# TD noté, 12 décembre 2007

*Le programme construit au fur et à mesure des questions devra être imprimé à la fin du TD et rendu au chargé de TD. Il ne faut pas oublier de mentionner son nom en commentaires au début du programme. Des commentaires insérés dans le programme permettront de retrouver les réponses de chaque question.*

Lorsqu'on se connecte à un site internet, celui-ci enregistre votre adresse IP, l'heure et la date de connexion ainsi que la page ou le fichier désiré. Ainsi le fichier *logpdf.txt*[1] recense ces quatre informations séparées par des espaces pour les fichiers d'extension *pdf* téléchargés sur le site http://www.xavierdupre.fr/.

**1)** La première étape consiste à charger les informations depuis le fichier texte dans une matrice de 4 colonnes (liste de listes, liste de t-uples). On utilisera pour cela les fonctions `open` et les méthodes `split`, `replace` associées aux chaînes de caractères. On pourra s'aider des corrections des TD précédents. (2 points)

**2)** On souhaite dans un premier temps faire des statistiques sur les dates : on veut compter le nombre de fichiers téléchargés pour chaque date présente dans les données. On pourra pour cela utiliser un dictionnaire dont la clé sera bien choisie. Le résultat devra être inséré dans une fonction prenant comme entrée une matrice (ou liste de listes, liste de t-uples) et retournant un dictionnaire comme résultat. (4 points)

**3)** On s'intéresse au programme suivant :

```
l = [ (1, "un"), (3, "deux",), (2, "deux"), (1, "hun"), (-1, "moinsun") ]
l.sort (reverse = True)
print l
```

Donc le résultat est :

```
[(3, 'deux'), (2, 'deux'), (1, 'un'), (1, 'hun'), (-1, 'moinsun')]
```

Que s'est-il passé ? (2 points)

**4)** On désire maintenant connaître les 10 dates pour lesquelles il y a eu le plus de téléchargements ces jours-là. L'inconvénient est que le dictionnaire élaboré à la question 2 ne retourne pas les réponses dans l'ordre souhaité : il faut classer les dates par nombre de téléchargements croissants. Il faut ici imaginer une fonction qui retourne ces dix meilleures dates et des dix fréquentations correspondantes en se servant de la remarque de la question 3. (4 points) A quels événements correspondent les quatre premières dates ? (Cette petite question ne rapporte pas de point.)

**5)** Effectuez le même travail pour déterminer les dix documents les plus téléchargés. (2 points)

**6)** Ecrire une fonction qui retourne l'heure sous forme d'entier à partir d'une date définie par une chaîne de caractères au format "`hh` : `mm` : `ss`". Par exemple, pour "`14` : `55` : `34`", la fonction doit retourner 14 sous forme d'entier. L'instruction `int`("`14`") convertit une chaîne de caractères en un entier. (2 points)

**7)** Calculer le nombre de documents téléchargés pour chaque heure de la journée. Le site est-il consulté plutôt le matin ou le soir ? Serait-il possible de conclure aussi rapidement pour un site d'audience internationale ? (4 points)

1. Il faut télécharger ce fichier depuis l'adresse http://www.xavierdupre.fr/enseignement/initiation/logpdf.txt

### Correction

```
# coding: latin-1
# la première ligne autorise les accents dans un programme Python
# la langue anglaise est la langue de l'informatique,
# les mots-clés de tous les langages
# sont écrits dans cette langue.

####################################
# exercice 1
####################################
#

# question 1
def lit_fichier (file) :
    f = open (file, "r")
    li = f.readlines ()           # découpage sous forme de lignes
    f.close ()
    res = []
    for l in li :
        s = l.replace ("\n", "")
        s = s.split (" ")         # le séparateur des colonnes est l'espace
        res.append (s)
    return res


mat = lit_fichier ("logpdf.txt")
for m in mat [0:5] :       # on affiche les 5 premières lignes
    print m                # parce que sinon, c'est trop long

# question 2
def compte_date (mat) :
    d = { }
    for m in mat :
        date = m [1]   # clé
        if date in d : d [date] += 1
        else : d [date] = 1
    return d

dico_date = compte_date (mat)
print dico_date

# remarque générale : si le fichier logpdf.txt contient des lignes
# vides à la fin, il se produira une erreur à la ligne 34 (date = m [1])
# car la matrice mat contiendra des lignes avec une seule colonne et non quatre.
# Il suffit soit de supprimer les lignes vides du fichier logpdf.txt
# soit de ne pas les prendre en compte lors de la lecture de ce fichier.

# question 3
# La méthode sort trie la liste mais comment ?
# Il est facile de trier une liste de nombres mais une liste de couples de
# nombres ? L'exemple montre que la liste est triée selon le premier élément de
# chaque couple. Pour les cas où deux couples ont un premier élément en commun,
# les éléments semblent triés selon le second élément. L'exemple suivant le monte :

l = [(1, "un"), (3, "deux",), (2, "deux"), (1, "hun"), (1, "un"), (-1, "moinsun")]
l.sort (reverse = True)
print l  # affiche [(3, 'deux'), (2, 'deux'), (1, 'un'),
         #          (1, 'un'),   (1, 'hun'), (-1, 'moinsun')]
```

```
# question 4
def dix_meilleures (dico) :
    # dans cette fonction on crée une liste de couples (valeur,clé) ou
    # la clé représente une date et valeur le nombre de téléchargement
    # pour cette date
    li = []
    for d in dico :
        cle = d
        valeur = dico [cle]
        li.append ( ( valeur, cle ) )
    li.sort (reverse = True)
    return li [0:10]

dix = dix_meilleures (dico_date)
print dix # la première date est (283, '26/Sep/2007')

# les quatre premières dates correspondent aux quatre premiers TD en 2007 à l'ENSAE

# question 5
# la date est en colonne 1, le document en colonne 3
# on fait un copier-coller de la fonction compte_date en changeant un paramètre
def compte_document (mat) :
    d = { }
    for m in mat :
        doc = m [3]   # clé, 3 au lieu de 1 à la question 2
        if doc in d : d [doc] += 1
        else : d [doc] = 1
    return d

dix = dix_meilleures ( compte_document (mat) )
print dix   # le premier document est
            # (323, '/mywiki/Enseignements?.....target=python_cours.pdf'),

# question 6
def heure (s) :
    hs = s [0:2]           # on extrait la partie correspondant à l'heure
    return int (hs)        # on retourne la conversion sous forme d'entiers

# question 7
# on recommence avec un copier-coller
def compte_heure (mat) :
    d = { }
    for m in mat :
        h = m [2]   # clé, 2 au lieu de 1 à la question 2
        cle = heure (h)
        if cle in d : d [cle] += 1
        else : d [cle] = 1
    return d

h = compte_heure (mat)
dix = dix_meilleures ( h )
print dix   # la première heure est  (432, 17), ce qui correspond à l'heure des TD

for i in h :
    print i, "h  ", h [i]

# Il y a beaucoup plus de téléchargement entre 20h et 2h du matin
# que le matin avant 10h.
```

```
# Le site est plutôt consulté le soir.
# La conclusion ne serait pas aussi évidente avec un site consulté par des gens
# du monde entier puisque 6h du matin est une heure de l'après midi au Japon.
# Il faudrait croiser l'heure avec la position géographique de la personne
# qui consulte le site.
```