```python
# -*- coding: utf-8 -*-
"""
Éditeur de Spyder
"""

############ I - Créer des fonctions ########

#### Q1 #####

import math
import cmath
def trinome(a,b,c):
    """Calcul des racines d'un polynôme"""
    delta=b**2-4*a*c
    if delta>0:
        x1=(-b-math.sqrt(delta))/(2*a)
        x2=(-b+math.sqrt(delta))/(2*a)
        print(x1,' et ',x2,' sont les racines réelles du pol
    elif delta==0:
        x1=-b/(2*a)
        print(x1)
    else:
        x1=(-b-1j*math.sqrt(-delta))/(2*a)
        x2=(-b+1j*math.sqrt(-delta))/(2*a)
        print(x1,' et ',x2,' sont les racines complexes du p

trinome(1,2,4)
trinome(1,-2,1)

#### Q2 #####

def entiers(n):
    liste=[]
    for k in range(n+1):
        liste.append(k)
    return liste

print(entiers(15))

#### Q3 #####

def pairs(n):
    liste=[]
```

```
    for k in range(n+1):
        if k%2==0:
            liste.append(k)
    return liste

print(pairs(15))

#### Q4 #####
def max2(x,y):
    if x>y:
        return x
    else:
        return y

print(max2(27,150))

#### Q5 #####

def bissextile(annee):
    if annee%400==0 or (annee%4==0 and annee%100!=0) :
        return True
    else:
        return False

print(bissextile(2004))
print(bissextile(2106))
print(bissextile(2300))
print(bissextile(2400))

#### Q6 #####

def somme(n):
    sum=0
    for k in range(n+1):
        sum=sum+k
    return sum

print(somme(10),somme(50))

#### Q7 #####

def sommeCube(n):
    sum=0
```

```
    for k in range(n+1):
        sum=sum+k**3
    return sum

print(sommeCube(10),sommeCube(3),somme(3)*somme(3))

#### Q8 #####

import math
def suite(n):
    un=0.5
    for k in range(n+1):
        un=math.sqrt(un)
    return un

print(suite(0),suite(1),suite(2),suite(5),suite(10))

#### Q9 #####

liste=[1,4,7,9,11,255]

def somme_Liste(L):
    sum=0
    for k in range(len(L)):
        sum=sum+L[k]
    return sum

print(somme_Liste(liste))

#### Q10 #####

def rectangle(L,H):
    for i in range(H):
        if i==0 or i==H-1:
                print('+',(L-2)*'-','+')
        else:
                print('|',(L-2)*' ','|')

rectangle(25,9)

#### Q11 #####

def asterix(chaine):
```

```
    chainex=chaine[0] + '*'
    for i in range(1,len(chaine)-1):
        chainex=chainex+chaine[i]+'*'

    chainex=chainex+chaine[len(chaine)-1]

    return chainex

print(asterix('etoile'))

#### Q12 #####

def inverse(nom):
    mon=''
    for i in range(len(nom)):
        mon=mon+nom[len(nom)-i-1]
    return mon

def palind(chaine):
    chinv=inverse(chaine)
    if chinv==chaine:
        print('Palindrome !')
    else:
        print('Ce n\'est pas un palindrome')

palind('papa')
palind('baab')

############ II - CARRE MAGIQUE #############


import time

def somme_ligne(m,i):
    sl=0
    for k in range(len(m[0])):
        sl=sl+m[i][k]
    return sl

mat=[[2,11,8],[13,7,1],[6,3,12]]
mat2=[[2,1,1],[1,1,1],[1,1,1]]
mat3=[[2,5,5],[7,4,1],[3,3,6]]
mat4=[[8,1,6],[3,5,7],[4,9,2]]
```

```
#print(mat)
#print(somme_ligne(mat,2))

def somme_colonne(m,i):
    sc=0
    for k in range(len(m[0])):
        sc=sc+m[k][i]
    return sc

#print(somme_colonne(mat,2))

def somme_diag(m,j):
    sd=0
    if j==0:
        for k in range(len(m[0])):
            sd=sd+m[k][k]
    else:
        for k in range(len(m[0])):
            sd=sd+m[k][len(m[0])-1-k]
    return sd

#print(somme_diag(mat,0))
#print(somme_diag(mat,1))

def affiche_carre(m):
    print(' _____________________',end='\n')
    for i in range(len(m)):
        print(' | ',m[i][0],' | ', m[i][1],' | ', m[i][2],'
        print(' _____________________',end='\n')

def est_magique(m):
    test=somme_ligne(m,0)
    bool=True
    for k in range(3):
        if test!=somme_ligne(m,k) or test!=somme_colonne(m,k
            bool=False
    return bool
#print(est_magique(mat2))

def tous_different(m):
    liste=[]
    for l in range(3):##Création de la liste
        for c in range(3):
```

```
            liste.append(m[l][c])
    j,k=0,1
    bool=True
    while j<8 and bool==True: ##création de boucle pour test
        while k<9 and bool==True :
            if liste[j]==liste[k]:
                bool=False
            k=k+1
        j=j+1
        k=j+1

    return bool

n=10

print(tous_different(mat4),est_magique(mat4))


#compteur=0
#tempsDebut=time.clock()
#for a in range(1,n):  ##méthode bourrin
#    for b in range(1,n):
#        for c in range(1,n):
#            for d in range(1,n):
#                for e in range(1,n):
#                    for f in range(1,n):
#                        for g in range(1,n):
#                            for h in range(1,n):
#                                for i in range(1,n):
#                                    m=[[a,b,c],[d,e,f],[g,h
#                                    if est_magique(m)==True
#                                        compteur=compteur+1
#                                        #affiche_carre(m)
#                                        duree=time.clock()-t
#                                        print(duree)
#print(compteur)



tempsDebut=time.clock()
compteur=0
for a in range(1,n):  ##méthode moins bourrin
```

```
    for b in range(1,n):
        for c in range(1,n):
            for d in range(1,n):
                for e in range(1,n):
                    s=a+b+c
                    if s>d+e and s>b+e and s>a+e and s>c+e:
                        f=s-d-e
                        g=s-a-d
                        h=s-b-e
                        i=s-c-f
                        if n>f>0 and n>g>0 and n>h>0 and n>i
                            m=[[a,b,c],[d,e,f],[g,h,i]]
                            if est_magique(m)==True and tous
                               compteur=compteur+1
                               affiche_carre(m)
                               duree=time.clock()-tempsDebut
                               print(duree)
print(compteur)

#matrice=[[a,b,c],[d,e,f],[g,h,i]]
```