1 Rapides rappels de Python

Exercice 1. Que vaut la variable `a` à la fin de l'exécution des deux scripts suivants ?

```
a=-1
if a>0:
    a=2
a=a+1
```

```
a=-1
if a>0:
    a=2
    a=a+1
```

Exercice 2. Dresser le tableau de suivi des variables lors d'une exécution ligne à ligne pour le code Python suivant.

Code Python :

```
x=3
y=x+2
x=y
print(y)
```

Tableau de suivi des variables :

Ligne	x	y
1		
2		
3		
4		

Exercice 3. Etant donné une variable x, et une y, écrire un script pour que la valeur contenue dans la variable x soit dans la variable y, et celle de la variable y dans x.

Exercice 4. Donner le type des variables suivantes :

```
x=3
s="cha"
nombre="4"
entier=4.43
d=1.0
L=True
y=x+d
n=s+nombre
```

Exercice 5. Qu'affiche les lignes suivantes ?

```
5//2
5/2
8%2
5%3
12%17
```

Exercice 6.

```
def signe(x):
    if x > 0:
        print("Strictement positif")
    else:
        print("Negatif")
```

1. Que se passe-t-il quand on execute ce code ?
2. Que faut-il écrire pour obtenir une sortie ?
3. Modifire cette fonction pour également séparer les cas "strictement négatif" et "nul".

Exercice 7. Qu'affiche le code suivant ?

```
x=3
def f(x):
    return x**2
def g(y):
    return x+y
print(f(5))
print(g(5))
```

Exercice 8. Donner le résultat de chaque ligne.

```
True and True
True and False
False or True
not(True)
not(True and (False or True))
```

Exercice 9. Ecrire une fonction qui prend en entrée un entier et renvoie True s'il est pair, False sinon.

Exercice 10. Quels sont les types des variables définis ci-dessous ?

```
x = 5%2 == 0
y = 5/2
z = 5//2
t=1.0+5
```

Exercice 11. Boucle For

1. On définit une suite (u_n) pour n entier naturel par $u_0 = 5$ et $u_{n+1} = 4u_n + 3$. Ecrire un script pour calculer u_1, puis u_2, puis u_{10}.
2. Qu'affiche le script suivant ?

```
for i in range(2, 4, -1):
    print(i)
for i in range(1, 6, 2):
    print(i)
```

3. Dresser le tableau de suivi des variables pour le programme suivant.

```
u=3
v=4
for i in range(4):
    u=u+2
u=v-u
print(u)
```

4. Ecrire une fonction qui calcule u_n étant donné n, avec (u_n) défini pour n entier naturel par $u_0 = 5$ et $u_{n+1} = 3u_n + 2 + n$.

Exercice 12. Boucle While :

1. On considère la suite $(u_n)_{n\in\mathbb{N}}$ définit par $u_0 = 10$ et $u_{n+1} = u_n + 5$ pour $n > 0$. Quelle est la limite de cette suite ? Ecrire un script qui affiche le premier rang n pour lequel $u_n > 100$.

2. Dresser un tableau de suivi des variables. Que fait cette suite d'instructions ?

```
a=30
div=4
q=0
while u > 0 :
    a=a-4
    q=q+1
print(q)
```

3. (a) Ecrire une fonction qui prend en entrée un premier terme u0, et une raison r positive, et renvoie le rang du premier terme supérieur à 100 de la suite arithmétique de premier terme u0, et de raison r.
   (b) Modifier la fonction pour que le seuil soit donné en argument.
   (c) La raison n'est plus supposée positive, modifier la fonction pour renvoyer -1 si la suite ne dépasse jamais le seuil, et le rang de dépassement sinon.

2 Prise en main de Python et Pyzo

1. Ouvrir Pyzo, et créer un nouveau fichier dans un dossier dévolu à l'ITC.
2. On enregistrera souvent son travail, avec le raccourci `Ctrl+S`.
3. Recopier le script suivant.

```
print("Hello world")

## cellule 1
x=5
print("Execution cellule 1")
print("x=", x)

## cellule 2
y=3
x=2
print("Cellule2, x=", x)
```

Avec Pyzo, on peut executer, c'est à dire faire interpréter par Python à l'intérieur de la console, à l'aide de différents raccourcis, à retrouver dans l'onglet Run.

