université de BORDEAUX	**ANNEE UNIVERSITAIRE 2017 / 2018** SESSION 1 DE PRINTEMPS **PARCOURS : L1 Maths/Info** **Code UE : 4TPM205** **Date : 16 mai 2018** **Heure : 9h00** **Durée :** 1h30 **Documents :** non autorisés. **Epreuve de Mme :** C. Blanc. **Consignes :** Ce sujet comporte 8 pages d'exercices et 2 pages d'annexe avec les codes des algorithmes de tri de tableau. Vous pouvez détacher l'annexe mais ne séparez pas les autres pages. Vous devez **répondre directement sur le sujet**, insérez ensuite votre réponse dans une copie d'examen comportant tous les renseignements administratifs. **Indiquez votre numéro d'anonymat sur chaque feuille du sujet.**	Collège Sciences et Technologies

Numéro d'anonymat :

Questions de cours :

1. Soit t un tableau de n entiers, pour chacune des actions suivantes donnez son temps de calcul au pire.

`Rechercher si une valeur x est dans t`
`Échanger le contenu de 2 cases de t`
`Insérer un nouvel élément dans t`

2. Soit `t =[2,17,13,9,8,15,3,1]`. Lorsqu'on applique le tri insertion sur t pour trier le tableau selon l'ordre croissant, quels sont les deux premiers éléments déplacés ? Justifiez

3. Stabilité des tris :
   3.1. A quelle condition peut-on dire qu'un algorithme de tri de tableau est stable ?

   3.2. Question piège : soit t un tableau d'entiers tous distincts dites quels sont les algorithmes de tri vus en cours qui produiront un tri stable sur t.

Exercice 1 : Date de naissance

Une date de naissance est représentée par un entier sur 8 chiffres : JJMMAAAA où JJ, MM, AAAA représentent respectivement le jour entre 01 et 31, le mois entre 01 et 12 et l'année de naissance. Par exemple une personne née le 10 août 1984 aura une date de naissance représentée par l'entier 10081984

1. Ecrivez une fonction `annee(n)` qui prend en entrée un entier n représentant une date de naissance et retourne l'entier AAAA représentant l'année de naissance.

2. Ecrivez une fonction `mois(n)` qui prend en entrée un entier n représentant une date de naissance et retourne l'entier MM représentant le mois de naissance.

3. Ecrivez une fonction `jour(n)` qui prend en entrée un entier n représentant une date de naissance et retourne l'entier JJ représentant le jour de naissance.

Soit t un tableau de dates de naissance. On souhaite trier t selon **l'ordre croissant des dates de naissances.** Lorsque le tri sera terminé la date de naissance de la personne la plus âgée sera dans la case 0 de t et celle de la personne la plus jeune sera dans la case n-1.
Par exemple si `t=[25031999,10081984,20082001,15031990,02031999]` après le tri on aura `t=[10081984,15031990,02031999,25031999,20082001]`
Pour effectuer ce tri on propose d'appliquer successivement 3 algorithmes de tri (non nécessairement différents), un pour les jours, puis un pour les mois et enfin un pour les années.

4. Le premier algorithme triera les dates de naissance de façon croissante selon le jour de naissance. Par exemple si `t=[25031999,10081984,20082001,15031990,02031999]` après le tri sur les jours de naissance on aura `t=[02031999,10081984, 15031990, 20082001, 25031999]` car `02<10<15<20<25.`

   4.1. Soit `v=[12111985,25041940,06111996,15112000]` comment v sera-t-il modifié après le tri sur les jours :

Numéro d'anonymat :

4.2. Parmi les algorithmes vus en cours choisissez-en un pour effectuer le tri sur les jours. **Justifiez votre choix** et indiquez quelle(s) modification(s) vous devez faire dans le code choisi en annexe (N° de ligne et nouveau code).

5. Pour la suite on suppose que le tableau des dates de naissances est déjà trié en ordre croissant des jours de naissance, on souhaite maintenant trier le tableau en ordre croissant des mois. Par exemple si `t=[02031999,10081984, 15031990, 20082001, 25031999]` on souhaite avoir `t=[02031999, 15031990,25031999,10081984, 20082001]` après le tri sur les mois de naissance. On remarquera que **pour deux dates de naissance sur un même mois l'ordre des jours est conservé**. On voit que `0203199` est avant `15031990` car `02<15`.

5.1. Comment sera modifié le tableau v de la question 4.1 après ce tri sur les mois ?

5.2. Parmi les algorithmes vus en cours choisissez-en un pour effectuer ce tri sur les mois de naissance. **Justifiez soigneusement votre choix** et indiquez quelle(s) modification(s) vous devez faire dans le code choisi (N° de ligne(s) et nouveau code).

Pour la suite on suppose que le tableau des dates de naissances est déjà trié en ordre croissant des jours et mois de naissance, on souhaite maintenant trier le tableau en ordre croissant des années. Par exemple si `t=[02031999,15031990,25031999,10081984,20082001]` on souhaite avoir `t=[10081984,15031990,02031999,25031999,20082001]` après le tri sur les années de naissance. On remarquera que **pour deux dates de naissance sur une même année l'ordre des jours et mois est conservé**. Par exemple `02031999` est avant `25031999` car `0203<2503`.

6. Parmi les algorithmes vus en cours choisissez-en un pour effectuer ce tri sur les années de naissance. **Justifiez soigneusement votre choix** et indiquez quelle(s) modification(s) vous devez faire dans le code choisi (N° de ligne(s) et nouveau code).

