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

Année Universitaire 2023 - 2024
L2 Licence Informatique
UEI-72 - M.BENAZZOUZ
Durée : 1 h 15 mn

PARTIEL D'ALGORITHMIQUE

Barème : exercice 1 (4 points), exercice 2 (6 points), exercice 3 (4 points),

Exercice 1 :

Étant donné un entier ***E*** et un tableau ***T*** contenant ***N*** entiers triés en ordre croissant.

1. Écrire une fonction ***récursive*** qui permet de tester si ***E*** appartient à ***T*** en suivant le principe de la recherche Trichotomique : à chaque étape de l'algorithme, le tableau courant doit être divisé en trois parties et la recherche de ***E*** doit être effectuée sur l'une de ces parties.
2. Écrire le premier appel de cette fonction.

Exercice 2 :

On dispose d'un entier E et d'un tableau *T* composé de N entiers distincts, et on souhaite écrire une fonction pour retourner l'indice de l'élément le plus proche de E dans T ; la fonction doit s'arrêter immédiatement si E existe dans le T.

1. Écrire une fonction itérative.
2. Écrire une fonction récursive (avec son premier appel).

Exemple :

	0	1	2	3	4	5	6	7	8	9
T	10	27	13	0	89	70	49	-6	66	53

- Pour E = 89, la fonction doit retourner 4.
- Pour E = 7, la fonction doit retourner 0.
- Pour E = 68, la fonction doit retourner 5 ou 8.

Exercice 3 :

Quels résultats fournira le programme suivant :

```
#include<stdio.h>
  int recursion(int);
  void by_adress(int *);
  int val = 10;
  int main() {
      int val = 5 , *ptr;
      ptr = &val;
      printf("Start program: val = %d *ptr = %d\n",val,*ptr);
      by_adress(&val);
      printf("End program :  val = %d *ptr = %d \n",val,*ptr);
     return  0 ;
           }

int recursion(int N)
      {
        printf("Start recursion:  N = %d\n",N);
        return (N<8) ? recursion(N+2):1;
        printf("End recursion:  N=%d \n",N);
       }

void by_adress(int *adr)
       {
        if (*adr){
              int val = recursion(++(*adr));
              printf("Start by_adress: *adr=%d   val=%d \n",*adr,val);
                  }
        *adr=*adr+10;
        val++;
        printf("End by_adress:  *adr = %d   val = %d \n",*adr,val);
       }
```

Corrigé

Exercice 1: Recherche Trichotomique

```
int RechTricho(int T[], int E, int Debut, int Fin){
      if  (Debut > Fin)
            return 0 ;
      int Taille = (Fin – Debut) + 1 ;
      int M1 = Debut + Taille / 3 ;
      int M2 = Debut + 2*(Taille / 3) ;
      if  ((T[M1] == E) || (T[M2] == E) )   return 1 ;
           else if  (E < T[M1])
                              returne RechTricho(T, E, Debut, M1 - 1);
                    else      if ((E > T[M1]) && (E < T[M2]))
                              return RechTricho(T, E, M1 + 1, M2 - 1);
                       else
                              return RechTricho(T, E, M2 + 1, Fin);
}
```

// Appel de la fonction

```
int Trouv = RechTricho(T, 0, N) ;
```

Exercice 2 :

Itérative :

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

int indicePlusProche(int E, int T[], int N) {
    int Ind_Plus_Proche,difference, Diff_Min = abs(T[0] - E);

    for (int i = 0; i < N; ++i) {

        if (T[i] == E)  return i;
        difference = abs(T[i] - E);


        if (difference < Diff_Min) {

            Ind_Plus_Proche = i;
            Diff_Min = difference;
        }
    }

    return Ind_Plus_Proche;
}

int main() {
    // Exemple d'utilisation
    int E = 68;
    int T[] = {10,27,13,0,89,70,49,-6,66,53};
    int N = sizeof(T) / sizeof(T[0]);

    int indice = indicePlusProche(E, T, N);


    printf("L'indice de l'element le plus proche de %d dans le tableau est : %d\n",
E, indice);

    return 0;
}
```

Récursive :

```
#include <stdio.h>

int indicePlusProche(int E, int T[], int ind, int N, int best_ind, int Diff_Min) {

    if (ind == N) {
        return best_ind;
    }



    if (T[ind] == E) {
        return ind;
    }

    int difference = abs(T[ind] - E);



    if (difference < Diff_Min) {

        best_ind = ind;
        Diff_Min = difference;
    }
         return indicePlusProche(E, T, ind + 1, N, best_ind, Diff_Min);
}

int main() {

    int E = 68;
    int T[] = {10,27,13,0,89,70,49,-6,66,53};
    int N = sizeof(T) / sizeof(T[0]);

    int Diff_Min = T[N-1] - T[0] + 1;

    int indice = indicePlusProche(E, T, 0, N, -1, Diff_Min);

    printf("L'indice de l'element le plus proche de %d dans le tableau est : %d\n",
E, indice);

    return 0;
}
//premier appel
int indice = indicePlusProche(E, T, 0, N, -1, abs(T[N-1] - T[0]+1) );
```

Exercise 3 :

Start program: val = 5 *ptr = 5

Start recursion: N = 6

Start recursion: N = 8

Start by_adress: *adr=6 val=1

End by_adress: *adr = 16 val = 11

End program : val = 16 *ptr = 16