DS 2 : lundi 8 décembre

Correction

Exercice 1 *(SQL : d'après le début de CCMP MP-PC-PSI, 2021).*

Correction : Rajout de la colonne `der_rid` et de la question 3 (pour avoir de la jointure).

1°
```
SELECT COUNT()
FROM Participant
WHERE ne >= 1999 AND ne <= 2003
```

2°
```
SELECT diff,AVG(duree)
FROM Rando
GROUP BY diff
```

3°
```
SELECT  pnom, rnom
FROM Rando R
   JOIN Participant P ON R.rid=P.der_rid
```

4°
```
SELECT pnom
FROM Participant
WHERE diff_max < (SELECT diff FROM Rando
                  WHERE rid=42)
```

5°
```
SELECT DISTINCT rid
FROM Rando
WHERE rnom IN (SELECT rnom FROM rando
               GROUP BY rnom
               HAVING COUNT() > 1)
```

Exercice 2 *(suite de Fibonacci et mémoïsation).*

Correction :

```
fibo_dic={}
def fibo(n):
    if n not in fibo_dic:
        if n==0 or n==1:
            b=n
        else:
            b=fibo(n-1)+fibo(n-2)
        fibo_dic[n]=b
    return fibo_dic[n]
```

Exercice 3 *(somme maximale de termes consécutif d'une suite).*

Correction :

1° (a)
```
suiteexemple=[5,-2,-6,4,3,-2,8,-2,1,-5]
    def somme_max_1(suite):
        n = len(suite)
```

```
    maxi = suite[0] # c'est s_{0,0}
    for i in range(n):
        for j in range(i,n):
            somme = 0
            for k in range(i,j+1):
                somme = somme + suite[k]
            if somme>maxi:
                maxi=somme
    return maxi

print(somme_max_1(suiteexemple))
```

(b) On imbrique trois boucles, l'instruction où on augmente `somme` est exécutée A fois, où $A = \sum_{i=0}^{n-1}\sum_{j=i}^{n-1}\sum_{k=i}^{j} 1$, ainsi (réindexation $j' = j-i+1$ puis $i' = (n-i+1)$) : $A = \sum_{i=0}^{n-1}\sum_{j=i}^{n-1}(j-i+1) = \sum_{i=0}^{n-1}\sum_{j'=1}^{n-i} j' = \sum_{i=0}^{n-1}\frac{(n-i)(n-i+1)}{2} = \sum_{i'=1}^{n}\frac{i'(i'+1)}{2} = \frac{1}{2}\left(\frac{n(n+1)(2n+1)}{6}+\frac{n(n+1)}{2}\right) = O(n^3)$.

(c) On répète plusieurs fois le même calcul, en effet dans la boucle calculant $S_{i,n}$ on a en fait calculé $S_{i,j}$ pour tout j, on peut donc s'épargner une boucle et ainsi passer en $O(n^2)$.

```
def somme_max_2(suite):
    n = len(suite)
    maxi = suite[0] # c'est s_{0,0}
    for i in range(n):
        somme = 0
        for j in range(i,n):
            somme = somme + suite[j]
            if somme>maxi:
                maxi=somme
    return maxi
print(somme_max_2(suiteexemple))
```

2° (a) Ce n'est rien d'autre que $\max(M_1,\dots,M_n)$

(b) Comme l'unique somme d'éléments consécutif qui se termine par a_1 est a_1, on a $M_1 = a_1$, pour la formule de récurrence on a : $M_{i+1} = \max_{d\in[\![1,i+1]\!]}\left(\sum_{k=d}^{i+1} a_k\right) = \max(\max_{d\in[\![1,i]\!]}\left(\sum_{k=d}^{i+1} a_k\right), a_{i+1}) = \max(\max_{d\in[\![1,i]\!]}\left(\sum_{k=d}^{i} a_k\right) + a_{i+1}, a_{i+1}) = \max(M_i + a_{i+1}, a_{i+1})$. Il ne reste plus qu'à remarquer que $M_i + a_{i+1} \geq a_{i+1}$ si et seulement si $M_i \geq 0$ pour conclure.

(c)
```
def somme_max_3(suite):
    n = len(suite)
    M=[0]*n
    M[0]=suite[0]
    for i in range(1,n):
        if M[i-1]>=0:
            M[i]=M[i-1]+suite[i]
        else:
            M[i]=suite[i]
    maxi=M[0]
    for i in range(1,n):
        if M[i]>maxi:
            maxi=M[i]
    return maxi
print(somme_max_3(suiteexemple))
```