# Python 3 –TDs-Fipa1

## Saison 1 - épisode 2



# 1 Découvrons et testons le langage Python

## 1.1 Séquences d'instructions

Les instructions d'un programme s'exécutent les unes après les autres, dans l'ordre où elles ont été saisies !

**Testons !**

```
>>> x,y = 4,8
>>> x=y
>>> y=x
>>> print(x,y)
8 8
>>>
>>> x,y = 4,8
>>> y=x
>>> x=y
>>> print(x,y)
4 4
```

## 1.2 Script Python

Programmation itérative !

La programmation par script s'impose vite à nous.

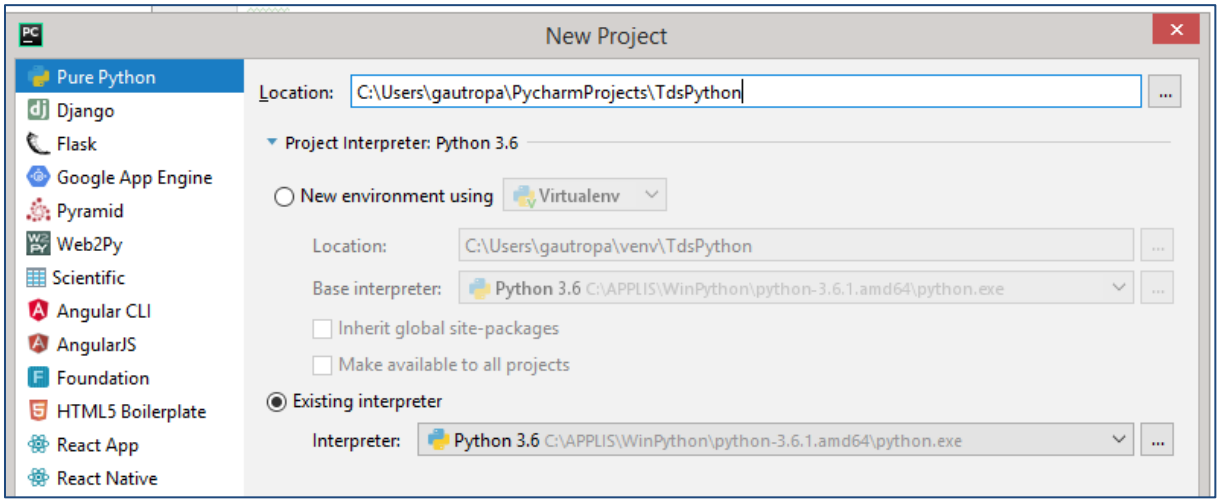
Il s'agit de créer un fichier dans lequel nous saisirons les instructions.

Commençons par définir notre environnement de travail avec Pycharm.

