```
## TP 9

## exo 1
# q1
def somme(L):
    s = 0
    for k in range(len(L)):
        s = s+(k+1)*L[k]
    return s

L = [2*k+1 for k in range(51)]
print(somme(L))

# q2
def positiver(L):
    for k in range(len(L)):
        if L[k]<0:
            L[k]=0
    return L

# q3
def impair(L):
    return L[1:len(L):2]

## exo 2
# q1
# C1 = C0 C0 = 1
# C2 = C0 C1 + C1 C0 = 2
# C3 = C0 C2 + C1 C1 + C2 C0 = 5
# C4 = C0 C3 + C1 C2 + C2 C1 + C3 C0 = 14

# q2
def nextcat(L):
    S = 0
    n = len(L)
    for k in range(n):
        S = S + L[k]*L[n-1-k]
    return S

# q3
def cat(n):
    C = 1
    L = [C]
    for k in range(n-1):
        C = nextcat(L)
        L.append(C)
    return L

# les nombres des Catalan interviennent dans différents problèmes de combinatoire.
# par exemple, le n-ème de Catalan est égal au nombre de façons de placer des
parenthèses
# dans une expression contenant n+1 caractères
# les premiers nombres de Catalan sont 1, 1, 2, 5, 14, 42, 132, 429

## exo 3
# q1
# on constate que la liste [x0, x1, x2, ..., x(n-1)] correspond au nombre
# x0*10**(n-1) + x1*10**(n-2) + ... + x(n-2)*10**1 + x(n-1)*10**0
def list_to_number(L):
    S = 0
    n = len(L)
    for k in range(n):
        S = S+L[k]*10**(n-1-k)
    return S
```

```python
# q2
def dernier(n):
    r = n%97
    return 97-r

# q3
def number_to_list(m):
    a = m%10 # chiffre des unités
    b = m//10 # chiffre des dizaines
    return [b,a]

# q4
def complete(L):
    n = list_to_number(L)
    m = dernier(n)
    M = number_to_list(m)
    return L + M

## exo 4
# q1
import random as rd
def liste_alea(n):
    L = [k for k in range(10)]
    return [rd.choice(L) for k in range(n)]

# q2
def bonne_place(L,M):
    compt = 0
    for k in range(len(L)):
        if L[k]==M[k]:
            compt += 1
    return compt

# q3
def distrib(L):
    M = 10*[0] # M est initialisé avec des 0 partout
    for x in L: # pour chaque élément x de L
        M[x] = M[x]+1 # on compte 1 de plus dans le nombre de x, c'est-à-dire dans
M[x]
    return M

# q4
def elem_commun(L,M):
    DL = distrib(L)
    DM = distrib(M)
    S = 0
    for k in range(10):
        S = S+min(DL[k],DM[k])
    return S

# q5
mystere = liste_alea(5)

def jeu(L):
    a = bonne_place(L,mystere)
    tot = elem_commun(L,mystere)
    b = tot - a
    return a,b
```