



Manipuler les listes et les chaînes de caractères

Utilisation des boucles

Méthode de travail. Pour chaque exercice :

1. prévoir le résultat sans lancer Python ;
2. **créer un nouveau script** ;
3. exécuter le code et relever le résultat exact ;
4. expliquer les éventuels écarts et ne pas hésiter à m'appeler ;
5. conserver les notes et les codes validés pour plus tard.

1 | Quelques boucles for

Question 1 : Déterminer ce que fait le programme suivant.

```
for i in range(1,10):  
    print(i**2)
```

Réponse 1 : Il affiche dans la console les carrés des nombres allant de 1 à 9.

Question 2 : Écrire un programme qui affiche la table de 7 via une boucle for. Faire en sorte que l'affichage soit par exemple 7 fois 0 font 0.

Réponse 2 : La fonction `print()` peut prendre plusieurs arguments et les affiche les uns après les autres sur la même ligne.

```
for i in range(11):  
    print("7 fois ", i, " font ", 7*i)
```

Question 3 : Modifier ce programme de manière à ce qu'il demande à l'utilisateur un entier n , et affiche la table de n .

Réponse 3 :

```
n = int(input("Entrer un nombre pour avoir sa table : "))  
# on pense à la conversion chaîne de caractère vers entier  
for i in range(11):  
    print(n, " fois ", i, " font ", n*i)
```

Question 4 : Écrire un programme qui demande à l'utilisateur d'entrer son nom, puis l'épelle ligne à ligne.

Réponse 4 :

```
nom = input("Entrer un nom : ")  
for char in nom:  
    print(char)
```

2 | Quelques boucles while

Question 5 : Déterminer ce que fait le programme suivant.

```
n = 1  
while n**2 < 1000:  
    n += 1  
print(n)
```

Réponse 5 : Ce programme affiche dans la console l'entier le plus petit dont le carré est supérieur à 1000.

Question 6 : Écrire un programme qui affiche la première valeur de $n \in \mathbb{N}$ pour laquelle $n^2 + 3n + 1 > 165$.

Réponse 6 :

```
n = 0
while n**2+3*n+1 < 165:
    n += 1
print(n)
```

Question 7 : Écrire un programme qui demande à l'utilisateur d'entrer un entier n au clavier, et qui réitère le processus tant que l'entier entré par l'utilisateur est inférieur à 1000.

Réponse 7 :

```
n = int(input("Entrer un entier : "))
while n < 1000:
    n = int(input("C'est tout ? Tu peux faire mieux : "))
print("Voilà quand tu veux !")
```

3 | Indices et éléments

Question 8 : Comparer le résultat des deux codes suivants.

```
L = [1,2,3,4,5]
for i in range(len(L)) :
    print(L[i]**2)
```

```
L = [1,2,3,4,5]
for j in L :
    print(j**2)
```

Réponse 8 : Les résultats sont identiques. Dans les boucles, soit on utilise les indices, soit directement les éléments.

Question 9 : Pour chacun des exemples suivants, écrire le code en remplaçant une boucle sur les éléments en son équivalent sur les indices, et vice-versa.

```
L = [1,2,3,4,5]
for l in L :
    if l > 3:
        print(1)
```

```
S = "abcdef"
for i in range(len(S)) :
    print(S[i]+'|')
```

Réponse 9 : Pour le premier cas :

```
L = [1,2,3,4,5]
for i in range(len(L)) :
    if L[i] > 3:
        print(L[i])
```

Et :

```
S = "abcdef"
for char in S :
    print(char+'|')
```

Bien qu'il soit possible d'utiliser les indices ou les éléments dans une boucle, pour un problème particulier, une des méthodes peut être à privilégier.

Question 10 : Choisir dans les cas suivants la méthode la plus adaptée à la tâche demandée.

- Écrire dans la console le premier élément d'une liste L d'entier donnée qui dépasse 2. *On peut par exemple utiliser la liste L=[1, 1, 0, -4, 12] qui devrait afficher 12.*
- Écrire dans la console l'indice du premier élément d'une liste L d'entier donnée qui dépasse 2. *On peut par exemple utiliser la liste L=[1, 1, 0, -4, 12] qui devrait afficher 4.*

4 | Extraction d'une liste

Cet exercice a pour vocation de se familiariser avec l'extraction à partir d'une liste. Dans un premier temps, définir une liste L = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].