### 1.2.1 Rappel de l'épisode 1

## 1. Créer un « projet » pour l'ensemble des Tds Python

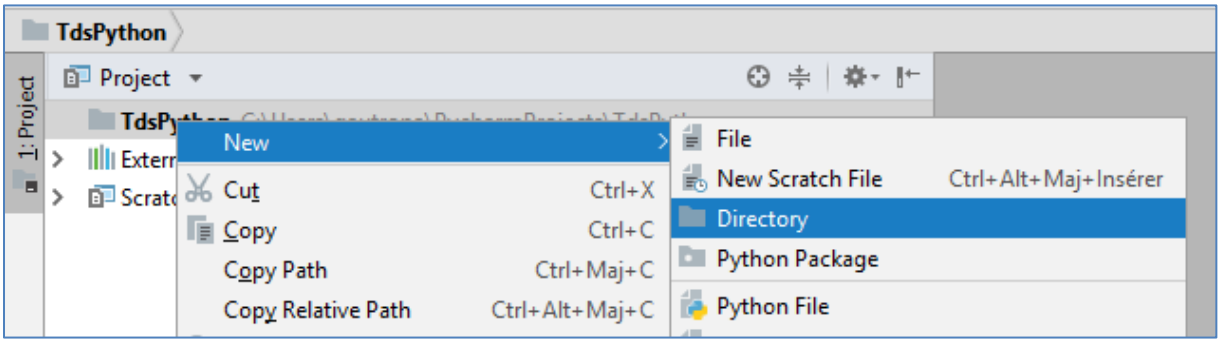
*Clic droit sur File > New Project*



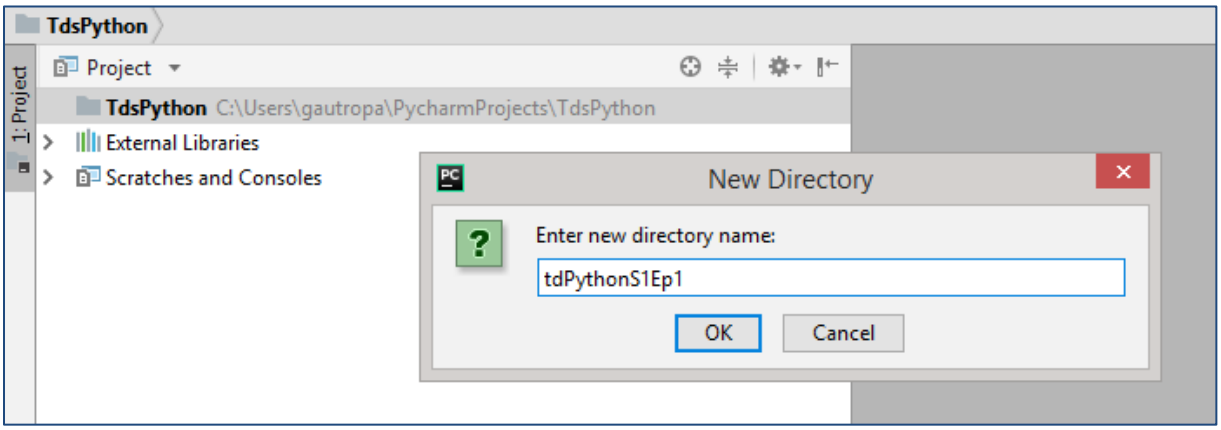

*Nommons le projet « TdsPython »puis validons en cliquant sur le bouton « **create** » en bas à droite de la fenêtre*

## 2. Créer une « directory » dans le projet pour chaque épisode

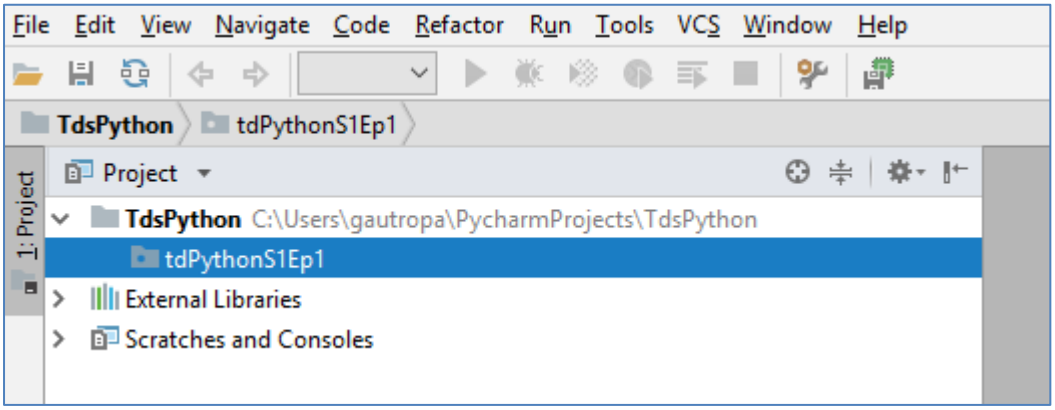
*Clic droit sur File > New > Directory ou Clic droit sur le projet TdsPython*




