Exercices – **Entraînement de début d'année**

Exercice 1 : Somme & moyenne

1) Ecrire un programme qui demande 10 nombres réels à l'utilisateur puis affiche leur somme.
2) Ecrire un programme qui demande 10 nombres réels à l'utilisateur puis affiche leur somme et leur moyenne**.**
3) Modifier le programme pour qu'il demande le nombre de nombres à l'utilisateur.

Exercice 2 : La statistique du 6

1) Ecrire un programme qui tire des nombres au hasard entre 1 (inclus) et 6 (inclus) jusqu'à trouver 6. Le programme affiche à la fin le nombre de tirages effectués.
2) Ecrire un programme qui calcule la moyenne de ce nombre de tirages sur 10000 essais.
3) La même chose sur 100 000 essais, et sur 1 000 000 essais. Qu'est-ce qu'on observe ?

Aide : le module `random` *contient une fonction* `randint(a,b)` *qui permet d'obtenir des nombres aléatoires.*

Exercice 3 : Exécution mentale

Quelle est la valeur des variables après la séquence d'instructions suivante ?

1)
```
def f(x, y):
    x = 3 + y
    y = 2 * x
    return x + y

a = 5
b = -2
x = 3 * b + 10
a = f(a, 3) + f (1, b)
```

2)
```
def g(a):
    global b
    b = -a + b
    c = a + b
    return c

a = 1
b = 7
c = a - b
a = g(b) + g(a)
```

Exercice 4 : Bowling

Au bowling on a deux chances (deux boules) pour faire tomber un total de 10 quilles.

1) Ecrire un programme qui demande le nombre de quilles renversées avec chacune des deux boules et affiche « X » si toutes les quilles sont tombées à la première boule, « / » si toutes les quilles sont tombées à la deuxième boule et sinon le nombre de quilles renversées.
2) Reprendre le programme précédent en ne demandant les informations de la deuxième boule que si elle a besoin d'être lancée.

Exercice 5 : Programmes mystères

Que font les programmes suivant ?

1)
```
for i in range(100):
    if (i % 3) == 0:
        print(i)
```

2
```
a = 1
b = 0
while b < 100:
    if (a % 3) == 0:
        print(a)
        b = b + 1
    a = a + 1
```

3)
```
a = 21
b = 1
while a > 0:
    c = int(input("?"))
    a = a - c
    b = 3 - b
print(3 - b)
```

Exercice 6 : Barre verticale

Ecrire un programme qui demande deux entiers x et y à l'utilisateur puis affiche :

- x tirets (- du 6)
- une barre verticale (AltGr 6)
- y − x − 1 tirets

sans aller à la ligne entre les caractères. Par exemple, x = 5 et y = 12 doit donner l'affichage :
`-----|------` Comme vous le voyez, l'affichage comporte y caractères (en ce cas 12)

Aide : Pour éviter que l'instruction `print` *aille à la ligne, il suffit de rajouter* `,end=""` *à la fin des arguments.*

Exercice 7 : Barre verticale mobile

En vous inspirant de l'exercice précèdent, écrire un programme qui demande un entier y à l'utilisateur puis affiche une animation de l'affichage précèdent pour x qui varie entre 0 et y − 1.
Par exemple, y = 12 doit donner l'animation qui contient les affichages ci-contre :

```
|-----------
-|----------
--|---------
---|--------
----|-------
-----|------
------|-----
-------|----
--------|---
---------|--
----------|-
-----------|
```

Exercice 8 : K2000

En vous inspirant de l'exercice précèdent, écrire un programme qui demande un entier y à l'utilisateur puis affiche une animation où la barre verticale rebondit quand elle arrive à un bord.
Par exemple, y = 4 doit donner l'animation qui contient les affichages suivants :

```
|---
-|--
--|-
---|
--|-
-|--
|---
-|--
--|-
---|
--|-
etcetera
```

Aide : Si on veut vraiment faire une animation, il faut utiliser la fonction `sleep(t)` *du module* `time` *pour attendre* `t` *secondes et la commande* `system('cls')` *du module* `os` *pour effacer la console. Dans ce cas il faudra lancer le programme dans un terminal windows avec la commande « Run/Run current script in terminal (Ctrl-T) » de thonny. En tout état de cause, commencez toujours par faire une version sans animation pour vérifier que tout fonctionne comme vous le voulez avant d'essayer de faire l'animation.*

NB : Il est possible de faire une version optimisée avec seulement 2 boucles imbriquées

Exercice 9 : Les diviseurs

Ecrire un programme qui demande un nombre n et affiche la liste de ses diviseurs.
Exemple : 45 a pour diviseurs 1, 3, 5, 9, 15, et 45

Exercice 10 : Lost

Le but de cet exercice est de simuler une bombe inspirée d'une série américaine que vous reconnaitrez peut-être... ou pas. Cette bombe doit exploser si l'utilisateur ne rentre pas régulièrement le code "4 8 15 16 23 42" (en une seule fois avec les espaces au bon endroit). S'il se passe plus de 15 minutes sans que l'utilisateur n'ait rentré ce code, la bombe explose.

1) Faire première version du programme qui demande encore et encore à l'utilisateur de rentrer un code et qui s'arrête lorsque le code est correct.
2) Faire une version 2 du programme qui s'arrête avec un message explosif lorsque le code est incorrect.
3) Faire une version 3 du programme qui s'arrête lorsque l'utilisateur se trompe un total de 5 fois.
4) Faire une version 4 du programme qui s'arrête lorsque l'utilisateur se trompe deux fois de suite.