Python peut être utilisé directement à l'aide d'une console en mode interactif. A chaque pression de la touche **return**, la réponse aux commandes écrites apparaîtra directement. Mais il peut aussi être utilisé avec un éditeur ce qui permettre de créer des programmes (scripts), de les sauvegarder et de les lancer (compiler) pour les tester. Le logiciel utilisé cette année est **spyder**, il possède un espace console et un éditeur.

Lancer le à l'aide de l'icône

Dans ce document, quand le code doit être entré directement dans la console, il sera précédé de >>> et quand il sera demandé d'utiliser l'éditeur, le logo  sera présent.

Calculs :

Exercice 1. Directement dans la Console de Spyder, recopier et exécuter (touche "Entrée") successivement les instructions suivantes :

```
>>> 1+2
>>> 1+3.5
>>> 1+3/2
>>> -1+2*3
>>> (-2+1)*(-4)
>>> 3**2
>>> 9**2
>>> 2**(1/3)
```

Exercice 2. Effectuer le calcul suivant directement dans la console $A = \dfrac{-2+\dfrac{11^2}{8}}{\dfrac{5}{3^{-2}+\dfrac{9}{4}}-1}$

À retenir

Compléter le tableau

Opération	Opérateur
Somme	+
Produit	*
Quotient	
Puissance	

Les fonctions mathématiques usuelles ne sont pas disponibles immédiatement à l'ouverture de Spyder ; il faut charger une librairie complémentaire : le module numpy.

Exercice 3. Directement dans la console,

```
>>> log(2)
>>> import numpy
>>> log(2)
>>> numpy.log(2)
>>> from numpy import log
>>> log(2)
```

Le nom du module à importer peut être un peu long, et il est alors peu pratique de l'écrire à chaque import de fonction. On peut alors utiliser la commande **import**...**as**.

Exercice 4. Directement dans la console,

```
>>> import numpy as np
>>> np.log(2)
```

À retenir

Pour importer et utiliser des fonctions d'un module :
Méthode 1 :
- ⋆ Importer le module numpy avec **import** numpy
- ⋆ Lister les commandes du module, utiliser la commande **dir()**
- ⋆ Utiliser une fonction du module numpy avec numpy.*nomdelafonction*
- ⋆ Il peut être pratique d'utiliser une abréviation du module en utilisant **import** numpy **as** np

Méthode 2 :
- ⋆ Importer une fonction du module numpy **from** numpy **import**
- ⋆ Importer toutes les fonctions du module numpy **from** numpy **import***. Cette méthode n'est pas la plus recommandée car il peut y avoir conflit avec les fonctions d'autres modules.

Exercice 5. Directement dans la console,

```
>>> import numpy
>>> dir numpy
>>> help(numpy)
>>> help(numpy.log)
>>> from numpy  import log
>>> help(log)
```

À retenir

- ⋆ Afficher la liste des fonctions présentes dans le module numpy avec la commande **dir()**.
- ⋆ Afficher l'aide d'un module ou seulement d'une fonction de ce module avec la commande **help()**.

Exercice 6. Effectuer le calcul suivant dans la console

$$B = \frac{\sqrt{2}+3}{\ln(2)+\mathrm{e}^2}$$

Types

Dans les calculs précédents, on a pu voir des nombres écrit avec virgule ou sans. Python considère différents types de nombres. Les entiers et les flottants (nombres à virgule)

Exercice 7. Directement dans la console

```
>>> type(3)
>>> type(3.0)
>>> type(3.5)
```

Il existe une fonction pour convertir un flottant en entier

Exercice 8. Directement dans la console

```
>>>int(3.5)
>>>int(3.0)
```

Les entiers peuvent être convertis en flottant

Exercice 9. Directement dans la console

```
>>>float(3)
```

Exercice 10. Afficher dans la console le résultats $\frac{3}{4}+\frac{1}{4}$ et son type

Il y a un module pour travailler avec les fractions.

Exercice 11. Directement dans la console

```
>>> from fractions import Fraction
>>> Fraction(3,4)
>>> Fraction(3,4)+Fraction(1,4)
>>> Fraction(3,4)+0.25
```

Il existe le type chaînes de caractères.