*Nommez la par exemple « tdPythonS1Ep1 »*

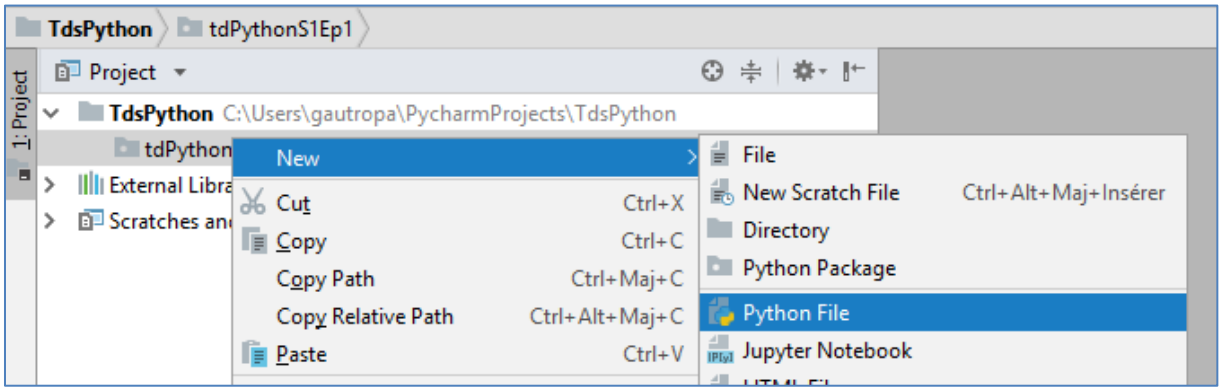


Résultat : la directory « tdPythonS1Ep1 » est ajoutée  dans le projet « TdsPython.

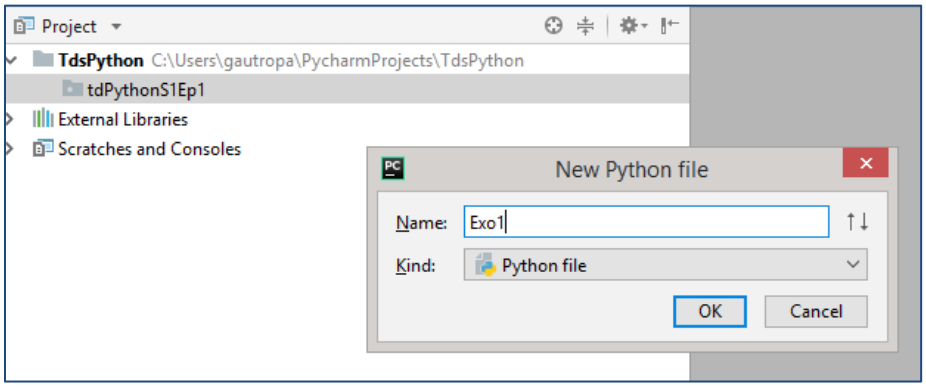



## 3. Créer un fichier (.py) dans une directory pour chaque script

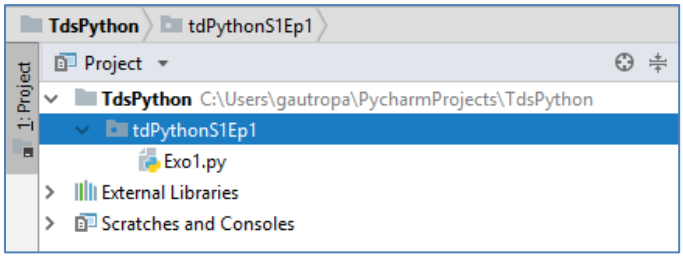
*Clic droit sur la directory tdPythonS1Ep1> new> Python File*




