Séquence 2

Algorithmique et Programmation

I. Les types de variables en Python

A. Qu'est-ce qu'un algorithme :

Il s'agit d'une suite finie d'instructions décrite sans ambiguïté qui permet de résoudre un problème donné. Avec un ordinateur, on utilise un langage de programmation pour faire comprendre cet algorithme à l'ordinateur et lui dire ce qu'il doit faire.

B. Qu'est-ce que Python ?

C'est un langage simple et efficace inventé dans les années 90. C'est un langage interprété qui a donc besoin d'un logiciel pour s'exécuter.

C. Les types de variables en Python :

Une variable est désignée par un nom et contient une valeur. On utilise 4 types de variables :

- Le type **int** désigne les nombres entiers relatifs : 5, -12…
- Le type **float** représente les nombres réels, les fractions, le nombre π…
- Le type **string** désigne les chaines de caractères : mot, texte, phrase entre guillemets.
- Le type **booleen** : variable qui ne peut prendre que deux états : Vrai ou faux.

Remarques Un nom de variables respecte des règles **:**

- Il peut avoir plusieurs caractères ;
- Il peut contenir des chiffres ;
- Il ne peut pas commencer par un chiffre ;
- Il ne peut pas contenir d'espace ;
- Il ne doit pas contenir de lettre accentuée.

- Instruction sur NUMWORKS

Pour connaitre le type de variable, on saisit type (nom de la variable)

Exemple : On veut connaitre le type de la valeur 17,564.

Programme Python : b = 17.564

Type(b)

<class 'float'>

Ainsi Python a répondu que c'est un float.

D. Affectation et comparaison en Python :

1. Affectation d'une variable :

Dans un algorithme, l'instruction « a prend la valeur… » s'écrit a ← 17.

En Python, le symbole de l'affectation est =.

Exemple :

Programme a=17

b=3.7

2. Symbole de la comparaison :

Il est utilisé pour savoir si des valeurs ou variables sont égales ou non. Le symbole est ==

La réponse est un booleen.

Programme Python : var=17

b=16

var==b

False

3. Symbole ≠ :

Il est utilisé pour savoir si deux valeurs ou variables sont différentes.

La réponse est un booléen. Le symbole est !=

Programme :

5 !=6

True

Symbole de la comparaison :

Le signe < désigne « plus petit », > désigne « plus grand »,>= désigne « plus grand ou égal » et <= désigne « plus petit ou égale ».

Programme :

a=32

a<50

true

a<10

false

4. Les instructions de calcul :
   - Les symboles +,-,×,/ s'écrivent en python : +,-,*,/.
   - x a la puissance n s'écrit x**n.
   - Le reste de la division euclidienne de a par b s'écrit a%b, et le quotient entier de cette division s'écrit a//b.
   - La racine carrée de x s'écrit sqrt(x) et le nombre pi s'écrit π

II. Les fonctions

La notion de fonction en Python est assez proche de la notion de fonctions en mathématiques.

A. Exemple de fonctions

Clara fabrique des crêpes qu'elle vend 1,5 euros l'unité. On souhaite créer une fonction ayant pour paramètres d'entée x le nombre de crêpes vendues et qui renvoie le montant total de ses ventes.

En langage Python, on définit une fonction par la commande def suivi du nom de la fonction, puis son argument e, entre parenthèse, suivi de deux points.

On renvoie le résultat par la commande return.

Programme Python :

```
def vente(x) :
V=1.5*x
Return v
```

Lors de l'exécution du programme, si l'on veut obtenir le chiffre d'affaire pour la vente de 148 crêpes, on écrit vente(148)

Propriétés :

- Une fonction ne renvoie qu'un seul résultat.
- Une fonction peut n'avoir aucun argument.

Exemple :

```
def imp() :
return (« impossible »)
```

- Une fonction peut être appelé dans un autre programme : il suffit pour cela de l'insérer dans une instruction en saisissant son nom et les valeurs des arguments. La valeur appelée peut été stocké dans une variable.

Exemple : livre p 20

III. Les instructions en Python

A. Les instructions de sortie en Python

Pour récupérer les résultats d'un programme, il existe deux possibilités.

- Return provoque un affichage dans la console. A privilégier quand on programme car correspond à la création d'une fonction.
- Print est une instruction de sortie pour afficher la valeur d'une variable, un texte…

B. Le test conditionnel « Si…Alors…Sinon »

Ce test permet d'exécuter des instructions sous certaines conditions.

Algorithme :

```
Si condition alors
    Instruction 1
Sinon
    Instruction 2
FinSi
```

Programme Python :

```
If {condition C} :
{instruction A}
Else :
{instruction B}
```

Exemple

C. La boucle Pour :

Une boucle permet de répéter un ensemble d'instructions. La boucle peut s'utiliser avec un compteur qui a une valeur de départ et une valeur de fin déterminée et qui s'incrémente à chaque tour de la boucle. On dit que c'est une boucle bornée.

Algorithme : Pour i allant de **1** à **n**

Instructions

FinPour

Programme en Python : for i in range (1, n+1) :

Instructions

Remarques : i in range (a,b) correspond aux valeurs entières de i telles que $a \leq i < b$.

Exemple :

D. La boucle Tant que :

La boucle TantQue s'exécute jusqu'à ce qu'une condition est réalisée et autorise la sortie. Elle peut donc se répéter un grand nombre de fois. Il faut donc mettre une condition de sortie.

Algorithme : TantQue condition

Instructions

Fin TantQue

Programme Python :

While condition :

Instructions

Exemple : Capacité 11 et 12 p 25