Exercice 12. Directement dans la console

```
>>> type(ceci est une chaîne de caractère)
>>> type('ceci est une chaîne de caractère')
>>> type ('1')
>>> '1'+1
>>> type(int('1'))
>>> type(float('1'))
```

Il existe d'autres types comme les listes, t-uples... qui seront étudiés plus en détail plus tard

Exercice 13. Directement dans la console

```
>>> type((1,2,3))
>>>  type([1,2,3])
>>> type(['1',2,'trois'])
>>> type(('1',2,'trois'))
```

À retenir

- ⋆ int() transforme un nombre en entier
- ⋆ float() transforme un nombre en flottant(nombre à virgule)

Variables

Les noms de variables doivent être choisis aussi explicites que possible, de manière à exprimer clairement ce que les variable sont censées contenir. Elle permette de manipuler les objets (nombres, chaîne de caractères, liste...) plus facilement.

À retenir

Sous Python, les noms de variables doivent en outre obéir à quelques règles simples :

- ⋆ Un nom de variable est une séquence de lettres et de chiffres qui doit toujours commencer par une lettre.
- ⋆ Seules les lettres ordinaires sont autorisées. Les lettres accentuées, les cédilles, les espaces, les caractères spéciaux sont interdits, à l'exception du caractère souligné.
- ⋆ La casse est significative (les caractères majuscules et minuscules sont distingués).Prendre l'habitude d'écrire l'essentiel des noms de variables en caractères minuscules
- ⋆ voilà les noms à ne pas utiliser pour un nom de variable

and	assert	break	class	continue	def
del	elif	else	except	exec	finally
for	from	global	if	import	in
is	lambda	not	or	pass	print
raise	return	try	while	yield	

Exercice 14. Directement dans la console

```
>>>x=3
>>>print(x)
>>>type(x)
>>>float(x)
>>>x=7.
>>>type(x)
>>>int(x)
>>>print(x+2)
>>>print('x')
>>>type('x')
>>>print('x'+2)
>>>print(X)
>>>x=x+2
>>>print(x)
>>>x=x+7
>>>4*x
>>>x='x'
>>>x=4*x
>>>clear
>>>print(x)
>>>del x
>>>print(x)
>>>x=y=2
>>>print(x+y)
>>>x,y=3,2
>>>x=x/3
>>>type(x)
>>>x**2+2
>>> y=y**x
>>>print(y)
```

La commande **input** permet de demander à l'utilisateur de faire une entrée

Exercice 15. Directement dans la console

```
>>> a=input('entrer un entier')
>>>print(a)
>>>type(a)
```

Comme remarqué, l'entrée est considérée comme une chaine de caractère. Il faudra préciser le type du nombre qu'on attend.

Exercice 16. Directement dans la console

```
>>>a=int(input('entrer un entier:'))
>>> print(a)
>>>type(a)
```

À retenir

- ⋆ La commande **input** permet de demander une entrée à l'utilisateur.
- ⋆ L'entrée est considérée comme un chaîne de caractère, il faudra changer le type si on veut un nombre.

Avec l'éditeur :

Spyder possède un éditeur, permet de taper plusieurs instructions à la suite et d'ensuite les exécuter, une fois toutes tapées. Il est donc plus aisé de travailler le code, le corriger...

À retenir

- ⋆ Une fois tapé dans l'éditeur , enregistrer le script avec l'extension **.py**
- ⋆ cliquer sur le bouton ▷ ou dans l'onglet **run**, sur la commande **run** pour lancer le programme.
- ⋆ On peut aussi aller le chercher à partir de la console en tapant le chemin y menant et son nom par exemple **runfile('/programsfiles/spyder/.../exemple.py')**

Exercice 17. Dans l'éditeur, taper puis exécuter

```
n=input('entrer un entier n?')
n=n*2
print(n)
```

Exercice 18. Dans l'éditeur, taper puis exécuter

```
x=3
x=2*x+4
x=15-x**2
print(x)
```

Exercice 19. Dans l'éditeur, taper puis exécuter

```
x=int(input('donnez un entier x :'))
y=int(input('donnez un entier y :'))
x=x+y
y=x-y
x=x-y
print('après traitement :')
print( 'x=', x, 'et y=', y)
```

1. Tester le script avec des valeurs de x et y.
2. Que fait-il ?
3. Proposer un script plus court qui aurait pu aboutir au même résultat. (Une 3e variable est nécessaire)

Exercice 20. Écrire un programme qui demande à l'utilisateur d'entrer un nombre flottant x, et qui affiche : le carré, la racine carrée, la partie entière, l'inverse, l'exponentielle, le logarithme, la valeur absolue du nombre. Le sauvegarder et le tester sur plusieurs fois.

Exercice 21. Écrire un programme qui permet de calculer le discriminant d'un polynôme du second degré. Le sauvegarder sous le nom "discriminant" et le tester sur plusieurs fois.

Exercice 22. Écrire un programme qui demande le temps, exprimé en secondes, et qui le transforme en $h : m : s$..