*Nommons le fichier puis validons en cliquant sur le bouton « **OK** »*

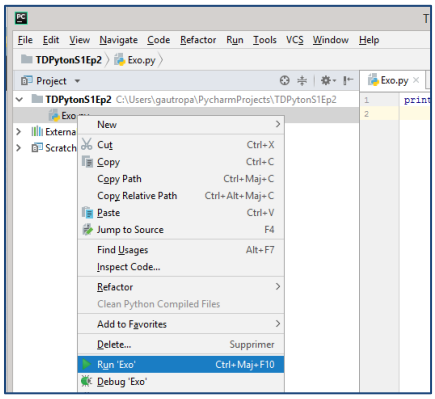



Résultat : le fichier python Exo1.py dans la directory tdPythonS1Ep1

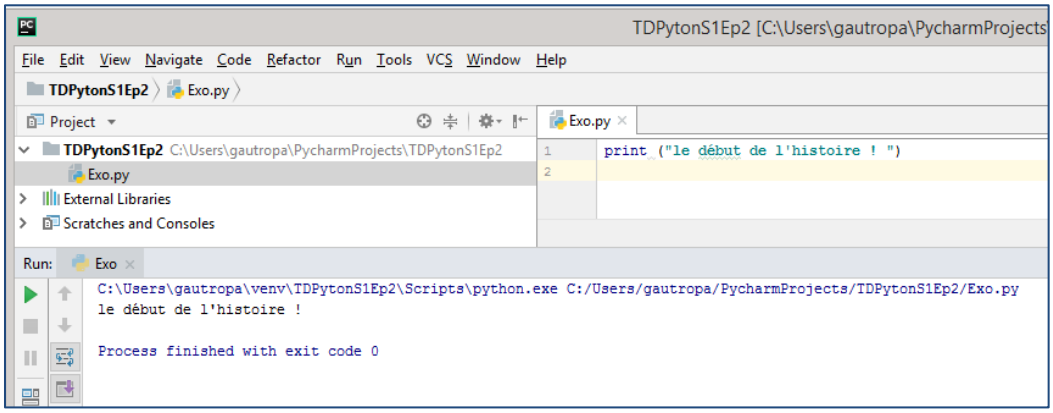


## 4. **Exécuter un script dans une** console

*Voir menu **Run** ou clic droit sur nom du script à exécuter puis **Run***




Résultat : la fenêtre de la Console d'exécution s'affiche (*# pour les commentaires)*

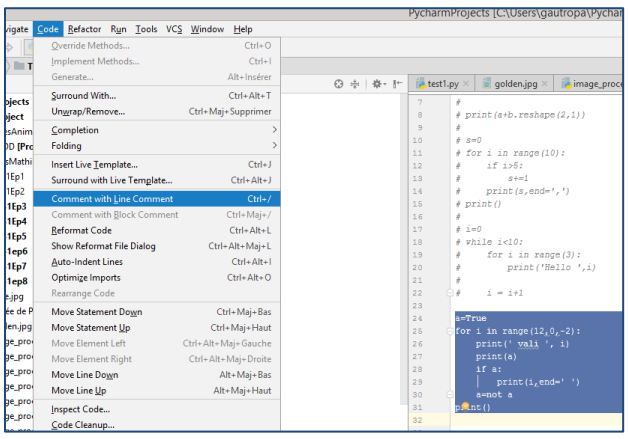



**Testons !** Reprenons un des exercices du TD précédent et créons un script !

## 5. **Mettre des lignes [de codes] en commentaires**

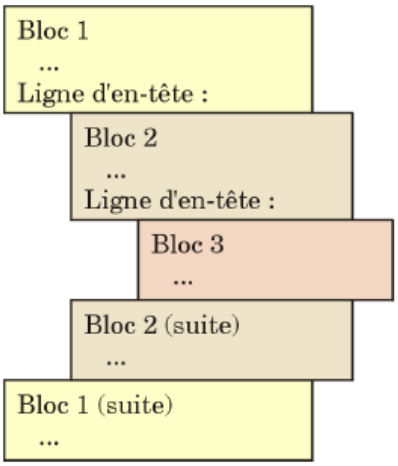
*Le caractère # permet de mettre **1 ligne** en commentaire*

*Voir aussi Menu **Code>Comment with Line Comment** après sélection **des lignes** à mettre en commentaires !*



## 1.3 Principe de saisie des instructions composées – blocs d'instruction

- Les blocs d'instructions sont toujours associés à une ligne d'en-tête contenant une instruction bien spécifique (if, elif, else, while, def, etc.) *se terminant par un double point*.
- Les blocs sont *délimités par l'indentation* : toutes les lignes d'un même bloc doivent être indentées exactement de la même manière (c'est-à-dire décalées vers la droite d'un même nombre d'espaces). Le nombre d'espaces à utiliser pour l'indentation est quelconque, mais la plupart des programmeurs utilisent des multiples de 4.
- Notez que le code du bloc le plus externe (bloc 1) ne peut pas lui-même être écarté de la marge de gauche (il n'est imbriqué dans rien).




