1 Introduction à Python

Introduction *Python* est un langage de programmation, créé en 1989 ar Guido Van Rossum et placé sous une licence libre. Les premières versions du logiciel sont publiées en 1995.
Par la suite, une version 2.0 en 2000, puis une version 3.0 en 2008 (que nous utiliserons) voient le jour.
Python est utilisé dans de nombreux domaines grâce à ses nombreux modules spécialisés.
Il est particulièrement utilisé dans le monde scientifique. Il existe en effet beaucoup de bibliothèques pour le calcul numérique et/ou formel.

I Environnement de développement

I.1 Programmation : console / IDE

Pour faciliter l'écriture de programmes, des environnements de développement (ou **I**ntegrated **D**evelopment **E**nvrironment) ont-été créés.
Ils permettent sur un même logiciel de pouvoir écrire un programme, le sauvegarder et l'exécuter, en plus de proposer des raccourcis spécialement pensés pour le langage en question.
Pour Python, un exemple de tel logiciel est *Spyder* (gratuit).
Dans toute la suite nous l'utiliserons pour écrire nos différents programmes.
À noter qu'il est possible d'utiliser directement Python dans une console. Cependant, lorsqu'on l'utilise ainsi, chaque ligne une fois validée ne peut plus être modifiée. De plus, les différentes commandes rentrées au fur et à mesure tombent dans l'oubli une fois que l'on ferme la console.
C'est pourquoi nous travaillerons exclusivement sur IDE.
Comment travaille-t-on sur un IDE ?
Nous écrirons des scripts (ou fichiers textes), que nous sauvegarderons et exécuterons afin de voir ce qu'ils effectuent.

I.2 Installation de *Spyder*

La page d'accueil de Spyder se trouve à l'adresse https://www.spyder-ide.org/.
Un onglet *DOWNLOAD* permet d'aller rapidement au lien de téléchargement pour installer le logiciel.

I.3 Premier script

Afin de tester l'installation :

1. lancer Spyder, puis cliquer "Fichier" → "Nouveau fichier".
2. Dans le fichier, écrire la ligne suivante : `print("Bienvenue sur Spyder")`
3. Sauvegarder le fichier à l'aide du raccourci "Ctrl + s".
4. Exécuter le script à l'aide du raccourci "F5".

Si l'installation est bien faite, alors vous devez voir apparaître dans la fenêtre "Console", le message écrit plus haut.

II Variables, opérateurs et affichage

II.1 Affichage

Définition 1.1 – print

La fonction _____ en Python, permet d'afficher du contenu lors de l'exécution d'un script.
Elle peut prendre plusieurs _____ , séparés d'une _____ .

Exemple 1.1 :
Qu'est-ce qui apparaît en sortie du script suivant ?

```
print("2+2 est égal à ",2+2)
```

Remarque(s) :

- Les guillemets servent à écrire un texte, et 2+2 sans guillemet est calculé par Python.

II.2 Les variables

Définition 1.2 – Variable

Une _____ est un symbole dans lequel il est possible de stocker une information.

Définition 1.3 – Type d'une variable

En Python, une variable peut avoir (entre autres) un des types suivants :

- nombre entier : ____ (pour _____)

- nombre décimal : _____
- chaîne de caractères : ____ (pour _____)
- booléen (True ou False) : _____ (pour _____)

Propriété 1.1 – Affectation

Pour _____ une valeur à une variable a, il suffit d'écrire _____ .
Le nom d'une variable est arbitraire, mais doit respecter certaines règles :

- le premier caractère ne doit pas être un chiffre.
- il ne doit pas y avoir d'espace.
- le nom de la variable ne doit pas être un mot clé du langage (e.g *print*).

NB : Pour connaître le type d'une variable a, on peut écrire `type(a)`.

Exemple 1.2 :

1. On considère le script suivant :

```
a = "Hello"
a = "World"
```

   (a) Quel est le type de a ?
   (b) Que renvoie `print(a)` ?

2. On considère le script suivant :

```
b = 1
b = b +1
```

   (a) Quel est le type de b ?
   (b) Que renvoie `print(b)` ?

3. On considère le script suivant :

```
c = 1.5
d = 3
e = c<d
f = c>d
```

   (a) Quel est le type de c ?
   (b) Quel est le type de e ?
   (c) Quel est le type de f ?
   (d) Que renvoie `print(e)` ?
   (e) Que renvoie `print(f)` ?

II.3 Opérateurs

a) Opérateurs de comparaison

Définition 1.4

Le tableau ci-dessous résume les différents opérateurs de comparaison :

Test	$<$	$>$	$\leq$	$\geq$	$=$	$\neq$
Code	<	>	<=	>=	==	!=

b) Opérateurs mathématiques

Définition 1.5

Le tableau ci-dessous résume les différents opérateurs mathématiques :

Opération	$+$	$-$	$\times$	puissance	$\div$	quotient	reste
Code	+	-	*	**	/	//	%

c) Opérateurs logiques

Définition 1.6

Le tableau ci-dessous résume les différents opérateurs logiques à utiliser entre booléens :

Logique	et	ou	négation
Code	and	or	not

Exemple 1.3 :

Voici un script :

```
a = 3
b = 2**4
c = 7//2
print((not a!=3) and a<5 and a*c<b)
```

1. Que contiennent les variables b et c ?
2. Qu'est ce qui apparaît en sortie lors de l'exécution de ce script ?

Exercices : A

III Intéraction avec l'utilisateur

On peut parfois avoir envie que la personne qui exécute un code doive saisir une information (e.g son prénom, son âge, etc).
Pour cela, on peut utiliser la fonction `input` de Python.
Pour comprendre son fonctionnement, voici un exemple :

```
nombre = input("À quel nombre pensez-vous ?")
print(nombre)
```

Lors de l'exécution du code ci-dessus, l'utilisateur va voir le message "À quel nombre pensez-vous ?" et devra entrer une réponse sur son clavier.
Le contenu de sa réponse sera stocké dans la variable `nombre`.
La deuxième ligne de code affichera ensuite le contenu de la variable `nombre`.

IV Modules Python

Dans le module de base, Python ne dispose pas des fonctions courantes utilisées en mathématiques. Il faut, par exemple pour pouvoir calculer le sinus d'un angle, importer le _____ (aussi appelé bibliothèque, ou encore librairie) `math`.

Les méthodes d'importation de modules, proposées ci-dessous pour le module `math` sont bien évidemment valables pour tous les modules.

Méthode Les trois méthodes ci-dessous sont illustrées dans une console afin de voir directement la sortie, mais on peut procéder exactement de la même manière dans un script sur Spyder.

1.
```
>>> import math
>>> a = math.sin(math.pi/4)
>>> print(a)
0.7071067811865475
```

2.
```
>>> import math as m
>>> a = m.sin(m.pi/4)
>>> print(a)
0.7071067811865475
```

3.
```
>>> from math import *
>>> a = sin(pi/4)
>>> print(a)
0.7071067811865475
```

Pour connaître puis tester les fonctions d'une librairie, on peut les lister à l'aide de la fonction `dir`.
Par exemple `print(dir(math))` permet de connaître toutes les fonctions de la librairie `math`.

Exercices : B