Des affichages à l'aide de boucles

1. Tester le code suivant :

```
print("Lundi",1)
print("Mardi",2)
print("Mercredi",3)
```

2. Écrire un code qui définit **n=5** puis qui affiche :

```
Ligne 1
Ligne 2
Ligne 3
Ligne 4
Ligne 5
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

3. Écrire un code qui définit **n=5** puis qui affiche :

```
Ligne 5
Ligne 4
Ligne 3
Ligne 2
Ligne 1
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

4. Tester le code suivant :

```
print("Lundi",1,end="")
print("Mardi",2,end="")
print("Mercredi",3)
```

En déduire l'intérêt de l'instruction **end=""**.
L'instruction **print()** sans paramètre effectue simplement un retour à la ligne.

5. Écrire un code qui définit **n=5** puis qui affiche :

```
1 2 3 4 5
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

6. Écrire un code qui définit **n=5** puis qui affiche :

```
*****
*****
*****
*****
*****
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

7. Écrire un code qui définit **n=5** puis qui affiche :

```
*
**
***
****
*****
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

8. Écrire un code qui définit **n=5** puis qui affiche :

```
*****
****
***
**
*
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

9. Écrire un code qui définit **n=5** puis qui affiche :

```
*****
 ****
  ***
   **
    *
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

10. Écrire un code qui définit **n=5** puis qui affiche :

```
*****
*   *
*   *
*   *
*****
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

11. Écrire un code qui définit **n=5** puis qui affiche :

```
*-*-*
-*-*-
*-*-*
-*-*-
*-*-*
```

Votre code devra fonctionner pour n'importe quelle valeur de n et vous le testerez ensuite avec $n = 10$ (par exemple).

Retour sur les boucles *while*

12. Écrire du code qui part d'une valeur **x=2500** (par exemple) puis qui diminue à étape par étape cette valeur de 45 à chaque fois jusqu'à aboutir à une valeur strictement négative. Le code affichera la valeur de **x** à chaque étape, la dernière valeur affichée étant devant être positive. Noter la dernière valeur affichée.

Réponse : _____

13. Écrire du code qui part d'une valeur **x=2500** (par exemple) puis qui diminue à étape par étape cette valeur de 10% à chaque fois jusqu'à aboutir à une valeur strictement inférieure à 10. Le code affichera la valeur de **x** à chaque étape, la dernière valeur affichée étant devant être la valeur strictement inférieure à 10. Noter la dernière valeur affichée.

Réponse : _____

14. Écrire du code qui part d'une valeur initiale `x=25` puis qui à chaque étape double la valeur de `x` jusqu'à dépasser 10000. Faire afficher la dernière valeur de `x` obtenue (celle qui dépasse 10000) ainsi que le nombre d'étapes effectuées.

Réponse : _____

15.

16. On considère un produit présent dans une solution avec une concentration initiale $C = 24$ (par exemple). À chaque étape, la concentration du produit double. On dit que l'on est en zone verte lorsque $C < 50$, en zone orange lorsque $50 \leqslant C < 100$ et en zone rouge lorsque $C \geqslant 100$. Écrire du code qui réalise les différentes étapes et affiche à chaque fois la couleur de la zone ainsi que la valeur de C. Le dernier affichage devra correspondre au moment où on se trouve pour la première fois en zone rouge. Par exemple, pour $C = 24$, on devra obtenir l'affichage :

```
Zone verte 24
Zone verte 48
Zone orange 96
Zone rouge 192
```

17. Exécuter votre code avec $C = 3$ et noter la valeur de C une fois en zone rouge.

Réponse : _____

Retour sur les fonctions

18. Écrire une fonction `somme_cubes(n)` qui calcule et renvoie la somme des cubes des nombres compris entre 1 et n au sens large (autrement dit la somme $1^3 + 2^3 + \cdots + n^3$). Noter la valeur obtenue avec `somme(50)`.

