Python

01 - Bases de Python

A - Python

1 Qu'est ce que Python

Python est un langage de programmation puissant et facile à apprendre. Il dispose de structures de données de haut niveau et permet une approche simple mais efficace de la programmation orientée objet. Parce que sa syntaxe est élégante, que son typage est dynamique et qu'il est interprété, Python est un langage idéal pour l'écriture de scripts et le développement rapide d'applications dans de nombreux domaines et sur la plupart des plateformes.

2 Documentation

L'interpréteur Python et sa vaste bibliothèque standard sont disponibles librement, sous forme de sources ou de binaires, pour toutes les plateformes majeures depuis le site Internet

https://www.python.org/ et peuvent être librement redistribués. Ce même site distribue et pointe vers des modules, des programmes et des outils tiers. Enfin, il constitue une source de documentation.

B - Terme de Base

1 Notion de Variable

Définition du mot ordinateur d'après "Le Petit Larousse" :

"*Machine automatique de traitement de l'information, obéissant à des programmes formés par des suites d'opérations arithmétiques et logiques.*"

Qui dit "traitement de l'information", dit donc données à manipuler. Un programme "passe" donc son temps à traiter des données. Pour pouvoir traiter ces données, l'ordinateur doit les ranger dans sa mémoire (RAM - Random Access Memory). La RAM se compose de cases dans lesquelles nous

allons ranger ces données (une donnée dans une case). Chaque case a une adresse (ce qui permet au processeur de savoir où sont rangées les données).

Alors, qu'est-ce qu'une variable ?

Eh bien, c'est une petite information (une donnée) temporaire que l'on stocke dans une case de la RAM. On dit qu'elle est "variable", car c'est une valeur qui peut changer pendant le déroulement du programme.

Une variable est constituée de 2 choses :

- une valeur présente en mémoire (par exemple le nombre entier 5)
- un nom

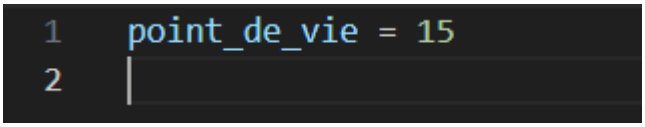
On dira donc qu'une variable est l'association d'un nom et d'une valeur

```
i = 12
```

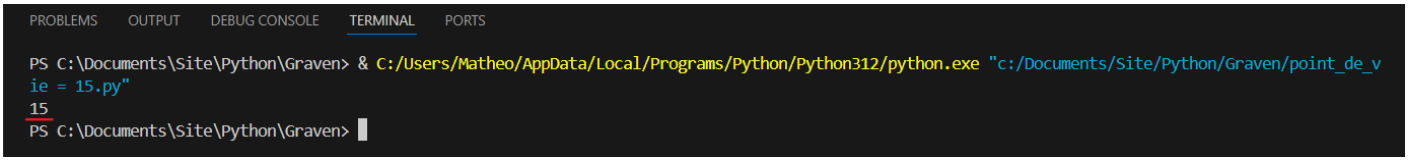
Grâce à cette ligne, nous avons défini une variable qui porte le nom i. Ce nom i est associé à la valeur 12.

Dans la partie "éditeur" de Visual Studio Code, saisissez le code suivant :

```
point_de_vie = 15
print (point_de_vie)
```



Après avoir exécuté le programme en cliquant sur le triangle vert, il est possible de connaitre la valeur associée à un nom en utilisant la partie "console" de Visual Studio Code.



Dans la suite la procédure sera toujours la même :

- Vous utiliserez la partie "éditeur" pour saisir votre programme
- vous utiliserez la partie "console" pour afficher la valeur associée à un nom

Les variables sont des espaces de stockages pour tout type d’élément comme des caractères ou des nombres, néanmoins on ne peut pas tout stocker. Le langage python

fait la différence entre les types d’éléments.

Ainsi on obtient des variables de type « int( ) » qui stocke des **nombres entiers** donc « SANS virgule ».

```
1   variable_int = 1
2
```

Nous avons les « float( ) » qui stocke des **nombres réels** donc, « avec ou sans virgules ».

```
1   variable_float = 1.111
2
```

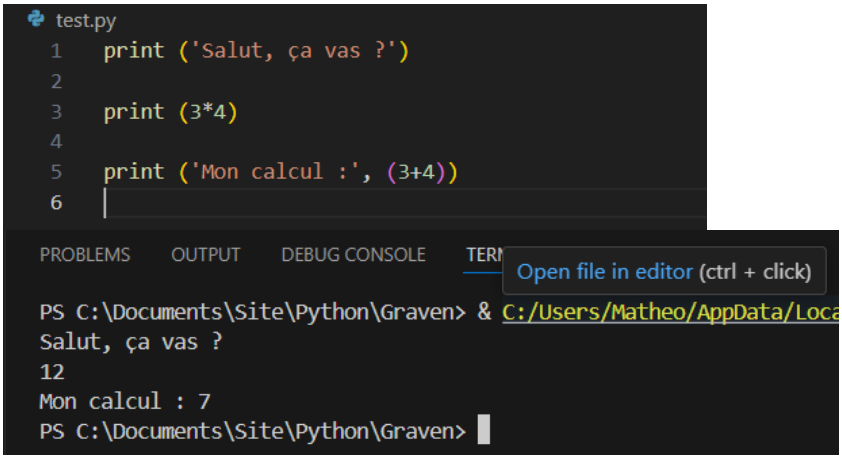
Et enfin les « strings » qui servent à **stocker les caractères**.

```
1   variable_string = 'Salut !'
2
```

C - Commande de Base

1 Print( )

La commande « print() » sert à afficher sur l’écran aussi bien des valeurs que des phrases. Il est obligatoirement précédé de parenthèses pour lui indiquer quoi écrire.



