TP Python : Boucles

- Auteur : Gwénaël LAURENT
- Date : 15/09/2024

Exo_1 : Afficher les tables de multiplication

Ecrivez un script python qui affiche la table de multiplication d'un nombre lu au clavier.

Vous afficherez les 12 premières lignes de la table.
La valeur 12 doit être stockée dans une constante (cf à la fin de l'exercice).

> Utilisez l'instructions de boucle la plus adaptée pour cet exercice.

Exemple de résultat à obtenir :

```
Entrez un nombre entier : 7

Table de mutiplication de 7 :
7 x 0 = 0
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
7 x 11 = 77
```

> **C'est quoi une constante :**
>
> En programmation, **une constante est une variable en lecture seule**. On ne peut pas modifier sa valeur dans le code.
>
> Les constantes servent, par exemple, à enregistrer le paramétrage du programme.
> Les constantes sont **déclarées au début du script** (juste après les `import`)
>
> En langage Python, nous ne pouvons pas déclarer une variable ou une valeur comme constante . Mais nous pouvons indiquer aux programmeurs qu'une variable est une constante en utilisant un **nom de variable entièrement en MAJUSCULES**. Même si nous pouvons modifier la valeur, il est alors de la responsabilité du programmeur de laisser cette variable inchangée.

Exo_2 : Divisions entières par 2

Lecture d'un nombre entier positif au clavier puis :
Tant que ce nombre est plus grand que 1 (au sens strict), on le divise par 2 (division entière) et on affiche le résultat.

A la fin du programme, on affiche le nombre de passage dans la boucle.

> Utilisez l'instructions de boucle la plus adaptée pour cet exercice.

Exemple de résultat à obtenir :

```
Entrez un nombre entier positif au clavier : 9
nb = 4
nb = 2
nb = 1
Nombre de boucles effectuées : 3
```

Exo_3 : Liste des nombres pairs compris entre deux limites min et max

Ecrivez un script python qui affiche la liste des nombres entiers pairs compris entre deux limites lues au clavier.

Contraintes :

- Affichage sur une seule ligne, les nombres seront séparés par des espaces
- L'affichage commence par le premier nombre pair supérieur ou égal à la limite minimale et se termine par le dernier nombre pair inférieur ou égal à la limite maximale
- Si l'utilisateur n'a pas respecté l'ordre de saisie (min / max), il faudra inverser les valeurs (*pour faire ça, vous pouvez utilisez l'affectation multiple*)

Exemple de résultat à obtenir :

```
Entrez les limites de recherche :
  Limite min : 15
  Limite max : 33
Liste des nombres pairs compris entre 15 et 33 :
16 18 20 22 24 26 28 30 32
```

```
Entrez les limites de recherche :
  Limite min : 4
  Limite max : 26
```

```
Liste des nombres pairs compris entre 4 et 26 :
4 6 8 10 12 14 16 18 20 22 24 26
```

```
Entrez les limites de recherche :
  Limite min : 33
  Limite max : 15
Liste des nombres pairs compris entre 15 et 33 :
16 18 20 22 24 26 28 30 32
```

Exo_4 : Conjecture de Syracuse

En mathématiques, on appelle conjecture, une règle qui n'a jamais été prouvée. Dans cet exercice, on se propose de tester une conjecture dite "Conjecture de Syracuse" (découverte par le mathématicien allemand Lothar Collatz en 1930).

> **Principe :** on part d'un nombre entier $N_0 > 1$ et on lui applique la règle suivante :
>
> - s'il est pair, on le divise par 2 ;
> - s'il est impair, on le multiplie par 3 et on lui ajoute 1.
>
> On obtient ainsi un nombre N_1. On recommence ensuite le même calcul sur N_1 pour obtenir N_2 et ainsi de suite jusqu'à obtenir 1.
>
> Cette conjecture affirme que : pour toute valeur initiale N_0, on arrive toujours à 1.

Etape n°1 : Analyse du fonctionnement

Pour $N_0 = 5$ et $N_0 = 11$, appliquez l'algorithme décrit dans la présentation ci-dessus :

- Donnez la succession des nombres obtenus (c'est ce qu'on appelle la suite de Syracuse).
- Combien d'itérations a-t-il fallu pour arriver à 1 ?
- Quelle est la valeur maximale atteinte ?

Etape n° 2 : Afficher la suite de Syracuse

Ecrivez un script python qui calcule et affiche les valeurs constituant la suite de Syracuse d'un nombre saisi au clavier (la valeur de N_0).

