Un autre bouclage de Python

1 Introduction

Dans ce TP, nous allons voir une autre manière de faire des boucles.
En effet, lorsque l'on effectue des boucles on ne sait pas toujours combien de temps on va effectuer une boucle. C'est pourquoi, nous allons utiliser la boucle dite non bornée qui s'arrête lorsqu'une condition n'est plus vérifiée (variable booléenne).
Tester ces programmes :

```
i = 0
while i < 20:
      print(i < 100)
      print(i)
      i = i + 1
print(i < 100)
print(i)
```

```
i = 12
while i <= 42:
      i = i + 2
      print(i)
```

```
for i in range(12, 43, 2):
      print(i)
```

.....

1. Expliquer l'affichage du programme 1.
   .....
2. Pourquoi a-t-on `False` à l'avant dernière ligne de l'affichage ?
   .....
3. Comparer les avantages et les inconvénients entre les programmes 2 et 3.
   Taille de code : .....
   .....
   Compter les répétitions : .....
   .....
   Autre : .....
   .....

Il faut faire la boucle `while` ..... la condition (booléenne) ..... est égale à .....

Exercice 1 ❑

Réaliser un programme qui renvoie le plus grand entier `n` dont le cube est supérieur ou égal à 1728.

Exercice 2 ❑

Réaliser un programme qui teste et qui renvoie le premier `x` positif tel que : $(7 - 2x)(10 - 2x) = 35$

Astuce : Mettre `x` à 0 puis lui ajouter un pas.

Exercice 3 ❑

Show `n` first negative numbers without using a loop `for`.

Exercice 4 *Réponse forcée* ❑

1. Quelle est la condition qu'il faudrait avoir pour que la boucle `while` soit infinie ?

   .....

2. Compléter le programme ci dessous qui demande indéfiniment à l'utilisateur une chaine de caractère si ce dernier dit `'Bonjour'` Arréter la boucle à l'aide de la commande `break`.

```
while ..... :
 reponse = ....
  if reponse == ..... : break \#stop the loop
 else : pass \# Do nothing
```

Exercice 5 *Compte à rebours* ❑

1. Créer un programme qui demande un nombre entier de secondes et qui affiche le compte à rebours de toutes les secondes à partir de ce nombre. À la fin du décompte, il renvoie la chaîne de caractère `'go'`.
2. Modifier ce programme qui demande en un nombre entier en seconde et qui affiche le décompte sous la forme minutes secondes avec la commande : `print(minute, 'min', seconde, 's')`
3. **Bonus :** Ajouter les heures et modifier l'entrée pour avoir la possibilité de le faire soit en seconde ou en minute.

Commandes utiles : Pour faire ce programme, vous aurez besoin d'importer la fonction `sleep(seconde)` qui permet d'attendre le nombre de seconde voulu. Pour cette fonction, il vous faut importer bibliothèque `time` avec la commande `from time import *`.
Astuce : Penser à la division euclidienne avec les commandes : `//` et `%` .

Exercice 6 ❑

Afficher les diviseurs d'un nombre entier donné par l'utilisateur.

Exercice 7 *Quelques rebonds* ❑

Une balle rebondissante est lâchée du haut d'un immeuble de 10 mètres. A chaque rebond la hauteur diminue d'un tiers. Quand le rebond est inférieur à 1 cm on considère que la balle s'arrête.

1. Au bout de combien de rebonds la balle s'arrête.
2. Combien de fois la personne du premier étage dont la fenêtre est située à 1,5 m verra t'elle passer la balle ?

Exercice 8 *Nombre mystère* ❑

Réaliser le jeu du nombre mystère où l'ordinateur choisi un nombre aléatoire entre 1 et 100. Vous rentrez une valeur et l'ordinateur compare les valeurs, si c'est égale vous gagnez sinon l'ordinateur vous renvoie + si la valeur est plus grande - sinon.

```
from random import randint
mystere = randint(1, 100)
...
```