

TP 4

Exercice 1

Soit un fichier typé intitulé concours.txt qui comporte les enregistrements relatifs aux candidats d'un concours. Chaque enregistrement est composé de : NCIN, NOM, PRENOM, AGE, DECISION : (type contenant les identificateurs suivants : admis, refusé, ajourné), et séparé par point virgule (;).

Travail demandé :

Définir la fonction saisir() qui permet de remplir les données relatives aux candidats dans le fichier concours.txt

Définir la fonction admis() qui permet créer le fichier admis.txt comportant les données relatives aux candidat admis

Afin de sélectionner en priorité les candidats admis et âgés moins de 30 ans, créer la fonction attente() qui produira à partir du fichier admis.txt, un nouveau fichier intitulé attente.txt comportant les données relatives aux candidats admis et âgés plus que 30 ans. Une ligne du fichier attente.txt comprend le NCIN, le NOM et PRENOM d'un candidat séparés par point virgule (;).

Définir la fonction statistiques(dec) qui permet de retourner le pourcentage des candidats pour la décision dec (admis, refusé et ajourné). Exemple : Le pourcentage des candidats admis = (Nombre des candidats admis / Nombre des candidats) *100

Définir la fonction supprimer() qui supprimera du fichier admis.txt les candidat âgés plus que 30

Proposition de solution à compléter

```
def saisir():
    new = "O"  # O -> oui ; N -> non
    fichier = open("concours.txt", "a")
    decision = {"a": "admis(e)", "r": "refuse(e)", "aj": "ajourne(e)"}
    while new == "O":
        cin = input("Saisir le Numero CIN : ")
        nom = input("Saisir le Nom : ")
        prenom = input("Saisir le prenom : ")
        age = input("saisir l age ")
        dec = input("saisir la decision a(admis(e))
                    r(refuse(e))
                    aj(ajourne(e)) : ")
        ligne = cin+";"+nom+";"+prenom+";"+age+";"+decision[dec]+\n"
        fichier.write(ligne)
        new = input("Saisir un nouveau candidat, (O / N) ?")
    fichier.close()
```

```
def admis():
    fichier = open("concours.txt")
    dest = open("admis.txt", "a")
    for ligne in fichier:
```

```

L = ligne.split(";");
if L[4].strip() == "admis(e)":
    dest.write(ligne)
fichier.close()
dest.close()

def attente():
    fichier = open("admis.txt")
    dest = open("attente.txt", "a")
    for ligne in fichier:
        L = ligne.split(";");
        if int(L[3]) >= 30:
            enreg = L[0]+";"+L[1]+";"+L[2]+\n"
            dest.write(enreg)
    fichier.close()
    dest.close()

def statistiques(dec):
    fichier = open("concours.txt")
    L = fichier.readlines()
    fichier.close()
    L1 = [] # candidats admis
    L2 = [] # candidats refuses
    L3 = [] # candidats ajournes
    for ligne in L:
        L = ligne.split(";");
        if L[4].strip() == "admis(e)":
            L1.append(ligne)
        elif L[4].strip() == "refuse(e)":
            L2.append(ligne)
        else:
            L3.append(ligne)
    if dec == "admis":
        return (len(L1)/len(L))*100
    elif dec == "refuse":
        return (len(L2)/len(L))*100
    else:
        return (len(L3)/len(L))*100

def supprimer():
    fichier = open("admis.txt")
    candidat = [] # contient les candidats restants
    for ligne in fichier:
        L = ligne.split(";");
        if int(L[3]) < 30:
            candidat.append(ligne)
    fichier.close()
    # reecrire la nouvelle liste
    fichier = open("admis.txt", "w")
    fichier.writelines(candidat)
    fichier.close()

```

Exercice 2

Après la réussite au baccalauréat, les meilleurs bacheliers sont orientés vers les classes préparatoires aux grandes écoles. En effet, la priorité est à celui qui a le score (nombre de points) le plus élevé. Ce score est appelé formule globale.

