(13-14)

Programmation et structures de données

Examen (durée 1h30)

Soient les déclarations suivantes :

```
const int taille=100 ;
typedef
    struct cell
      { int inf ;
        struct cell * suiv ;
       } cellule ;
Int T[taille] ;
```

Et soit une séquence de nombres entiers se terminant par 999. La séquence n'est pas vide et ne dépasse en aucun cas 100 nombres entiers.

En utilisant ces déclarations écrire les sous-programmes suivants :

a) La sauvegarde de cette séquence dans le tableau T. (3 pts)

b) Le tri du tableau T dans l'ordre croissant en utilisant la méthode suivante :

On cherche la plus grande valeur parmi les n entiers et on la permute avec la dernière valeur du tableau. Puis on cherche la plus grande valeur parmi les n-1 valeurs et on la permute avec le nombre de l'avant dernière valeur. Et on refait la même chose avec les n-2 valeurs , les n-3 et ainsi de suite jusqu'à l'obtention d'un tableau complètement trié. (4 pts)

c) L'existence d'un nombre donné ainsi que sa position dans ce tableau T en utilisant la recherche dichotomique. X est l'élément recherché et M est l'élément du milieu de T. Si X = M alors on s'arrête . Si X > M alors la recherche se fait dans la partie droite de M. Si X < M alors la recherche se fait dans la partie gauche.(4 pts)

d) La création d'une liste dynamique chaînée contenant uniquement les nombres impairs du tableau T. (5 pts)

e) L'Impression de cette liste chaînée dans l'ordre décroissant. (4 pts)

(13-14)

Programmation et structures de données

Corrigé de l'examen

a) Int sauvegarde (int T[taille]) (1)

```
{int i=0,x ;
 scanf (''%d'',&x) ;
while (x != 999)
{T[i]=x ;
 i++ ;
 scanf (''%d'', &x) ;
}
return i ;
}
```

(2)

b) void tri (int T[taille], int n) (1)

```
{int i,j,imax,temp;
for (i=n-1;i>0;i--)
 {imax = 0;
 for (j=1;j<=i;j++)
     if (T[imax]<T[j])
         imax = j;
 temp=T[imax];
 T[imax]=T[i];
 T[i]=temp;
 }
}
```

(3)

c) int rechdichotomique(int T[taille], int X,int *pos, int g,int d) (1)

```
{int m;int tr=0;
 while (g<=d && !tr)
 {m=(g+d)/2;
  if (T[m]<X)
    g=m+1;
  else
   if (T[m]>X)
    d=m-1;
   else
    tr=1;
 }
if (tr) *pos=m+1;
return T[m]==X;
}
```

(3)

(13-14)

Programmation et structures de données

Corrigé de l'examen

a) Int sauvegarde (int T[taille]) (1)

```
{int i=0,x ;
 scanf (''%d'',&x) ;
while (x != 999)
{T[i]=x ;
 i++ ;
 scanf (''%d'', &x) ;
}
return i ;
}
```

(2)

b) void tri (int T[taille], int n) (1)

```
{int i,j,imax,temp;
for (i=n-1;i>0;i--)
 {imax = 0;
 for (j=1;j<=i;j++)
     if (T[imax]<T[j])
         imax = j;
 temp=T[imax];
 T[imax]=T[i];
 T[i]=temp;
 }
}
```

(3)

c) int rechdichotomique(int T[taille], int X,int *pos, int g,int d) (1)

```
{int m;int tr=0;
 while (g<=d && !tr)
 {m=(g+d)/2;
  if (T[m]<X)
    g=m+1;
  else
   if (T[m]>X)
    d=m-1;
   else
    tr=1;
 }
if (tr) *pos=m+1;
return T[m]==X;
}
```

(3)

d) cellule * insdebut(cellule *L, int x) 0.5

```
{cellule *q ;
 q = malloc(sizeof(cellule)) ;
 q->inf = x ;
 q->suiv = L ;
 return q ;
 }
```

2

```
cellule * creation (int T[taille], int n)
{int i ; cellule * L = NULL ;
 for (i=0 ;i<n ;i++)
     if (T[i] % 2)
       L=insdebut(L,T[i]) ;
 return L ;
 }
```

0.5

2

e) void impression(cellule *L) 1

```
{cellule *p ;
 p = L ;
 while (p != NULL)
    {printf ("%d  ", p->inf ) ;
     p = p->suiv ;
    }
 }
```

3