Programmation et structures de données II
Date : 22 Mai 2024
Durée : 2h
Semestre 2

Université Abou Bakr Belkaïd - Tlemcen
Faculté des Sciences
Département de Mathématiques
L1-Maths-MI

Examen Final

Les solutions doivent être rédigées en **C**
Les appareils portables doivent être éteints et posés sur le bureau du surveillant
Note CC = (Affichage1 + Ex2) *2
Note EF = Affichage (1 + 2) + Ex2 + Ex3

---

1 Affichage (8.5 pts)

Qu'affichent les deux programmes suivants :

(5.5pts)

```
void tab(int *T , int n){
   for (int i=0 ; i <8 ; i++)
    { *T= *(T+1);
      printf ("T[%d] = %d\n" , i , *T);
      T++; }
}
void mat(int n , int m , int T[n] , int A[][m])
{ int k=0;
  for(int i=0 ; i<n ; i++){
    for(int j=0 ; j<m ; j++)
       { A[i][j]=T[k++];
         printf("%d " , A[i][j]); }
         printf("\n"); }
}
void main(){
  int T[10]={ 1, 2, 3, 4, 5, 6};
  int A[2][4] = {0};
  mat(2,4,T,A);
  printf ("%d  %d  " , **A , *(*(A +1)));
  printf ("%d  " , *(*(A +1)+2));
  printf ("%d\n" , *(*A +1)+2);
  int  *p = T;
  printf ("*p = %d \n" , *p);
  tab(T,8);   }
```

(3pts)

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

void  MOT( char *T , char c){
  while (*T) {
  if (*T==c)     strcpy (T,T+1) ;
      else  T++;}
 }

 int main ()
 { char T[20]= "Hello!", S[20]= "Cool";
    int i;
  puts(strchr(T,'l'));
  MOT(T,'l');
  puts(T);
  char *P=S;
  int n=strlen(S);
  for ( i=n ; i <2*n ;  i++)
   *(P+i) = *(P+2*n-i-1) ;
  puts(S);
 return 0;
}
```

2 Fusionner et trier (4.5 pts)

1. Écrire la fonction **FusionTrie** qui prend en entrée deux tableaux d'entiers, les fusionne en un seul tableau, puis trie ce tableau en ordre décroissant.

2. La fonction **main()** doit :
   - Demander à l'utilisateur de saisir la taille et les éléments des deux tableaux.
   - Appeler la fonction **FusionTrie**
   - Afficher le tableau fusionné et trié.

Exemple d'exécution:
Taille du tableau A: 4
Éléments du tableau A: 5 4 9 1
Taille du tableau B: 5
Éléments du tableau B: 8 7 2 3 6
Tableau fusionné et trié : 9 8 7 6 5 4 3 2 1

3 Gestion des Employés d'une Entreprise (7 pts)

Une entreprise souhaite informatiser la gestion de ses employés. Pour ce faire, elle a besoin d'un programme qui permet de saisir et de gérer les informations de chaque employé. Les informations requises sur chaque employé comprennent :

* Nom de l'employé
* Identifiant de l'employé
* Poste occupé par l'employé
* Salaire de l'employé
* Ancienneté de l'employé en mois

1. Créer la structure **Employe**.
2. Écrire les deux fonctions **SaisirEmploye** et **AfficherEmploye**.
3. Écrire la fonction **Augmentation** qui permet d'augmenter le salaire d'un employé si son salaire actuel est inférieur à une certaine valeur donnée. L'augmentation de salaire doit être appliquée en pourcentage (*Exp*. 15%).
4. Écrire la fonction **Changement** qui permet de changer le poste d'un employé. Le changement de poste ne doit être autorisé que si l'employé a une ancienneté d'au moins 12 mois et que son poste actuel est différent du nouveau poste spécifié (*Exp*. Manager). Afficher un message approprié dans le cas où aucun changement n'a été effectué.
5. Écrire la fonction **main**() qui permet de tester toutes les fonctions ci-dessus, en affichant à la fin les informations mises à jour sur l'employé.

Exemple d'exécution :

```
Informations sur l'employé :
   Nom : Mohamed
   ID : 32716
   Poste : Assistant
   Salaire : 35000 DA
   Ancienneté en mois : 16

Le salaire de l'employé est déjà supérieur ou égal à 30000 DA.
Aucune augmentation n'a été effectuée.
Le poste de l'employé a été changé pour : Manager

Informations mises à jour sur l'employé :
   Nom : Mohamed
   ID : 32716
   Poste : Manager
   Salaire : 35000 DA
   Ancienneté en mois : 16
```

« Bon courage »

Final Exam

Solutions should be written in **C**
Portable devices should be turned off and placed on the supervisor's desk
CC = (Display1 + Ex2) *2
FE = Display (1 + 2) + Ex2 + Ex3

1 Display (9 pts)

What do the following two programs display :

(6pts)