***Source - Figure de Gérard Swinnen « Apprendre à programmer avec Python3 »***

## 1.4 Exécution conditionnelle

### 1.4.1 Choisir : if - [elif] - [else]

***Les limites des instructions et des blocs sont définies par la mise en page***

Contrôler une alternative :

```
>>> a = 10
>>> if a>0 :
...     print("a est positif")
... elif a< 0 :
...     print("a est negatif")
... else:
...     print("a est null")
...
a est positif
>>>
```

La condition évaluée peut contenir des opérateurs de comparaison :

**==**
**!=**
**> et >=**
**< et <=**

```
>>> a=8
>>> if (a%2==0):
...     print("a est pair")
... else:
...     print("a est impair")
...
a est pair
>>>
```

**Testons !**

On peut utiliser un équivalent d'opérateur ternaire :

```
>>> x=6
>>> y=5
>>> z=x if x < y else y
>>> print(z)
5
>>>
```

Expressions imbriquées :

```
>>> if manger:
...     if boire :
...         if dormir:
...             print("dormir")
...         print("boire")
...     print("manger")
... else:
...     print("perdu !")
```

## 1.5 Instructions répétitives

### 1.5.1 Répéter while

Répéter une portion de code tant qu'une expression booléenne est vraie (vrai) :

```
a=0
while a<5:
    a=a+1
    print("Hello")
print ("valeur de a",a)
```

Utilisation classique :

```
n = int(input("Entrez une valeur entre 1 et 10 :"))
while not(1 <= n <= 10) :
    n = int(input("Entrez un entier entre 1 et 10 svp :"))
```

### 1.5.2 Parcourir : for

Parcourir un ***itérable***, c'est-à-dire un conteneur que l'on peut parcourir élément par élément !

```
>>> for lettre in "ô rage ":
...     print(lettre)
...
ô

r
a
g
e
```

```
>>> for x in range(5):
...     print(x,end=" ")
...
0 1 2 3 4 >>>
```

## 1.6 Ruptures de séquences

***break*** pour **interrompre** une boucle :

```
>>> for x in range(1,6):
...     if x==4:
...         break
...     print(x,end=" ")
...
1 2 3 >>>
```

***continue*** pour **court-circuiter** une boucle :

```
1 2 3 >>> for x in range(1,6):
...     if x==4:
...         continue
...     print(x,end=" ")
...
1 2 3 5 >>> # la boucle a sauté la valeur 4
```

# 2 Mise en pratique

### Exercice 1

L'utilisateur donne un entier positif n et le programme affiche « PAIR « s'il est divisible par 2, sinon « IMPAIR »

### Exercice 2

L'utilisateur donne un entier positif et le programme annonce combien de fois de suite cet entier est divisible par 2.

### Exercice 3

Ecrivez un programme qui affiche les 20 premiers termes de la table de multiplication par 7, en signalant au passage (à l'aide d'un astérisque) ceux qui sont des multiples de3.

Exemple : 7 14 21 * 28 35 42 * 49 ...

### Exercice 4

Ecrivez un programme calculant le nombre de chiffre de  factorielle 1000 : 1000 !

## Exercice 5

La suite de Fibonacci est définie par la relation :

$$\begin{cases} u_0 = 0 \\ u_1 = 1 \\ \forall i > 1, u_n = u_{n-1} + u_{n-2} \end{cases}$$

Pour calculer cette suite, il faut deux variables annexes que l'on nommera x et y.
Initialement, x=0, et y=1.
À chaque itération :

- on calcule z=x+y ;
- x prend la valeur de y ;
- y prend la valeur de z.

À la dernière itération, **y** a la valeur désirée (testez avec 267914296).

Écrivez un programme permettant de calculer les valeurs prises par la suite de Fibonacci. La dernière affiche la valeur de y pour une suite valant 267914296.

Afin de vérifier votre programme, voici les premières valeurs prises par la suite de Fibonacci :

| n | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| F(n) | 0 | 1 | 1 | 2 | 3 |

## Exercice 6

Le but de l'exercice est d'écrire un programme de calcul approché de PI par une méthode itérative. On peut obtenir une valeur approchée de PI de la manière suivante :

$$\pi = (s_n + 4)/2$$ avec $$s_n = \sum_{i=0}^{n} u_i$$ et $$u_i = \frac{8(-1)^i}{(2i+1)(2i+3)}$$

Quelle valeur fournit le programme après 100000 itérations ?

## Exercice 7

Ecrivez un script qui détermine si une chaîne contient ou non le caractère « y ». Affichez la fréquence de ce caractère.

## Exercice 8

Ecrire un script qui recopie une chaîne  (dans une nouvelle variable) en l'inversant.

Ainsi par exemple « bizar » donnera « razib »