— `Alt+Entrée` (ou `Alt+Return`) exécute les lignes sélectionnés, ou à défaut la ligne où se trouve votre curseur.

— `Ctrl+Entrée` exécute la cellule où se trouve le curseur, c'est à dire la zone délimitée par des lignes commençant par `##`.

— `Ctrl+E` exécute le fichier en entier.

4. Tester ces différentes possibilités d'éxecutions.
5. Choisir les bonnes lignes/cellules à exécuter pour que la console affiche "Cellule2, x=5".
6. Essayer également de taper des lignes de code directement dans la console, par exemple "x" ou "x=x+1".

3 Premières exercices de programmation

Exercice 13. Suites arithmétiques

1. Ecrire une fonction qui prend en entrée q et $u0$ un premier terme, deux int, et une raison et renvoie True si la suite géométrique de raison q et de premier terme u_0 est convergente, False sinon.
2. Ecrire une fonction qui prend les mêmes entrées et affiche une chaîne de caractère indiquant si la suite est convergente ou divergente.

Exercice 14. Condition

1. Ecrire une fonction `solution_snd_degre` qui prend en entrée a, b et c et renvoie un entier correspondant au nombre de solutions distinctes de l'équation $ax^2+bx+c=0$ avec x un réel.
2. Ecrire une fonction `valeur absolue`, qui demande à l'utilisateur un nombre x, affiche x si x est positif, -x s'il est négatif.
3. Ecrire une fonction `triangle`, qui prend en entrée les longueurs de 3 côtés et dit si le triangle est rectangle, s'il est isocèle ou s'il est équilatéral.

Exercice 15. Affichage et table de multiplication

1. Tester le code suivant :

```
x=5
y=2
chaine=" x + y"
print("La valeur de x est ", x)
print(" et celle \t de y est ", y, "\nEt on a", chaine, "=", x+y)
```

2. Que font `t` et `n` ?
3. Ecrire une fonction `table_de_multiplication_7()` qui affiche la table de multiplication de 7, sous la forme :

```
7  x  1  =  7
7  x  2  =  14
...
7  x 10  = 70
```

4. Ecrire une nouvelle fonction `table_de_multiplication(n)` qui affiche de même la table de multiplication de l'entier n. On n'hésitera pas à recourir au copier-coller avec le raccourci usuel `Ctrl+C, Ctrl+V`.
5. Ecrire une fonction qui prend en entrée un entier `n` et affiche sa table de multiplication s'il est inférieur à 10, affiche son carré s'il est inférieur à 25, et affiche "Rien à connaître sur ce nombre" s'il est supérieur à 25. On pourra appeler la fonction `table_de_multiplication`, mais on ne recopiera pas le code de celle-ci dans notre nouvelle fonction.

Exercice 16. Puissances de 2

Ecrire une fonction qui prend en entrée un entier seuil et renvoie toutes les puissances de 2 inférieurs à ce seuils, avec leur rang. Par exemple pour un seuil de 5 on veut afficher :

```
2**0 = 1
2**1 = 2
2**2 = 4
```

Exercice 17. Maximum

1. Ecrire une fonction qui prend en entrée deux entiers et renvoie le plus grand des deux (sans recourir à la fonction max).
2. Ecrire une fonction qui prend en entrée trois entiers et renvoie le plus grand des trois (sans recourir ni à max, ni au opérateurs de comparaisons, mais vous pouvez utiliser la fonction définie au dessus).
3. De même pour 4 entiers.

Exercice 18. Forêt

Une forêt compte 10 000 arbres. Chaque année on coupe 10 % de la forêt, et on replante 50 arbres. Au bout de combien d'année restera-t-il moins de 100 arbres dans la forêt ?

Exercice 19. Des dés

On veut savoir le nombre de façons d'obtenir un certains score n comme somme des valeurs sur les faces de trois dés à 6 faces numérotées de 1 à 6. Il y a par exemple 3 façons d'obtenir 4 (1+1+2, 1+2+1, 2+1+1).

1. Ecrire une fonction qui, étant donnée n, renvoie ce nombre.
2. Quelles valeurs peut prendre n ?
3. Etendre la fonction précédente pour ajouter une entrée de type bool, "affiche". Si `affiche` est vrai, en plus de renvoyer à la fin le nombre de façon d'obtenir n avec 3 dés, on affiche toutes les façons de faire.(par exemple une des lignes affichera 1+3+2 pour 6).
4. Quelle valeur de somme de trois dés sort le plus souvent ?

