Informatique 2 : (12-13) **Examen**

Ex1 : (4 points)

Soit N un nombre entier positif. Ecrire en C les sous-programmes suivants :

a) Déterminer si N est premier ou non ?

b) Retourner le plus petit nombre premier plus grand ou égal à N.

Ex2 : (6 points)

Soit une file d'entiers statique circulaire.

Exemple :

	12	3	42	

Où tête = 1 et queue = 3

En C donner la structure de données qui correspond à cette file.

Ecrire en C les sous-programmes de gestion d'une telle file.

Ex3 : (4 points)

Ecrire en C un sous-programme qui lit le nombre d'éléments d'un tableau , réserve dynamiquement l'espace mémoire nécessaire pour ce tableau et remplit cet espace par des entiers. Le sous-programme communique au programme appelant l'adresse du tableau ainsi que le nombre d'éléments de ce tableau.

Ex4 : (6 points)

Soit une séquence de nombres entiers qui se termine par 999.

On voudrait sauvegarder cette séquence dans une liste linéaire chaînée dans le même ordre que la séquence.

Donner la structure de données correspondante en C pour cette liste chaînée.

Ecrire en C les sous-programmes suivants :

a) Insérer un nombre donné en queue de liste.

b) Créer cette liste.

c) Afficher la liste dans le même sens que la séquence.

Informatique 2 : (12-13) Corrigé de l'examen

Ex1 : a) int premier (int N)
{int i ;
for (i=2 ;i<=N/2 ;i++)
if ((N % i)==0)
return 0;
return 1;
}

2

b) Int prochain_premier(int n)
{while (!premier(n))
n++ ;
return n ;
}

2

EX2 : const int n=... ;
Typedef int tab[n] ;
Typedef struct
{tab t ;
Int tete,queue;
}file;

0.5

file initialiser(file f)
{f.tete=f.queue=-1 ;
return f ;
}

0.5

Int vide(file f)
{return (f.queue==-1) ?1:0;}

0.5

Int pleine(file f)
{return (f.tete==0&&f.queue==n-1 || f.tete==f.queue+1)?1:0;}

0.5

void deposer(file *f,int x)
{if (f->queue==n-1)
f->queue=0 ;
else
f->queue++ ;
if (f->tete==-1)
f->tete=0;
f->t[f->queue]=x;
}

2

```
Int enlever(file *f)
{int x;
 x=f->t[f->tete] ;
If (f->tete==f->queue)
  f->tete=f->queue=-1;
else
 If (f->tete==n-1)
   f->tete=0;
 else
   f->tete++;
return x;
}
```

2

Ex3 : int * remplirtab(int *nbelt)

```
{int *T ; int n,i ;
 printf("Donner la taille du tableau") ;
 scanf ("%d ",&n) ;
 *nbelt=n ;
 T=(int *)malloc(sizeof(int)*n);
 for (i=0;i<n,i++)
   scanf("%d",&T[i]);
 return T ;
 }
```

4

Ex4 : typedef

```
Struct cellule
  {int val ;
   Struct cellule *suiv ;
   }cell ;
```

0.5

a) cell * insereenqueue(cell *L, int elt)

```
{cell *p,*q ;
 q=(cell *)malloc(sizeof(cell));
 q->val = elt;
 q->suiv = NULL;
 if (L == NULL)
   L = q ;
 else
  {p = L;
   while (p->suiv != NULL)
     p = p->suiv;
   p->suiv = q;
   }
 return  L;
}
```

2

b)
```
cell * creation()
{int x;
 cell * L=NULL;
 scanf("%d",&x);
 while (x!=999)
   {L=insereenqueue(L,x);
    scanf("%d",&x);
    }
  return  L;
 }
```

2

c)
```
void affichage(cell*L)
{cell * p;
 p = L;
 while (p!=NULL)
  {printf("%d",p->val);
   p = p->suiv;
  }
 }
```

1.5