2 Input ( )

La commande input() en Python est utilisée pour demander à l'utilisateur d'entrer des données à partir du clavier pendant l'exécution du programme. Elle affiche généralement un message à l'utilisateur, attend que celui-ci entre une valeur depuis le clavier, puis retourne cette valeur sous

forme de chaîne de caractères.

Précédemment nous avons vu comment déclarer des variables, maintenant, nous allons voir comment récupérer une variable demandée à l'utilisateur.

```
variable_float=float(input("rentre une variable de type float"))

variable_string=str(input("rentre une variable de type float"))

variable_int=int(input("rentre une variable de type int"))

```

La commande « = » permet d'affecter une variable à une valeur.

Exemple :

```
test.py > ...
print(' ')
print(' ------ Bonjour ------ ')
print(' ')
a=float(input(" Veuillez entrer la valeur de a "))
d=2*a - 5
print(" La valeur de d vaut : ",d)
```

```
PS C:\Documents\Site\Python\Graven> & C:/Users/Matheo/AppD

 ------ Bonjour ------

 Veuillez entrer la valeur de a 3
 La valeur de d vaut :  1.0
PS C:\Documents\Site\Python\Graven>
```

Le programme demande à l'utilisateur la valeur de a, ici a = 3 , calcule d = 2a–3 et affiche la valeur de d.

3 La Fonction import ( )

Il est possible d'utiliser des fonctions déjà existantes dans Python, pour cela, il faut les importer !
Au début du programme, il faut rentrer l'instruction : « from math import * ». Elle permet dans le programme d'utiliser l'instruction « sqrt( ) » pour calculer la racine carrée mais aussi l'instruction « pi » pour π , « floor » pour partie entière, « cos » pour le cosinus d'un angle. On peut aussi utiliser « from random import * » qui permet d'utiliser l'instruction « random() » qui permet de générer un nombre aléatoire.

Exemple :

```python
import random
```

4 Les Fonctions if, then, else, elif

En langage Python, une condition IF (“si” en français) permet d'exécuter une ou plusieurs instructions spécifiques seulement si une condition est vérifiée.

A la suite de l'instruction « if », il faut écrire le test voulu et faire suivre cela par « : » Une fois validée la ligne, le programme va faire un décalage et celui marque les lignes d'instruction qui seront effectuées si la condition est vérifiée. « == » permet de vérifier si deux valeurs sont égales et «!= » permet de vérifier si elles sont différentes.

On peut rajouter une instruction si la condition n'est pas vérifiée avec « else : »

Enfin, l'instruction « elif » permet de rajouter une condition si la première n'est pas vérifiée.

Exemple :

```python
choix1 = 1
choix2 = 2

choix = int(input('Quel est votre choix en 1 et 2 ? '))
if choix == choix1:
    print ('Vous avez fait le choix 1 !')
elif choix == choix2:
    print ('Vous avez fait le choix 2 !')
else:
    print ('Votre choix est incorrect !')

```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Documents\Site\Python\Graven> & C:/Users/Matheo/Ap
Quel est votre choix en 1 et 2 ? 1
Vous avez fait le choix 1 !
PS C:\Documents\Site\Python\Graven>
```

Ici, on peut voir que si l'utilisateur sélectionne 1, le print du choix 1 sera exécuté et si il sélectionne 2, le print du choix 2 sera exécuté. Si le résultat entré ne correspond à aucune variable, le print "incorrect" s'exécutera.

5 La Fonctions while

La fonction « while » permet de faire les instructions qui suivent tant que l'instruction précédée de « while » n'est pas vérifié (boucle infini), par exemple, « while a!=b : » qui signifie tant

que a
est différent de b faire ...
Il ne faut pas oublier les « : » après l'instruction.

Exemple :

```python
import random

def jeu1():
    petit = int(input('Plus petit chiffre:'))
    grand = int(input('Plus grand chiffre'))
    chiffre = random.randint (petit, grand)
    print (chiffre)

    while True:
        line = int(input('Chiffre aléatoire: '))
        if line > chiffre:
            print ('Trop grand')
        elif line < chiffre:
            print ('Trop petit')
        elif line == chiffre:
            break
        print('Mauvais resultat')
    print ('Bon résultat !')

```

On peut voir une boucle infinie dans la partie rouge, au début de ce script un chiffre aléatoire est défini et tant que l'utilisateur n'a pas trouvé ce chiffre la boucle continue.