Réponse : _____

Corrections

Q 2.

```
n = 5
for k in range(1,n+1):
    print("Ligne",k)
```

```
Ligne 1
Ligne 2
Ligne 3
Ligne 4
Ligne 5
```

Q 3. Une méthode :

```
n = 5
for k in range(0,n):
    print("Ligne",n-k)
```

```
Ligne 5
Ligne 4
Ligne 3
Ligne 2
Ligne 1
```

Une autre méthode :

```
n = 5
k = 5
while k>=1:
    print("Ligne",k)
    k = k-1
```

```
Ligne 5
Ligne 4
Ligne 3
Ligne 2
Ligne 1
```

Q 5.

```
n = 5
for k in range(1,n+1):
    print(k," ",end="")
```

```
1 2 3 4 5
```

Q 6. Une méthode :

```
n = 5
for k in range(0,n):
    for i in range(0,n):
        print("*",end="")
    print()
```

```
*****
*****
*****
*****
*****
```

Une autre méthode :

```
n = 5
for k in range(0,n):
    print(n*"*")
```

```
*****
*****
*****
*****
*****
```

Q 7. Une méthode :

```
n = 5
for k in range(1,n+1):
    for i in range(0,k):
        print("*",end="")
    print()
```

```
*
**
***
****
*****
```

Une autre méthode :

```
n = 5
for k in range(1,n+1):
    print(k*"*")
```

```
*
**
***
****
*****
```

Q 8. Une méthode :

```
n = 5
for k in range(0,n):
    for i in range(0,n-k):
        print("*",end="")
    print()
```

```
*****
****
```

```
***
**
*
```

Une autre méthode :

```
n = 5
for k in range(0,n):
    print((n-k)*"*")
```

```
*****
****
***
**
*
```

Q 9. Une méthode :

```
n = 5
for k in range(0,n):
    print(k*" "+(n-k)*"*")
```

```
*****
****
***
**
*
```

Une autre méthode :

```
n = 5
for k in range(0,n):
    for i in range(0,k):
        print(" ",end="")
    for i in range(0,n-k):
        print("*",end="")
    print()
```

```
*****
****
***
**
*
```

Q 10.

```
n = 5
for k in range(0,n):
    if k==0 or k==n-1:
        for i in range(0,n):
            print("*",end="")
    else:
        print("*",end="")
        for i in range(0,n-2):
            print("+",end="")
        print("*",end="")
    print()
```

```
*****
*+++*
*+++*
*+++*
*****
```

J'ai mis **+** car les espaces ne sortent pas bien sur mon document.

Q 12.

```
x = 2500
while x>=0:
    print(x)
    x = x-45
```

```
2500
2455
2410
2365
2320
2275
2230
2185
2140
2095
2050
2005
1960
1915
1870
1825
1780
1735
1690
1645
1600
1555
1510
1465
```

```
1420
1375
1330
1285
1240
1195
1150
1105
1060
1015
970
925
880
835
790
745
700
655
610
565
520
475
430
385
340
295
250
205
160
115
70
25
```

Q 13.

```
x = 2500
while x>=10:
    print(x)
    x = x*90/100
print(x)
```

```
2500
2250.0
2025.0
1822.5
1640.25
1476.225
1328.6025
1195.74225
1076.168025
```

```
968.5512225
871.69610025
784.526490225
706.0738412025
635.46645708225
571.919811374025
514.7278302366226
463.25504721296033
416.92954249166434
375.2365882424979
337.7129294182481
303.9416364764233
273.54747282878094
246.19272554590282
221.57345299131256
199.41610769218133
179.4744969229632
161.52704723066688
145.3743425076002
130.8369082568402
117.75321743115617
105.97789568804055
95.3801061192365
85.84209550731285
77.25788595658156
69.5320973609234
62.57888762483106
56.32099886234795
50.68889897611315
45.62000907850184
41.058008170651654
36.95220735358649
33.25698661822784
29.931287956405058
26.938159160764553
24.244343244688096
21.819908920219287
19.637918028197358
17.674126225377623
15.906713602839861
14.316042242555875
12.884438018300289
11.59599421647026
10.436394794823235
9.392755315340912
```

Q 14. J'interprète « x dépasse 10000 » comme $x > 10000$ et je considère que le nombre d'étapes effectuées est le nombre de fois où on a multiplié x par 2 (noté k dans le code).

```
x = 25
k = 0
while x<=10000:
    x = x*2
    k = k+1
print(x,k)
```

```
12800 9
```

Q 16.

```
C = 24
while C<100:
    if C<50:
        print("zone verte",C)
    else:
        print("zone orange",C)
    C = C*2
print("zone rouge",C)
```

```
zone verte 24
zone verte 48
zone orange 96
zone rouge 192
```

Q 18.

```
def somme_cubes(n):
    s = 0
    for i in range(1,n+1):
        s = s+i**3
    return s
print(somme_cubes(50))
```

```
1625625
```