Examen – Session principale	
Matière : Atelier programmation II Enseignants : Dorsaf SEBAI et Amina JARRAYA Filière : MPI Nombre de pages : 6 pages	Semestre: II Date: 04/06/2021 Durée: 1h30 Documents : ***non autorisés***

Les réponses doivent être rédigées obligatoirement sur la feuille de réponses (pages 5 et 6)
L'examen contient 6 pages. Seulement les pages 5 et 6 sont à rendre.

Exercice 1:

Considérons les déclarations suivantes :

```
typedef struct Nœud
{
 int valeur;
struct Nœud * suivant;
}      Noeud;

 Typedef Noeud * liste;
```

Déterminer le rôle de chacune des fonctions suivantes (Répondre sur la feuille de réponses).

Question 1

```
// Ici les listes lst1 et lst2 sont triées par ordre croissant
void fonction_1 ( liste lst1 , liste lst2 , liste * p_lst ) {
        liste lst ;
        if ( lst1 == NULL )
                * p_lst = lst2 ;
        else if ( lst2 == NULL )
                * p_lst = lst1 ;
        else
        {
if ( lst1 → valeur < lst2 → valeur)
        {
fonction_1 ( lst1 → suivant , lst2 , & lst ) ;
                        lst1 → suivant = lst ;
                        * p_lst = lst1 ;
        }
                else {
fonction_1 ( lst1 , lst2 → suivant , & lst ) ;
                        lst2 → suivant = lst ;
                        * p_lst = lst2 ;
                }
        }
}
```

Question 2

```
void fonction_2 ( liste lst , liste * p_lst1 , liste * p_lst2
)
{
        if ( lst == NULL || lst → suivant == NULL )
        {
                * p_lst1 = lst ;
                * p_lst2 = NULL ;

        }
        else
        {
        liste lst1 = NULL , lst2 = NULL ;
fonction_2 ( lst → suivant → suivant ,
& lst1 , & lst2 ) ;
        lst → suivant → suivant = lst2 ;
        * p_lst2 = lst → suivant ;
        lst → suivant = lst1 ;
        * p_lst1 = lst ;
        }
}
```

Question 3

```
void fonction_3 ( liste * p_lst )
 {
        liste lst1 , lst2 ;
if ((* p_lst) != NULL && (* p_lst ) → suivant !=
NULL )
                {
        fontion_2  (* p_lst , & lst1 , & lst2 ) ;
        fonction_3 (& lst1 ) ;
        fonction_3 (& lst2 ) ;
        fonction_1 ( lst1 , lst2 , p_lst ) ;
                }
}
```

Question 4

```
liste  fonction_4(liste l, int n)  {
  liste R;
  if (l == NULL) {


    return (l);
  }
  if (l→valeur==n) {
    R= l;
    l= l→suivant;
    free(R);
    return (l);
  }
  else {
     l→suivant= fonction_4 (l→suivant,n);
     return  (l);
  }
}
```

Question 5

```
void fonction_5(liste la)
{
 liste p,q,h;
 int nb;
 p=la;
 while(p→suivant!=NULL){
        q=p;
        h=p→suivant;
        nb=0;
  while(h!=NULL){
   if(h→val==p→val){
        if(nb<2){
                nb++;
                h=h→suivant;
                q=q→suivant;
                 }
    else{
        q→suivant=h→suivant;
        free(h);
        h=q→suivant;
       }
   }
   else{
         h=h→suivant;
        q=q→suivant;
      }
  }
  p=p→suivant;
}}
```

Exercice 2:

Question 1 : Soit la fonction suivante :

```
#include<stdio.h>
#include<stdlib.h>

struct element{
int donnee;
element * suivant;
};

int fct (element * A, element * B){
 element * tmp, * ptr;

 tmp = B;
 while(tmp!=NULL){
       ptr=A;
       while(ptr != NULL){
              if(ptr->donnee == tmp->donnee){
                         ptr = ptr->suivant;
                         tmp = tmp->suivant;
              }
              else
                   break;
        }
       if(ptr == NULL)
                 return 1;
       else
       if(tmp->donnee != A->donnee)
                tmp = tmp->suivant;
}
 return 0;
}
```

Expliquez le rôle de la fonction *fct* en donnant des exemples (Répondre sur la feuille de réponses).

Question 2 : La fonction *dupliquer* permet de dupliquer les nombres pairs dans une liste d'entiers L. Exemple : si la liste L contient au début les éléments [3,11,8,5,16,7,4], L contiendra [3,11,8,8,5,16,16,7,4,4] après l'exécution de la fonction *dupliquer*.

Compléter le code de la fonction dupliquer (Répondre sur la feuille de réponses) :

```
element * dupliquer(element * debut){
 element * tmp, * nouv;
 tmp = debut;
 while(tmp!=NULL){
     if(tmp->donnee % 2 == 0){

            ……..

            ……..

            ……..
     } else
            ……..

 }
 return debut;}
```

Question 3 : Soit les deux programmes suivants (Répondre sur la feuille de réponses) :

```c
#include<stdio.h>
#include<stdlib.h>

struct element{
   int donnee;
   element * suivant;
};

element * operation(element * debut){
element * ptmp, * tmp;
ptmp = debut;
tmp = debut->suivant;
while(tmp != NULL){
    if(ptmp->donnee == tmp->donnee){
       ptmp->suivant = tmp->suivant;
       free(tmp);
       tmp = ptmp->suivant;
    }
    else{
       ptmp = tmp;
       tmp = tmp->suivant;
    }
}
return debut;
}
```

```c
#include<stdio.h>
#include<stdlib.h>

struct pile{
int donnee;
pile * precedent;
};

pile * mystere(pile *A, pile *B){
pile * C;
int x;
C = NULL;
while(est_vide(A)==0 && est_vide(B)== 0){
    if(sommet(A) <= sommet(B))
       x = depiler(&A);
    else
       x = depiler(&B);
       C = empiler(C, x);
    }
while(est_vide(A) == 0){
    x = depiler(&A);
    C = empiler(C, x);
}
while(est_vide(B) == 0){
    x = depiler(&B);
    C = empiler(C, x);
}
return C;
}
```

1. Soit une liste chainée L contenant les valeurs suivantes <7, 3, 3, 9, 6, 6, 6, 2>. Donnez le rôle et le résultat d'exécution de la fonction *operation* pour la liste L.

2. Donnez la pile C obtenue en précisant son sommet après exécution de la fonction *mystere* pour les piles A = [8, 7, 3, 2] (2 est le sommet de la pile A) et B = [16, 12, 11, 6, 4] (4 est le sommet de la pile B). (On suppose qu'on fait appel aux fonctions de manipulation des piles).