

Devoir surveillé n°1 - Corrigé

Exercice 1 : 1. Somme de deux matrices parcimonieuses :

```
1 def somme(M,N):
2     """ ajoute deux matrices parcimonieuses codées sous forme de dictionnaire """
3     S={'dim':M['dim']} #le dictionnaire où sera stockée la matrice somme
4     for clef in M:
5         if clef!='dim':
6             if clef in N:
7                 somme=M[clef]+N[clef]
8                 if somme!=0: #si le coefficient est nul on ne le stocke pas
9                     S[clef]=somme
10            else:
11                S[clef]=M[clef]
12    for clef in N:
13        if clef!='dim':
14            if clef not in M:
15                S[clef]=N[clef]
16    return S
```

2. Pour calculer la complexité, nous devons remarquer deux choses :

- la première boucle "for clef in M" va tourner $(c_1 + 1)$ fois (le nombre de clefs dans M), et la deuxième boucle va tourner $(c_2 + 1)$ fois
- le test "if clef in N" va se réaliser **en temps constant** (particularité des dictionnaires : la recherche d'une clef dans un dictionnaire se fait en temps constant), de même que le test "if clef not in M".

La complexité de cette fonction est : $C(c_1, c_2) = 1 + (1 + c_1)(1 + 1 + 2) + (1 + c_2)(1 + 1 + 1) = 4c_1 + 3c_2 + 8$,

donc la complexité est en $O(c_1 + c_2)$.

Exercice 2 : 1. id_titre est un entier (type int) égal à 49987654 à l'issue des instructions

zones est une liste de deux entiers (type list) égale à [1,3]

date_fin est une liste de trois entiers (type list) égale à [2025,8,31]

2. Voici le code :

```
1 passages=[] #passage sera une liste de listes
2 for i in range(2, len(lignes)): #on parcourt la fin des lignes du fichier
3     donnees_passage=lignes[i].rstrip('\n').split(',') #on retire le retour
4     #chariot et on sépare selon les virgules
5     date=donnees_passage[0].split('-') #on sépare ensuite la date selon
6     #les tirets
7     heure=donnees_passage[1].split(':') #on sépare l'heure selon les
8     #deux points
9     passages.append([int(date[0]), int(date[1]), int(date[2]), int(heure[0]),
10    int(heure[1]), int(heure[2]), int(donnees_passage[2])]) #on rajoute la
11    #liste demandée en convertissant toutes les données de chaînes
12    #de caractères à entiers.
```

3. date1 est antérieur à date2 si son année est strictement antérieure, ou si les années sont identiques et le mois est strictement antérieur, ou si les années et mois sont identiques et le jour est antérieur.

```
1 def estAvant(date1,date2):
2     return (date1[0]<date2[0]) or (date1[0]==date2[0] and date1[1]<date2[1])
3     or (date1[0]==date2[0] and date1[1]==date2[1] and date1[2]<=date2[2])
```

4. Voici le code :

```
1 def nbSecondesEntre(heure1,heure2):
2     return 3600*(heure1[0]-heure2[0])+60*(heure1[1]-heure2[1])+heure1[2]-heure2[2]
```

5. On passe en arguments toutes les données définies dans l'énoncé, les données du lecteur avec des majuscules et les données du titre avec des minuscules.

```
1 def testPassage(Zone,Id_point, Liste_noire, Maintenant, id_titre, zones, date_fin, passages):
2     for x in Liste_noire: #on vérifie si l'id du titre est dans la liste noire
3         if x==id_titre:
4             print('Titre refusé')
5             return False
6     if Zone<zones[0] or Zone>zones[1]: #on vérifie si la Zone dépasse les zones autorisées
7         print("Non valide dans cette zone")
8         return False
9     if estAvant(date_fin, Maintenant[:3]): #on vérifie si la date
10    #d'expiration est avant ou après la date actuelle
11        print("Titre expiré")
12        return False
13    for x in passages: #on regarde tous les passages de la carte
14        heure_passage=x[3:6]
15        point=x[6]
16        if (point==Id_point and Maintenant[:3]==x[:3] and
17            nbSecondesEntre(Maintenant[3:],heure_passage)<450): #on vérifie
18            #qu'il n'y a pas eu de passage trop récent au même point.
19                print("Titre déjà validé")
20                return False
21    return True #on renvoie True si le titre a passé tous les tests
```

Remarque : nous avons éludé le cas où un passage récent se serait fait au même point, mais le jour précédent un peu avant minuit, de sorte que l'on pourrait passer moins de 450 secondes plus tard mais le lendemain...

Exercice 3 : 1. Obtention du numéro correspondant à un couple (x,y) :

```
1 def numero(x,y):
2     if x==0 and y==0:
3         return 0
4     if y==0: #dans ce cas le couple (x,y) est sur l'axe des abscisses
5         return numero(0,x-1)+1
6     return numero(x+1,y-1)+1
```

2. Obtention du couple correspondant à un numéro n :

```
1 def couple(n):
2     if n==0:
3         return (0,0)
4     x,y=couple(n-1) #on regarde le couple correspondant au numéro précédent
5     if x==0: #dans ce cas le couple précédent est sur l'axe des ordonnées
6         return (y+1,0)
7     else:
8         return (x-1,y+1)
```