Contraintes :

- L'affichage se fera sur une seule ligne.
- N'utilisez pas de `List` python

Résultat attendu :

```
Entrez un nombre entier supérieur à 1 : 3
Suite de Syracuse pour N0 = 3
```

```
3-10-5-16-8-4-2-1
```

Etape n° 3 : Afficher le temps de vol et l'altitude max

Modifiez votre script pour qu'il affiche en plus :

- le **temps de vol** : c'est-à-dire le nombre d'itérations qui ont été réalisées pour arriver à 1
- l'**altitude maximale** : c'est-à-dire la valeur maximale de la suite

Aide : *Pour trouver la valeur maximale entre plusieurs valeurs, vous pouvez utiliser la fonction de python* `max()`. *Par exemple* `max(1, 5, 3)` *retourne* `5` (documentation sur python.org ou w3schools.com)

Résultat attendu :

```
Entrez un nombre entier supérieur à 1 : 11
Suite de Syracuse pour N0 = 11
11-34-17-52-26-13-40-20-10-5-16-8-4-2-1
Temps de vol = 14
Altitude maximale = 52
```

Exo_5 : Liste de carrés et de cubes

Ecrivez un script python qui affiche les carrés et les cubes de tous les nombres entre 0 et une valeur maximale lue au clavier.

Contraintes :

- Affichage pour tous les nombres de 0 à la valeur maximale comprise
- Affichez sur une même ligne le carré et le cube (*utilisez des* `f-strings`)
- Chaque nombre sera affiché en occupant un espace de 10 caractères

Exemple de résultat à obtenir :

```
Liste des carrés et cubes des nombres.
Entre 0 et quel nombre ? 5

|    nb     |  nb**2    |  nb**3    |
|-----------|-----------|-----------|
|    0      |    0      |    0      |
|    1      |    1      |    1      |
|    2      |    4      |    8      |
|    3      |    9      |    27     |
|    4      |    16     |    64     |
|    5      |    25     |   125     |
```

Exo_6 : Recherche des divisieurs d'un nombre

Ecrivez un script python qui lit un nombre entier positif au clavier puis qui affiche tous ses diviseurs (autres que 1 et lui-même).

> Rappel : Un nombre est divisible par « div » si le reste de la division de ce nombre par « div » vaut 0.

Etape n°1 :

Exemples de résultat à obtenir :

```
Entrez un nombre entier positif : 345
3 5 15 23 69 115
```

```
Entrez un nombre entier positif : 240
2 3 4 5 6 8 10 12 15 16 20 24 30 40 48 60 80 120
```

Etape n°2 : Modifiez votre script pour ajouter ces 2 fonctionnalités :

1. Le programme ne doit s'exécuter que si le nombre lu au clavier est plus grand ou égal à 2. Si ce n'est pas le cas, on affiche un message d'avertissement
2. On affiche le nombre total de diviseurs trouvés. S'il n'y en a aucun, on affiche "Pas de diviseur"

Exemples de résultat à obtenir :

```
Entrez un nombre entier positif : 120
2 3 4 5 6 8 10 12 15 20 24 30 40 60
Le nombre 120 possède 14 diviseurs.
```

```
Entrez un nombre entier positif : 1
Saisissez un nombre > 1 SVP
```

```
Entrez un nombre entier positif : 7
Le nombre 7 n'a aucun diviseur.
```

Etape n°3 : Modifiez votre script pour ajouter ces 2 fonctionnalités :

1. Chaque diviseur sera affiché sur 6 caractères et sera séparé du suivant de 1 espace
2. On limitera l'affichage à 8 diviseurs par ligne

> Vous utiliserez des constantes pour définir le nombre de caractères (`NB_CARS`) et le nombre de diviseurs par ligne (`NB_PAR_LIGNE`)

> **Aide** pour l'affichage de K nombres par lignes : Chaque fois qu'un nouveau nombre est trouvé, vous incrémentez de 1 un compteur. Quand ce compteur vaut K (ou l'un de ses multiples), il faut passer à la ligne après avoir affiché le nombre.

Exemple de résultat à obtenir :

```
Entrez un nombre entier positif : 24000
     2      3      4      5      6      8     10     12
    15     16     20     24     25     30     32     40
    48     50     60     64     75     80     96    100
   120    125    150    160    192    200    240    250
   300    320    375    400    480    500    600    750
   800    960   1000   1200   1500   1600   2000   2400
  3000   4000   4800   6000   8000  12000
Le nombre 24000 possède 54 diviseurs.
```

Exo_7 : Recherche du PGCD (Plus Grand Diviseur Commun)

Ecrivez un script python qui recherche le plus grand diviseur commun de 2 nombres saisies au clavier.

Modifiez le travail effectué dans l'exercice précédent pour enregistrer dans 2 listes python les diviseurs des 2 nombres.