Question 11 : Prévoir le résultat des commandes suivantes, puis vérifier sous Python.

Commande	Résultat prévu
<code>print(L[3])</code>	
<code>print(L[2:5])</code>	
<code>print(L[2:8:2])</code>	
<code>print(L[-2:-5])</code>	
<code>print(L[-2:-5:-2])</code>	
<code>print(L[-5:-2:-2])</code>	
<code>print(L[5:])</code>	
<code>print(L[:-4])</code>	
<code>print(L[::-1])</code>	
<code>print(L[3::3])</code>	

Réponse 11 :

Commande	Résultat
<code>print(L[3])</code>	4
<code>print(L[2:5])</code>	[3, 4, 5]
<code>print(L[2:8:2])</code>	[3, 5, 7]
<code>print(L[-2:-5])</code>	[]
<code>print(L[-2:-5:-2])</code>	[9, 7]
<code>print(L[-5:-2:-2])</code>	[]
<code>print(L[5:])</code>	[6, 7, 8, 9, 10]
<code>print(L[:-4])</code>	[1, 2, 3, 4, 5, 6]
<code>print(L[::-1])</code>	[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
<code>print(L[3::3])</code>	[4, 7, 10]

Question 12 : Déterminer les commandes permettant d'obtenir les objets suivants.

Objet attendu	Commande
9	
[5, 6, 7, 8]	
[3, 5, 7]	
[10, 9, 8]	
[6, 5, 4]	
[8, 6, 4, 2]	

Réponse 12 :

Objet attendu	Une commande possible
9	L[8]
[5, 6, 7, 8]	L[4:8]
[3, 5, 7]	L[2:7:2]
[10, 9, 8]	L[9:6:-1]
[6, 5, 4]	L[5:2:-1]
[8, 6, 4, 2]	L[7::-2]

5 | Palindrome

Un palindrome est un mot, ou une phrase, s'écrivant de la même manière de gauche à droite ou de droite à gauche. Par exemple, bob est un palindrome.

Dans un premier temps, définir une variable `maStr` de type `str` qui contient "abcdef".

On rappelle que les `str` sont immuables.

Question 13 : Déterminer une commande d'extraction permettant d'écrire `maStr` à l'envers (de l'inverse). Réfléchir notamment au pas.

Réponse 13 :

```
maStr = "abcdef"
maRts = maStr[::-1]
```

Question 14 : Faire de même avec une boucle.

Réponse 14 :

```
maStr = "abcdef"
maRts = ''
for i in range(len(maStr)-1, -1, -1):
    maRts = maRts + maStr[i]
```

ou aussi pour la boucle :

```
for i in range(len(maStr)):
    maRts = maStr[i] + maRts
```

Question 15 : En déduire un programme qui demande à l'utilisateur d'entrer une chaîne de caractères, et qui affiche `True` si cette chaîne est un palindrome, et `False` sinon. Faire deux versions, l'une avec une condition et l'autre sans utiliser la structure conditionnelle `if`.

Réponse 15 :

```
maStr = input("Entrer une chaine de caractere : ")
maRts = maStr[::-1]
if maStr == maRts:
    print(True)
else :
    print(False)
```

Ou aussi sans condition :

```
maStr = input("Entrer une chaine de caractere : ")
maRts = maStr[::-1]
rep = maStr == maRts
print(rep)
```

Question 16 : Tester le programme sur les mots suivants (sans accent ni majuscule) : radar, rotor, baobab, coloc, ananas, 1221.

Réponse 16 : On obtient : True, True, False, True, False et True *en effet, 1221 est une chaine de caractère (retour de la fonction `input()`) et est bien un palydrome.*

6 | Décimale de π

On cherche à accéder à la n -ième décimale de π sous Python. Cette constante est stockée dans la variable `pi` de la librairie `math`.

Question 17 : En utilisant des conversions, réaliser un programme qui demande à l'utilisateur la valeur de n puis affiche dans la console la n -ième décimale de π (au sens commun de la numérotation : $n = 2$ doit écrire 4).

Réponse 17 :

```
from math import pi

valeur_n = int(input("Numéro de la décimale : ")) # on pense à convertir str vers int
pi_str = str(pi) # conversion de pi en chaine de caractere
print(pi_str[valeur_n+1]) # prise en compte de 3. et de la numerotation a 0
```