Bibliographie

1. www.python.org (Python Tutorial pour commencer)
2. https ://python.sdv.univ-paris-diderot.fr
3. http ://www.inforef.be/swi/python.htm
4. Learn Python 3 the Hard Way

1 TP 1

1.1 Premier programme

Exercice 1. Interpréteur Python
Ouvrir un shell (avec l'application Terminal) et lancer la commande
python
pour lancer l'interpréteur. Le triple chevron >>> est l'invite de commande.
Exécutez la commande
`print("Hello world!")`

Exercice 2. Premier programme
Créer un fichier `test.py` contenant
`print("Hello world!")`
Exécuter ce programme avec la commande
`python test.py`
dans le shell.

Exercice 3. Commentaire
Modifier `test.py` de la façon suivante :

```
print("Hello # world!") # Tout ce qui à droite de # est ignoré
# seulement les # en dehors d'une chaîne de caractères
```

Exercice 4. Aide
Dans le shell, exécutez pydoc print pour obtenir la documentation sur la fonction `print`.
Dans l'interpréteur Python, utilisez `help(print)`

1.2 Variable et types de base

Exercice 5. Booléens
Dans l'interpréteur Python, exécutez :

```
type(False)
x = (1<2)
type(x)
2 > 8
```

```
2 <= 8 < 15
(3 == 3) or (9 > 24)
(9 > 24) and (3 == 3)
```

Exercice 6. int, float, complex
Dans l'interpréteur Python, exécutez :

```
20 + 3
20 * 3
2 ** 100 # puissance
type(20)
type(20.0)
2 / 1 ; type(2 / 1)
x = 1 + 1j
type(x)
x**2
x.real
x.imag
abs(x)
```

Exercice 7. Chaîne de caractères (str)
Dans l'interpréteur Python, exécutez :

```
type('abc')
c1 = "L'eau vive"
c2 = ' est "froide" !'
c1+c2
c1*3
c1[0]
c1[-1]
c1[2:5]
len("abc")
"abcde".split("c")
'a-ha'.replace('a', 'o')
'-'.join(['ci', 'joint'])
'abracadabra'.count('bra')
'PETIT'.lower()
'grand'.upper()
```

Exercice 8. Les listes
Une liste est une collection hétérogène, ordonnée et modifiable d'éléments séparés par des virgules, entourée de crochets. Dans l'interpréteur Python, exécutez :

```
my_list = [4, 7, 3.7, 'E', 5, 7]
type(my_list)
```

```
len(my_list)
my_list[0]
my_list[-1]
my_list[1:3]
[0,1] + [2,4]
list(range(5))
list(range(2,9,2))
```

Quelques méthodes pour les listes.
Dans l'interpréteur Python, exécutez :

```
nombres =[17, 38, 10, 25, 72]
nombres.sort()
nombres
nombres.append(12)
nombres
nombres.reverse()
nombres
nombres.index(17)
nombres[0]
nombres
nombres[0]=11
nombres.remove(38)
nombres
nombres[1:3]
nombres[1:3]=[14, 17, 2]
nombres
nombres.count(17)
```

Un liste est de type “référence" :

```
x = ['a']
y = x
y[0] = 1
x
```

Exercice 9. tuples
Un tuple est une collection hétérogène, ordonnée et immuable (non-modifiable) d'éléments séparés par des virgules, entourée de parenthèses. Dans l'interpréteur Python, exécutez :

```
t = (5,7)
type(t)
t[0]
t[0] = 2
```

```
(x,y) = t
x
(x,y) = (y,x)
x
```

Exercice 10. Conversion de types
On utilise les fonctions `int()`, `float()`, `str()`. Dans l'interpréteur Python, exécutez :

```
i = 3
str(i)
i = '456'
int(i)
float(i)
list((1,2,3))
tuple([1,2,3])
list('azerty')
```

Exercice 11. Prédire le résultat des instructions suivantes :

```
(1+2)**3
"Da" * 4
"Da" + 3
("Pa" + "La") *2
("Da" * 4) / 2
5 / 2
5 // 2
5 % 2
```

1.3 Affichage

Exercice 12. Dans l'interpréteur Python, exécutez :

```
print('Hello') ; print('Joe')

print('Hello', end="") ; print('Joe')

print('Hello', end=" ") ; print('Joe')

x = 32
nom = "John"
print(nom, "a", x, 'ans')
print(nom, "a", x, 'ans',sep="-")
```

Exercice 13. Formatage des chaînes de caractères
Créez un script addition.py contenant :

```
x=1/3
y=40/3
z=7/3
print("{:10.3f}".format(x))
print(f"{y:10.3f}")
print("{:10.3f}".format(z))
print("-"*10)
print("{:10.3f}".format(x+y+z))

print(f"On a bien {x} + {y} + {z} = {x+y+z}")
```

1.4 Tests

Exercice 14. Créer un script tests.py contenant :

```
x = int(input("Donnez la valeur de l'entier x: "))

if x < 0:
    print("x est négatif")
    print("x est même strictement négatif")
elif x % 2:
    print("x est positif et impair")
else:
    print("x n'est pas négatif et est pair")
```

Remarquer que les blocs d'instructions sont déterminés par l'indentation (soit avec tab soit avec 4 espaces mais pas les deux en même temps)

1.5 Boucles

Exercice 15. Boucle for
Dans l'interpréteur Python, exécutez :

```
for lettre in "ciao":
    print(lettre)

for x in ["\n", 2, 'a', 3.14, '\n']:
    print(x)

x = [k**2 for k in range(20,10,-1)]
sum(x)
```

```
for n in range(2, 10):
     for x in range(2,n):
             if n % x == 0:
                     print(n, 'est égal à', x, '*', n//x)
                     break
     else:
             print(n, 'est premier')
```

Exercice 16. Créer un script pyramide.py qui dessine une pyramide comme celle-ci :

```
      *
     ***
    *****
   *******
  *********
```

Le nombre de lignes sera demander à l'utilisateur.

Exercice 17. Boucle while Créer un script test_while.py contenant :

```
N=0
x = int(input("Entrez un nombre entier positif : "))

while x>0:
    x//=2
    N+=1
print(f"Une approximation de log_2({x}) est {N}")
```

Exercice 18. Programmer le jeu du plus ou moins dont les règles sont les suivantes : l'utilisateur doit deviner un nombre entier que l'ordinateur a tiré au hasard entre 0 et 99. A chaque coup l'utilisateur peut faire une proposition et l'ordinateur indique si le nombre saisi est plus grand ou plus petit que le nombre à trouver. Une fois trouvé, le nombre de coups utilisés est affiché.
Pour obtenir un nombre entier au hasard compris entre 0 et 99 utiliser l'instruction :
`random.randint(0,99)`
et ajouter la ligne
`import random`
au début de votre script.