Les élèves admissibles seront classés par ordre décroissant selon la formule globale. Puis, une fois classés, ces bacheliers seront divisés en 4 groupes de la façon suivante :

Questions pour cet exercice

(1) Combien de résidus nucléotidiques sont présents dans la séquence du gène du numéro d'accession KR711802.1 ?

(2) Com

Groupe	description	Nb d'étudiants	Répartition
Principale	Liste principale	30	40% (SMA) ; 30% (SMB) ; 20% (PC) ; 10% (SVT)
Tranche 1	Liste d'attente N°1	40	50% (SMA) ; 25% (SMB) ; 20% (PC) ; 5% (SVT)
Tranche 2	Liste d'attente N°2	60	40% (SMA) ; 30% (SMB) ; 20% (PC) ; 10% (SVT)
Tranche 3	Liste d'attente N°3	50	55% (SMA) ; 25% (SMB) ; 20% (PC) ;

on dispose d'un fichier nommé 'PSI.txt' contenant la liste des bacheliers admissibles de la section Physique et sciences industrielles. Dans ce fichier, chaque bachelier est défini par :

Num_insc : le numéro d'inscription

NP : le nom et prénom

FILIÈRE : le nom de la filière (SMA, SMB, PC ou SVT)

MG : moyenne générale.

FS (formule spécifique) : un réel déjà calculé à partir des notes obtenues dans les diverses matières.

i : un réel=1 si l'élève est redoublant en BAC et 1.10 sinon

On souhaite réaliser les fonctions suivantes :

On souhaite réaliser les fonctions suivantes :

- 1- formule_gen() qui permet de créer un autre fichier ‘PSI_FG.txt’, à partir du fichier ‘PSI.txt’, et y stocker, pour les mêmes bacheliers, les informations suivantes : Num_insc, NP, FILIERE et FG. Sachant que la formule globale (FG) de chaque élève est calculée par l’équation :
- $$FG=((5*MG+FS)*i)$$
- 2- classer() qui permet de classer les bacheliers du fichiers ‘PSI_FG.txt’ par ordre décroissant selon la formule globale(FG). N.B : pour faire cette tache, on doit transmettre les informations dans un tableau, les trier puis les transmettre ordonnées dans le fichier, implémenter un algorithme de tri que vous voulez pour trier le tableau
- 3- generer() qui permet d’extraire, dans le même répertoire, 4 autres fichiers (‘PSI_princ.txt’, ‘PSI_t1.txt’, ‘PSI_t2.txt’, ‘PSI_t3.txt’) contenant respectivement les bacheliers appartenant au groupe liste principale, tranche 1, tranche2 et tranche3.
- 4- afficher(Num_insc) qui permet d’afficher, pour un candidat donné, en fonction de son numéro d’inscription (donnée fournie en argument), le groupe auquel il appartient

Proposition de solution a completer

```

def formule_gen():
    source = open("PSI.txt")
    dest = open("PSI_FG.txt", "a")
    for ligne in source:
        s = ligne.strip()
        tab = s.split(":")
        fg = ((5*float(tab[3])+float(tab[4]))*float(tab[5]))
        etd = str(tab[0])+":"+str(tab[1])+":"+str(tab[2])+":"+str(fg)+"\n"
        dest.write(etd)

    source.close()
    dest.close()

def classer():
    source = open("PSI_FG.txt")
    L = source.readlines()
    for i in range(len(L)-1):
        m = i
        for j in range(i+1, len(L)):
            t1 = L[j].strip().split(":")
            t2 = L[m].strip().split(":")
            if float(t1[3]) > float(t2[3]):
                m = j
        L[i], L[m] = L[m], L[i]
    source.close()
    source = open("PSI_FG.txt", "w")
    source.writelines(L)
    source.close()

def generer():
    l1 = [] # SMA
    l2 = [] # SMB
    l3 = [] # PC

