Université Abou Bekr Belkaid – Tlemcen
Faculté des Sciences
Département d'Informatique

Année Universitaire 2021 - 2022
L2 Licence Informatique
UEI-72 - M.BENAZZOUZ
Durée : 1 h

Partiel d'Algorithmique

Exercice 1 : (6 pts)

Quels résultats fournira le programme suivant :

```
#include <stdio.h>
int fct1(int *);
void fct2(int);
float pi=3.14;
int val=10;
    int main()
    {
            int val=5, *ptr;
            char c='\x31';
            ptr=&val;
            printf("debut main: c=%d val=%d *ptr =%d \n",c,val,*ptr);
            fct2(val);
            c=(int)pi;
            printf("fin main : c = %d  val =%d *ptr=%d \n",c,val,*ptr);
    return 0 ;}

 int fct1(int *n)
    {
      printf( "debut fct1 : n= %d\n",*n);
      if(*n<6)  { *n=*n+3;   return fct1(n);}
      else {printf("fin fct1 : n= %d \n",*n);  return 1;}
    }

 void fct2(int x)
    {
       if(x){ int val= fct1(&x)+1;
           printf("debut fct2:  x = %d val = %d \n", x,val); }
           x= x+1;
           printf("fin fct2:  x =%d val= %d \n", x,val);
```

Exercice 2 : (6 pts)

On dispose de deux tableaux d'entiers T1 [N] et T2 [M] triés par ordre croissant(strictement croissant). On souhaite vérifier que l'intersection de T1 et T2 est un ensemble vide.

- Écrire une fonction itérative. (2 pts)
- Écrire une fonction récursive. (4 pts)

Exemples :

T1

1	4	5	8	10

T2

2	6	9	15

Condition vérifiée, on retourne 1

T1

1	4	**6**	8	10

T2

2	**6**	9	15

Condition non vérifiée, on retourne 0

Exercice 3 : (6 pts)

Une chaîne « correctement parenthésée » contient le même nombre de parenthèses ouvrantes et fermantes. De plus, en tout point de la chaîne, on ne doit jamais avoir vu plus de parenthèses fermantes que de parenthèses ouvrantes.

Écrire la fonction récursive Inc_paren_recu(char chaine[ ]) prenant une chaîne de caractères en entrée et retournant suivant les cas :

— L'opposé de la position (à partir de 1) de la première parenthèse fermante en trop, s'il y a lieu ;

— Le nombre de parenthèses ouvrantes en trop (ou 0 si la chaîne est correctement parenthésée).

Exemples :

Inc_paren_recu ("(abc(defij)klmnop)") 0 = 2-2

Inc_paren_recu ("(abc(defghijklmn)") 1 = 2-1

Inc_paren_recu ("(abc(defghijklmn()") 2 = 3-1

Inc_paren_recu ("(abc(def)))ghijklmn**()**") -11

Solutions

Exercice 1 :

```
debut main: c=49 val=5 *ptr =5
debut fct1 : n= 5
debut fct1 : n= 8
fin fct1 : n= 8
debut fct2:   x = 8 val = 2
fin fct2:  x =9 val= 10
fin main : c = 3  val =5 *ptr=5
```

Exercice 2 :

- **Version itérative :**

```
int verif_id(int T1[],int N,int T2[],int M) {
 int i=0,j=0;
 while(i<N && j<M)
         {  if(T1[i]==T2[j]) return 0;
           if(T1[i]<T2[j]) i++; else  j++;
         }
  return 1;
}
```

- **Version récursive :**

```
int verif_id_rec(int T1[],int N,int T2[],int M)
 {
  if(M<0 || N<0)return 1;
        if(T1[N]==T2[M]) return 0;
         if(T1[N]<T2[M]) return verif_id_rec(T1,N,T2,M-1) ;
              else return verif_id_rec(T1,N-1,T2,M);
}
```

Exercice 3 :

```
int incor_par_rec(char str[],int ct,int i)
{
    if (ct < 0) return -i;
         if(str[i] == '\0') return ct;
                    switch (str[i]) {
                       case '(':++ct; break;
                       case ')':--ct; break;
                             }
                  ++i;
     return incor_par_rec(str, ct, i);
 }
```

Le premier appel de la fonction :

```
Int res= Inc_paren_recu ("(abc(defghijklmn()",0,0) ;
```