DEVOIR SURVEILLÉ N° 1

« Aucun document autorisé »

Matière : Informatique **Filière : MP2**
Professeur : A. ZBAKH **Durée : 2h**

Remarque :

Si au cours du DS, un candidat repère ce qui peut lui sembler être une erreur d'énoncé, il le signale sur sa copie et poursuit sa composition en expliquant les raisons des initiatives qu'il est amené à prendre.

L'épreuve se compose de **04** exercices indépendants

Rappel sur les chaines de caractères :

Les chaînes de caractères sont représentées en Python par le type **str**. Pour affecter une chaine de caractères à une variable, il suffit d'écrire S='BONJOUR'. Cette chaine peut, en fait, être considérée comme un tableau de caractères (non modifiable), où chacune de ses cases contient un seul caractère. Ainsi, elle peut être schématisée comme suit :

B	O	N	J	O	U	R

Ceci implique que le premier caractère de S est exprimée par S[0], le deuxième par S[1] et le dernier par S[len(S)-1] où **len** est une fonction prédéfinie qui calcule la langueur de la chaine passée en paramètre.

Les opérateurs de comparaison '==', ' !=' , ' <' , ' >', '<=' et '>=' peuvent être également appliqués sur les chaines de caractères. Ainsi les expressions '06'>='00', '09'>'08', '05'<'08' et '0654342310'==' 0654342310' valent True.

N.B : Il est à noter que les fonctions demandées dans les exercices peuvent être utilisées par la suite même si vous n'avez pas pu les définir.

Exercice 1 :

Soit la fonction tri suivante :

```
1  def tri(L):
2      n = len(L)
3      for i in range(1, n):
4          j = i
5          x = L[i]
6          while 0 < j and x < L[j-1]:
7              L[j] = L[j-1]
8              j = j-1
9          L[j] = x
```

1) Lors de l'appel tri(L) lorsque L est la liste [5, 2, 3, 1, 4], donner le contenu de la liste L à la fin de chaque itération de la boucle for.

2) Evaluer la complexité dans le meilleur et dans le pire des cas de l'appel tri(L) en fonction du nombre n d'éléments de L. Justifier votre réponse.

3) Citer un algorithme de tri plus efficace dans le pire des cas. Quelle en est la complexité dans le meilleur et dans le pire des cas ?

4) On souhaite, partant d'une liste constituée de couples (chaîne, entier), trier la liste par ordre croissant de l'entier associé suivant le fonctionnement suivant :

```
>>> L = [['Bresil', 76], ['Kenya', 26017], ['Ouganda', 8431]]
>>> tri_chaine(L)
>>> L
[['Bresil', 76], ['Ouganda', 8431], ['Kenya', 26017]]
```

Ecrire en Python une fonction **tri_chaine(L)** réalisant cette opération.

Exercice 2 :

1) Écrivez une fonction **sommeImpairs(L)** qui prend en paramètre une liste d'entiers et qui retourne la somme des éléments impairs de la liste.

2) Calculer la complexité temporelle de cette fonction dans le pire des cas et dans le meilleur des cas. .

Exercice 3 :

Considérons la fonction définie ci-dessous :

```
def P(x, n) :
        if (n == 0) return 1
        elif (n == 1) return x
        elif (n % 2 == 0) return P(x*x, n/2)
        else: return  x*P(x*x, n//2)
```

1) Que calcule cette fonction ?

2) Établir la relation de récurrence exprimant le coût de cette fonction en nombre de multiplications dans le cas le plus favorable. En déduire la complexité dans ce cas.

3) Établir la relation de récurrence exprimant le coût de cette fonction en nombre de multiplications dans le cas le plus défavorable. En déduire la complexité dans ce cas.

Exercice 4 :

Le but de ce problème est de gérer les appels (reçus, émis et manqués) des clients d'un opérateur téléphonique. Ces appels sont stockés dans une matrice **Client** à n lignes (n est le nombre d'appels enregistrés). Chaque ligne de la matrice contient les informations relatives à un appel donné. Il s'agit du numéro de téléphone (appelé ou appelant) représenté par dix caractères (première colonne), la durée d'appel en minutes (deuxième colonne) et le type d'appel (-1 si reçu, 1 si émis et 0 si manqué) (troisième colonne) en plus de l'heure (cinq caractères) et la date (dix caractères) stockés respectivement dans la quatrième et la cinquième colonne.

Un exemple de la matrice **Client**:

0654342310	5	-1	00h23	31/03/2014
0660324524	0	0	15h10	30/03/2014
0660324524	50	1	15h10	15/03/2014
0660324524	10	1	15h10	22/03/2014
0846394503	60	-1	19h03	01/02/2014
0654394503	13	-1	05h40	04/02/2014

1) Ecrire la fonction **sousChaine(S, d, f)** permettant de retourner la sous-chaine de S contenant les caractères de S se trouvant entre la position d et f.
Exemple : Si S='0654342310', alors l'appel suivant **sousChaine(S, 2, 5)** retournera la chaine '5434'.

Vu que certains clients sont dérangés par certains numéros qui les appellent entre minuit et 8h (inclues), on envisage définir, pour chaque client, une liste noire qui contiendra tous les numéros qui ont appelé au moins une fois dans cette période. Il est clair qu'un numéro ne doit pas apparaitre plus qu'une fois dans la liste noire. Ainsi, il faut, avant de l'insérer, vérifier s'il n'y figure pas déjà.

Exemple : Dans l'exemple ci-dessus, les numéros '0654342310' et '0654394503' devront apparaitre dans la liste noire.

2) Ecrire la fonction **appartient(num, L)** qui retourne True si le numéro num figure dans la liste L et False sinon.
Exemple : Si L=['0654342310','0660406587','0673536472'], alors l'appel suivant **appartient(**'0660406587'**, L)** retournera True

3) Ecrire la fonction **ListeAppels(Client)** qui prend en paramètre la matrice Client et qui retourne une matrice nommée R contenant la liste des appels : NumTél et HeureAppel.
Exemple : D'après l'exemple précédent, la fonction retourne la matrice R suivante :

0654342310	00h23
0660324524	15h10
0660324524	15h10
0660324524	15h10
0846394503	19h03
0654394503	05h40

4) Donner la complexité de la fonction **ListeAppels(Client)**

5) Ecrivez, ensuite, la fonction **listeNoire(R)** qui prend en paramètre la matrice R et qui retourne une liste contenant ces numéros.
Exemple : D'après l'exemple précédent (question 3), la fonction retournera la liste suivante : ***L=['***0654342310' ,'0654394503'***]***

6) Nous voulons définir pour chaque client les numéros qu'il a appelé ou qui l'ont appelé. Pour ce faire, on vous demande d'écrire une fonction d'entête **numeros(Client)** qui prend en paramètre la matrice Client et qui retourne une liste contenant tous ces numéros à l'exception des numéros publicitaires (les numéros publicitaires commencent par '08').

Attention : Un numéro ne doit pas apparaitre plus qu'une fois.
Exemple : D'après l'exemple, la fonction retournera la liste suivante : ***L=['***0654342310' ,'0660324524','0654394503'***]***