Informatique - TP 3

Utilisation des boucles `for`

M. Marmorat, M. Morel

4 Octobre 2023

Exercice 1 **Un premier test** Voici deux exemples d'utilisation de la boucle `for`. Analyser les programmes et prévoir la sortie affichée à l'écran. Les tester.

```
for k in range(1,10):
     print(k)
```

```
for lettre in "Bonjour":
    print(lettre)
```

Exercice 2 **La fonction `range`**

Q1 Écrire un programme affichant tous les nombres impairs entre 101 et 121 (inclus).

Q2 Écrire un programme affichant tous les nombres multiples de 5 de 20 à -20 (inclus !). On respectera bien ici le sens de l'affichage souhaité : on part de 20 pour aller à -20.

Exercice 3 **Les multiples de 7** Ecrire un programme qui affiche tous les multiples de 7 entre 0 et 70.

Exercice 4 **La fonction factorielle** On rappelle que la factorielle de n, notée $n!$, vaut par définition

$$n! = 1 \times 2 \times \cdots \times (n-1) \times n.$$

Q1 Que valent $5!$, $6!$ et $7!$?

Q2 Écrire une fonction `factorielle` qui calcule la factorielle de n.

Q3 Vérifiez votre fonction en l'appelant pour différentes valeurs de l'argument n.

Exercice 5 **Calcul d'une somme** Écrire une fonction `somme1` prend en argument un entier n et qui calcule

$$\sum_{k=0}^{n} \sqrt{2k+1} = \sqrt{3} + \sqrt{5} + \sqrt{7} + \cdots + \sqrt{2n+1}.$$

On vérifiera que pour $n = 100$, cette somme vaut environ 957.06.

Exercice 6 Autour des suites

Q1 Écrire une fonction `suite1` qui prend en argument un entier n et qui renvoie le n-ème terme de la suite (u_n) définie par $u_0 = 1$ et

$$\forall n \in \mathbb{N}, \quad u_{n+1} = 2u_n + 3.$$

Q2 Écrire une fonction `suite2` qui prend en argument un entier n, deux réels a et b, ainsi qu'un nombre u et qui renvoie le n-ème terme de la suite arithmético-géométrique (u_n) définie par $u_0 = u$ et

$$\forall n \in \mathbb{N}, \quad u_{n+1} = au_n + b.$$

Q3 Testez le comportement de la fonction `suite2` en calculant le 10-ème terme de :

— La suite géométrique de raison 2 et de premier terme 1.

— La suite arithmétique de raison 4 et de premier terme 7.

— La suite arithmético-géométrique définie à la question précédente.

Les résultats sont-ils cohérents ?

Exercice 7 Au service des suites récurrentes et des sommes

On considère la suite (u_n) définie par $u_0 = 1$ et

$$\forall n \in \mathbb{N}, \quad u_{n+1} = 3\sqrt{u_n} + 1.$$

Q1 Écrire une fonction `suite3` qui prend en argument un entier n et qui renvoie u_n (testez sur de petites valeurs de n).

Q2 Écrire une fonction somme, utilisant la fonction `suite3`, qui prend en argument un entier n et qui renvoie $\sum_{k=0}^{n} u_k$. A nouveau, testez sur de petites valeurs de n.

Q3 Combien de fois Python calcule-t-il u_1 pour calculer la valeur `somme(5)` ?

Q4 Écrire une fonction `somme_bis` qui prend en argument un entier n et qui renvoie $\sum_{k=0}^{n} u_k$, sans le défaut évoqué à la question précédente.

Exercice 8

Écrire une fonction qui prend en argument un mot (une chaîne de caractères) et qui renvoie le nombre de voyelles contenues dans ce mot. Testez cette fonction sur le mot « informatique ».

Exercice 9 Au hasard !

Écrire une fonction prend en argument un nombre entier N et qui renvoie la moyenne de N nombres tirés au hasard entre 0 et 100 (on pourra utiliser la fonction `randrange`).

Exercice 10 Évolution d'une population de grues du Canada

Dans cet exercice, on modélise l'évolution d'une population de grues du Canada à partir de la population initiale et d'un taux de croissance.

Q1 Écrire une fonction population prenant en argument la population initiale de grues, un taux de croissance par années, et le nombre n d'années considérées. Cette fonction renverra le nombre de grues après n années d'évolution selon le taux de croissance choisi.

Q2 Vérifier que pour une population initiale de 425 grues et une croissance de 1.94% par an, la population de grues au bout de 30 ans est de 756 individus.

Exercice 11 Spirale de Fermat des graines de tournesol

On peut modéliser l'implantation des fleurs de tournesol (c'est-à-dire des “graines” rassemblées au cœur de la fleur principale) par une spirale de Fermat. La n-ème fleur (ou graine) se situe alors à une distance d_n du centre de l'inflorescence et forme un angle θ_n avec l'horizontale avec :

$$d_n = \sqrt{n} \quad \text{et} \quad \theta_n = \pi(3-\sqrt{5})n$$

. On peut alors exprimer les coordonnées de la n-ème fleur x_n et y_n par

$$x_n = d_n \times \cos(\theta_n) \quad \text{et} \quad y_n = d_n \times \sin(\theta_n).$$

Q1 On propose le squelette de script suivant :

```python
# on importe le module de trace de graphiques et les fonctions mathematiques
import matplotlib.pyplot as plt
from math import sqrt, pi, cos, sin

N =
x = [] #x est la liste des abscisses des graines, on cree la liste vide
y = [] #y est la liste des ordonnees des graines, on cree la liste vide

for n in range(N):
    d_n =
    theta_n =
    x_n =
    y_n =
    x.append(x_n) #on ajoute x_n a la liste x
    y.append(y_n) #on ajoute y a la liste y
```

Complétez les trous dans le script ci-dessus pour calculer les coordonnées de 1000 fleurs de tournesol.

Q2 Complétez le script précédent avec les deux instructions suivantes

```python
plt.plot(x,y, "oy", markeredgecolor='gray')
plt.show()
```

pour obtenir la figure suivante

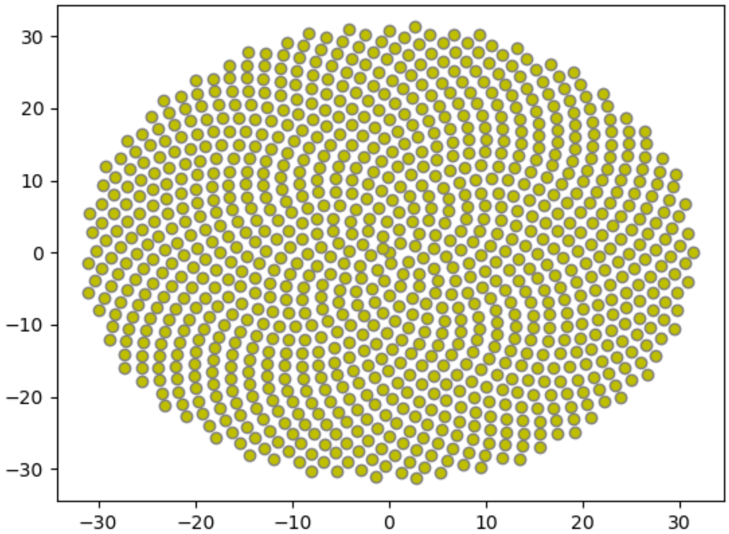

Figure 1 – Une spirale de Fermat à 1000 points !