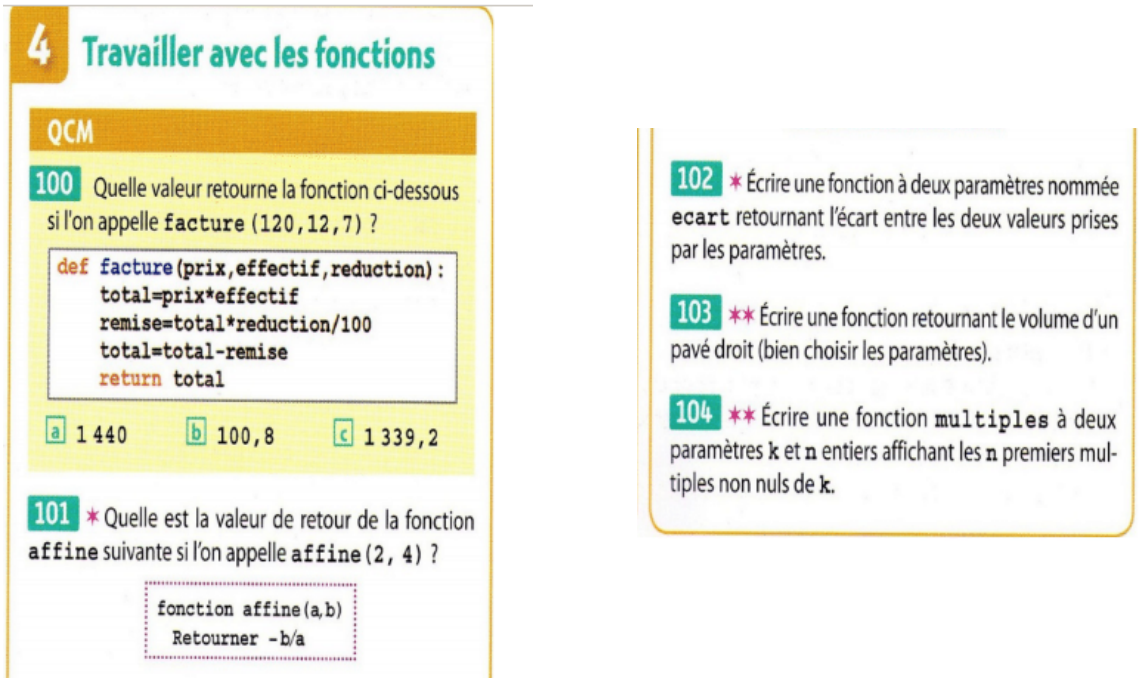
a `4` b `-10`
c `image positive` d `image négative`

94 * On considère la fonction f définie par :

$$f(x) = \begin{cases} 6x-2 \text{ si } x>5 \\ 4x+8 \text{ si } x\leqslant 5 \end{cases}$$

Écrire un algorithme ou un programme demandant à l'utilisateur de saisir une valeur de x et affichant son image $f(x)$.

Exercice 2



4 Travailler avec les fonctions

QCM

100 Quelle valeur retourne la fonction ci-dessous si l'on appelle `facture(120,12,7)` ?

```
def facture(prix,effectif,reduction):
    total=prix*effectif
    remise=total*reduction/100
    total=total-remise
    return total
```

a `1 440` b `100,8` c `1 339,2`

101 * Quelle est la valeur de retour de la fonction `affine` suivante si l'on appelle `affine(2, 4)` ?

```
fonction affine(a,b)
  Retourner -b/a
```

102 * Écrire une fonction à deux paramètres nommée `ecart` retournant l'écart entre les deux valeurs prises par les paramètres.

103 ** Écrire une fonction retournant le volume d'un pavé droit (bien choisir les paramètres).

104 ** Écrire une fonction `multiples` à deux paramètres `k` et `n` entiers affichant les `n` premiers multiples non nuls de `k`.

Exercice 3

En vous inspirant des exemples précédents, écrire une fonction PYTHON appelée `hypotenuse(a,b)` qui renvoie la longueur de l'hypoténuse d'un triangle rectangle dont les longueurs des côtés de l'angle droit sont `a` et `b`. Tester ensuite cette fonction en vérifiant les résultats à la main.