```

```

l4 = [] # SVT
# classser()
# classser dans 3 listes les etudiants selon la filiere
source = open("PSI_FG.txt")
for ligne in source:
    t = ligne.split(":")
    if t[2] == "SMA":
        l1.append(ligne)
    elif t[2] == "SMB":
        l2.append(ligne)
    elif t[2] == "PC":
        l3.append(ligne)
    else:
        l4.append(ligne)
source.close()
# construire la liste principale
nb1 = 12 # nombre des etudiants SMA
nb2 = 9 # nombre des etudiants SMB
nb3 = 6 # nombre des etudiants PC
nb4 = 3 # nombre des etudiants SVT
isma = 0 # indice de debut pour SMA
ismb = 0 # indice de debut pour SMB
ipc = 0 # indice de debut pour PC
isvt = 0 # indice de debut pour SVT
princ = open("PSI_princ.txt", "a")
for i in range(nb1):
    if i < len(l1):
        princ.write(l1[i])
for i in range(nb2):
    if i < len(l2):
        princ.write(l2[i])
for i in range(nb3):
    if i < len(l3):
        princ.write(l3[i])
for i in range(nb4):
    if i < len(l4):
        princ.write(l4[i])
princ.close()
# construire la tranche 1
isma = nb1
ismb = nb2
ipc = nb3
isvt = nb4
nb1 += 20
nb2 += 10
nb3 += 8
nb4 += 2
princ = open("PSI_t1.txt", "a")
for i in range(isma, nb1):
    if i < len(l1):
        princ.write(l1[i])
for i in range(ismb, nb2):
    if i < len(l2):
        princ.write(l2[i])
for i in range(ipc, nb3):
    if i < len(l3):
        princ.write(l3[i])
for i in range(isvt, nb4):
    if i < len(l4):

```

```

        princ.write(l4[i])
princ.close()
# construire la tranche 2
isma = nb1
ismb = nb2
ipc = nb3
isvt = nb4
nb1 += 24
nb2 += 18
nb3 += 12
nb4 += 6
princ = open("PSI_t2.txt", "a")
for i in range(isma, nb1):
    if i < len(l1):
        princ.write(l1[i])
for i in range(ismb, nb2):
    if i < len(l2):
        princ.write(l2[i])
for i in range(ipc, nb3):
    if i < len(l3):
        princ.write(l3[i])
for i in range(isvt, nb4):
    if i < len(l4):
        princ.write(l4[i])
princ.close()
# construire la tranche 3
isma = nb1
ismb = nb2
ipc = nb3
nb1 += 27
nb2 += 12
nb3 += 11
princ = open("PSI_t3.txt", "a")
for i in range(isma, nb1):
    if i < len(l1):
        princ.write(l1[i])
for i in range(ismb, nb2):
    if i < len(l2):
        princ.write(l2[i])
for i in range(ipc, nb3):
    if i < len(l3):
        princ.write(l3[i])
for i in range(isvt, nb4):
    if i < len(l4):
        princ.write(l4[i])
princ.close()

```

```

def afficher(Num_insc):
    princ = open("PSI_princ.txt")
    L1 = princ.readlines()
    tr1 = open("PSI_t1.txt")
    L2 = tr1.readlines()
    tr2 = open("PSI_t2.txt")
    L3 = tr2.readlines()
    tr3 = open("PSI_t3.txt")
    L4 = tr3.readlines()
    etat = False
    tranche = ""
    # chercher dans la liste principale
    for ligne in L1:

```

```
if str(Num_insc) == ligne[:len(str(Num_insc))]:
    etat = True
    tranche = "Liste principale"
    break
# si l etudiant n appartient pas a liste principale rechercher dans la
tranche 1
if etat == False:
    for ligne in L2:
        if str(Num_insc) == ligne[:len(str(Num_insc))]:
            etat = True
            tranche = "Tranche 1"
            break
# si l"etudiant n appartient pas a tranche 1 rechercher dans la tranche
2
if etat == False:
    for ligne in L3:
        if str(Num_insc) == ligne[:len(str(Num_insc))]:
            etat = True
            tranche = "Tranche 2"
            break
# si l"etudiant n appartient pas a tranche 2 rechercher dans la tranche
3
if etat == False:
    for ligne in L4:
        if str(Num_insc) == ligne[:len(str(Num_insc))]:
            etat = True
            tranche = "Tranche 3"
            break
return (etat, tranche)
```