Exercice 20. Proposer trois façons d'afficher les 100 premiers entiers pairs.

Exercice 21. Factorielle

La factorielle d'un entier positif n est un entier, noté $n!$ (lire : factorielle n), défini par : $0! = 1$ et pour $n > 0$, $n! = (n-1)! \times n$.

1. Ecrire une fonction calculant $n!$ en fonction de n.
2. Ecrire une fonction qui renvoie la première valeur n tel que $n! > 10^{10}$
3. Ecrire une fonction qui renvoie la première valeur n tel que $1! + 2! + ... + n! > 10^{15}$

On considère la suite (u_n) défini par $u_n = \sum_{k=0}^{n} \frac{1}{n!}$, pour n entier naturel.

4. Ecrire une fonction calculant u_n en fonction de n.

On considère la suite (v_n) défini, pour n entier naturel non nul, par $v_n = \left(1 + \frac{1}{n}\right)^n$.

5. Ecrire une fonction calculant v_n en fonction de n.

On peut montrer que les deux suites (u_n) et (v_n) sont toutes deux convergentes, de même limite la constante e. On peut obtenir la valeur de la constante e dans Python dans le module math par exemple, avec l'instruction :

```
from math import e
```

6. Ecrire une fonction qui prend en entrée un seuil epsilon, et renvoit le 1er rang tel que u_n soit distant de e de moins de epsilon, de même pour v_n. Comparer les vitesses de convergence.

Exercice 22. Nombre premier

Ecrire une fonction qui prend en entrée une entier n et renvoie True si celui-ci est premier, False sinon. On pensera à utiliser l'opérateur modulo %. On utilisera une boucle While, et on pourra utiliser la fonction `sqrt` du module `math` qui calcule la racine carré (sqrt : SQuare RooT).

Exercice 23. Un grand nombre
456^{231} est un nombre composé de nombreux chiffres en base 10, on veut trouver le plus grand produit de 5 chiffres consécutifs présent dans l'écriture en base 10 de ce nombre. Par exemple, pour 347451218 le plus grand produit de 3 chiffres consécutifs est $7 \times 4 \times 5 = 245$.

1. Ecrire une fonction qui prend en entrée un entier et renvoie le plus grand produit de 5 chiffres consécutifs dans l'écriture en base 10 de ce nombre.
2. La tester avec 456^{231}.
3. Modifier cette fonction pour que le nombre de chiffres consécutifs à considérer soit donné en entrée.
4. Modifier cette fonction pour que les chiffres qui constituent le produit maximum soit affichés ainsi que leur position dans le nombre.

Exercice 24. Algorithme d'Euclide
Implémenter l'algorithme d'Euclide, qui permet de calculer le PGCD de deux entiers positifs, en utilisant seulement la soustraction et la comparaison à 0 `>=0`. On pourra commencer par écrire une fonction implémentant l'algorithme de la division euclidienne qui permet de calculer le dividende et le reste dans la division euclidienne. Si vous ne connaissez pas l'algorithme d'Euclide vous pouvez faire une recherche sur internet, et par exemple aller

Exercice 25. N'hésitez pas à vous entraîner en implémentant et en testant les programmes des exercices traités sur papier.

Exercice 26. Chaîne de caractère

1. Tester le code suivant :

```
s="Bonjour"
for c in s :
    print(c)
```

2. Puis le code suivant :

```
s="Bonjour"
print(s[5])
print(len(s))
```

3. Que renvoie `len(s)` et `s[i]` pour i un entier, et quelles valeurs peut prendre `i` ?
4. Ecrire d'une manière différente de la question 1 un script qui affiche les caractères d'une chaîne les un après les autres.
5. Ecrire une fonction qui teste l'appartenance d'un élément `e` donné en entrée à une chaîne de caractère. On proposera une version avec une boucle For, et une avec une boucle While.
6. Ecrire une fonction qui compte le nombre d'occurence d'un élément dans une chaîne.
7. Ecrire une fonction `miroir` qui prend en entrée une chaîne de caractère est renvoie la chaîne avec les mêmes caractères dans le sens inverse. Par exemple "Bonjour" doit devenir "ruojnoB".