7. Indiquez en le justifiant quelle est la complexité totale de la méthode complète pour trier un tableau de date de naissance.

8. **Question Bonus** : Sans écrire de fonction proposez une autre méthode pour réaliser le tri complet des dates de naissance sans parcourir plusieurs fois le tableau de dates.

Numéro d'anonymat :

Exercice 2 : Récursivité

Soit t un tableau de n éléments, on considère la fonction suivante :

```
def mystere(t,n):
  if n==0:
     return True
  return(t[n-1]%2==(n-1)%2 and mystere(t,n-1))
```

1. Si `v1=[2,5,4,9,8,1]`, quel sera le résultat des appels `mystere(v1,6)` ? Justifiez.

2. Si `v2=[12,3,5,7,8,11,2]`, quel sera le résultat de l'appel `mystere(v2,7)` ? Justifiez.

3. Quelle propriété possède un tableau `t` pour lequel `mystere(t,n)` retourne `True` ?

Exercice 3 : Tri de tableau

1. Soit `t=[1,6,8,11,3,7,15,9]` un tableau obtenu après l'application d'un algorithme de tri qui a été interrompu avant la fin de son exécution. En justifiant votre réponse indiquez quel(s) algorithme(s) vu(s) en cours a pu être utilisé pour obtenir `t` ?

2. Appliquez le tri fusion sur le tableau t. Vous détaillerez l'exécution du tri fusion par exemple en dessinant l'arbre des appels et en indiquant l'ordre des appels.

3. Est-ce que le fait d'appliquer le tri fusion sur un tableau dont les premiers éléments sont triés, améliore l'efficacité du tri fusion ? Justifiez.

Numéro d'anonymat :

Exercice 4 :

On souhaite écrire une fonction `rang(t,n,k)` qui prend en entrée un tableau de `n` entiers **non trié** et retourne la valeur du $k^{ième}$ plus petit élément de `t` (avec `k<n`). C'est-à-dire la valeur de l'élément de `t` qui serait dans la case d'indice `k-1` si le tableau était trié dans l'ordre croissant. Par exemple si `t=[5,2,4,12,9,15,6]` et `k=4` alors `rang(t,7,4)` retournera 6, c'est-à-dire le $4^{ème}$ plus petit élément de `t`.

1. Soit `t=[8,2,9,12,4,6,7,1,5,13]` un tableau de 10 entiers. Quelle valeur sera retournée par l'appel `rang(t,10,7)` ? Justifiez.

2. Pour écrire la fonction rang on choisit d'utiliser un des algorithmes de tri vu en cours en le modifiant. Quels sont les algorithmes de tri qui peuvent être utilisés pour obtenir un résultat sans trier complètement le tableau ?

3. Parmi les solutions possibles choisissez la plus efficace. Justifiez votre choix.

4. Modifiez le code de l'algorithme choisi à la question précédente pour écrire la fonction `rang(t,n,k)`

Annexe : t est un tableau d'entiers de taille n. Les algorithmes produisent un tri croissant.

Tri sélection :

```
1  def triSelection (t,n):
2    for  i in range(n) :
3      iMin=i
4      for j in range(i+1,n) :
5        if(t[j]<t[iMin]):
6          iMin=j
7      if(i != iMin):
8        echanger(t,i,iMin)
```

Tri insertion :

```
1  def triInsertion(t, n):
2    for i in range(1,n):
3      e=t[i]
4      j=i-1
5      while (j>=0 and t[j]>e):
6        t[j+1]=t[j]
7        j=j-1
8      t[j+1]=e
```

Tri à bulle :

```
1  def triBulle(t,n):
2    for i in range(n-1,0,-1) :
3      for j in range(i) :
4        if(t[j]>t[j+1]) :
5          echange(t,j,j+1)
```

Tri Fusion :

```
1  def fusion(t, d, f):
2    r=creerTableau(f-d)
3    m=(d+f)//2
4    i1=d
5    i2=m
6    k=0
7    while (i1<m and i2<f):
8      if (t[i1] < t[i2]):
9        r[k] = t[i1]
10       i1=i1+1
11     else :
12       r[k] = t[i2]
13       i2=i2+1
14     k=k+1
15   while (i1 <m):
16     r[k] = t[i1]
17     i1=i1+1
18     k=k+1
19   while (i2 <f):
20     r[k] = t[i2]
21     i2=i2+1
22     k=k+1
23   for k in range(f-d):
24     t[d+k]=r[k]
```

```
1  def triFusion (t,d,f):
2    if (d<f-1):
3      m=(d+f)//2
4      triFusion(t,d,m)
5      triFusion(t,m,f)
6      fusion(t,d,f)
```

1er Appel :

```
triFusion(t,0,n)
```

Tri rapide :

```
1  def separationPivot(t,n, d, f):
2    p = d
3    i = d
4    j = f
5    while(i<=j):
6      if (t[i]<t[p]):
7        i=i+1
8      else :
9        echanger(t,i,j)
10       if (i==p):
11         p=j
12       else :
13         if (j==p):
14           p=i
15       j=j-1
16   echanger (t, i, p)
17   return (i)
```

```
1  def triRapideRec(t, n, d, f):
2    if(d<f):
3      p=separationPivot(t,n,d,f)
4      triRapideRec(t,n,d,p-1)
5      triRapideRec(t,n,p+1,f)

   def triRapide(t, n):
     triRapideRec(t,n,0,n-1)
```