Rappels de seconde sur les algorithmes et le langage Python

I - Affectations et variables

Définition

> Un algorithme est une suite finie d'opérations élémentaires, à appliquer dans un ordre déterminé, à des données.

Dans un algorithme ou un programm en Python :

- on utilise des **variables** ;
- les variables représente des nombres ou d'autres objets (listes, chaînes de caractère, ...) ;
- on utilise une lettre ou un mot pour les désigner ;
- on modifie leur valeur lors d'**affectations**.

On effectue par exemple les affectations suivantes :

$$
\begin{aligned}
&b \leftarrow 3\\
&a \leftarrow 4\\
&b \leftarrow a+b
\end{aligned}
$$

La traduction en Python est la suivante :

```
b = 3
a = 4
b = a + b
```

II - Instructions conditionnelles

Définition

> La structure `si ... alors ... sinon ...` (qui se traduit par `if ... then ... else` en anglais) permet de définir une condition: *si* cette condition est remplie, **alors** on effectuera certaines instructions ; **sinon** on effectuera d'autres instructions.

La structure générale est la suivante :

Si Condition **alors**

> Traitement 1

sinon

> Traitement 2

La condition est soit *vraie* soit *fausse*, si elle est vraie le Traitement 1 est effectué, si elle est fausse c'est le Traitement 2 qui est effectué. L'intruction `sinon` n'est pas obligatoire.

Sa traduction en Python est la suivante :

```
if Condition:
    Traitement1
else:
    Traitement2
```

III - Les boucles

Définition

> Une **boucle bornée** est utilisée lorsque l'on veut **répéter un certain nombre de fois** les mêmes instructions.

Remarque

> Une variable sera utilisée pour **compter** le nombre de répétitions, on dit parle ainsi **d'itérations**. En général, elle prendra des valeurs entières. À chaque itération, la variable est incrémentée d'une unité (sauf indication contraire).

La structure générale est la suivante :

Pour i **allant de** 0 **jusqu'à** 9 **faire**

> Instructions

FinPour

Dans cette boucle, les *instructions* sont répétées 10 fois.

Remarque

> La variable i prendra successivement les valeurs : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9.

Sa traduction en Python est la suivante :

```
for i in range(10):
    Instructions
```

Définition

> Une **boucle non bornée** est utilisée lorsque l'on veut **répéter** les mêmes instructions **tant que** une **condition** est réalisée.

La structure générale est la suivante :

Tant que Condition

> Instructions

FinTantque

Remarque

> Lorsque l'on sort de la boucle la "*Condition*" est fausse.

Sa traduction en Python est la suivante :

```
while Condition:
    Instructions
```

IV - Les fonctions

Information

> Les fonctions permettent de décomposer un algorithme complexe en une série de sous-algorithmes plus simples, lesquels peuvent à leur tour être décomposés en fragments plus petits, et ainsi de suite. Une fonction **retourne** en **sortie** un ou des objet(s) et/ou exécute une tâche, pour cela on peut lui fournir un ou des **arguments** en **entrée**.
> Une fonction est ensuite **appelée** dans le cœur de l'algorithme.

La structure générale est la suivante :

Définition Fonction(paramètres):

> Instructions
> Retourne objets

FinDéfinition

...
`# Appel de la fonction`
A ← Fonction(valeurs des paramètres)

Sa syntaxe Python est la suivante :

```
def Fonction(paramètres):
    Instructions
    return objets

# Appel de la fonction
A = Fonction(valeurs)
```

Remarques :

- Il faut bien distinguer la définition de la fonction et son appel à l'intérieur de l'algorithme.
- Lors de cet appel, l'objet retourné par la fonction est affecté à la variable A.