```
void tab(int *T , int n){
   for (int i=0 ; i <8 ; i++)
    { *T= *(T+1);
      printf ("T[%d] = %d\n" , i , *T);
      T++; }
}
void mat(int n , int m , int T[n] , int A[][m])
{ int k=0;
  for(int i=0 ; i<n ; i++){
    for(int j=0 ; j<m ; j++)
       { A[i][j]=T[k++];
         printf("%d " , A[i][j]); }
         printf("\n"); }
}
void main(){
  int T[10]={ 1, 2, 3, 4, 5, 6};
  int A[2][4] = {0};
  mat(2,4,T,A);
  printf ("%d  %d  " , **A , *(*(A +1)));
  printf ("%d  " , *(*(A +1)+2));
  printf ("%d\n" , *(*A +1)+2);
  int  *p = T;
  printf ("*p = %d \n" , *p);
  tab(T,8);   }
```

(3pts)

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

void  MOT( char *T , char c){
  while (*T) {
  if (*T==c)     strcpy (T,T+1) ;
     else  T++;}
 }

 int main ()
 { char T[20]= "Hello!", S[20]= "Cool";
    int i;
  puts(strchr(T,'l'));
  MOT(T,'l');
  puts(T);
  char *P=S;
  int n=strlen(S);
  for ( i=n ; i <2*n ;  i++)
   *(P+i) = *(P+2*n-i-1) ;
  puts(S);
 return 0;
}
```

2 Merge and Sort (4 pts)

1. Write the function **MergeSort** that takes two arrays of integers as input, merges them into a single array, then sorts this array in descending order.

2. The function **main()** should :
   - Ask the user to enter the size and elements of the two tables.
   - Call the **MergeSort** function.
   - Display the merged and sorted table.

Execution Example:
Size of table A: 4
Elements of table A: 5 4 9 1
Size of table B: 5
Elements of table B: 8 7 2 3 6
Merged and sorted table: 9 8 7 6 5 4 3 2 1

3 Management of a Company's Employees (7 pts)

A company wants to computerise the management of its employees. To do this, it needs a program that captures and manages each employee's information. The information required on each employee includes:

* Employee Name
* Employee ID
* Position Held by the Employee
* Employee Salary
* Employee Seniority in Months

1. Create the **Employee** Structure.
2. Write the two functions **EnterEmployee** and **DisplayEmployee**.
3. Write the function **Increase** which increases an employee's salary if their current salary is less than a certain value. The salary increase must be applied as a percentage (*Exp*. 15%).
4. Write the function **Change** which allows you to change an employee's position. The change of position should only be authorised if the employee has at least 12 months' a seniority and their current position is different from the new position specified (*Exp*. Manager).
   Display an appropriate message if no changes have been made.
5. Write the function **main()** that tests all of the above functions, displaying the updated employee information at the end.

Execution Example :

```
Employee information :
  Name : Mohamed
  ID : 32716
  Position : Assistant
  Salary : 35000 DA
  Seniority in months : 16

The employee's salary is already greater than or equal to 30000 DA.
No increase has been made.
The employee's position has been changed to: Manager

Updated employee information:
  Name : Mohamed
  ID : 32716
  Position : Manager
  Salary : 35000 DA
  Seniority in months: 16
```

« Good luck »

Programmation et structures de données II
Date : 22 Mai 2024
Durée : 2h
Semestre 2

Université Abou Bakr Belkaïd - Tlemcen
Faculté des Sciences
Département de Mathématiques
L1-Maths-MI

Examen Final

Les solutions doivent être rédigées en **C**
Les appareils portables doivent être éteints et posés sur le bureau du surveillant
Note CC = (Affichage1 + Ex2) *2
Note EF = Affichage (1 + 2) + Ex2 + Ex3

---

1 Affichage (8.5 pts)

Qu'affichent les deux programmes suivants :

(5.5pts)

```
void tab(int *T , int n){
   for (int i=0 ; i <8 ; i++)
    { *T= *(T+1);
      printf ("T[%d] = %d\n" , i , *T);
      T++; }
}
void mat(int n , int m , int T[n] , int A[][m])
{ int k=0;
  for(int i=0 ; i<n ; i++){
    for(int j=0 ; j<m ; j++)
       { A[i][j]=T[k++];
         printf("%d " , A[i][j]); }
         printf("\n"); }
}
void main(){
  int T[10]={ 1, 2, 3, 4, 5, 6};
  int A[2][4] = {0};
  mat(2,4,T,A);
  printf ("%d  %d  " , **A , *(*(A +1)));
  printf ("%d  " , *(*(A +1)+2));
  printf ("%d\n" , *(*A +1)+2);
  int  *p = T;
  printf ("*p = %d \n" , *p);
  tab(T,8);   }
```

(3pts)

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

void  MOT( char *T , char c){
  while (*T) {
  if (*T==c)     strcpy (T,T+1) ;
     else  T++;}
 }

 int main ()
 { char T[20]= "Hello!", S[20]= "Cool";
   int i;
  puts(strchr(T,'l'));
  MOT(T,'l');
  puts(T);
  char *P=S;
  int n=strlen(S);
  for ( i=n ; i <2*n ;  i++)
   *(P+i) = *(P+2*n-i-1) ;
  puts(S);
 return 0;
}
```

(5.5pts)

1 2 3 **4**
5 6 0 0
1 5 0 4
*p = 1
T[0]=2
T[1]=3
T[2]=4
T[3]=5
T[4]=6
T[5]=0
T[6]=0
T[7]=0

(0.25 par valeur)

(3pts)

llo**!**
Heo!
CoollooC

(1pt par affichage)

2 Fusionner et trier (4.5 pts)

1. Écrire la fonction **FusionTrie** qui prend en entrée deux tableaux d'entiers, les fusionne en un seul tableau, puis trie ce tableau en ordre décroissant.

2. La fonction **main()** doit :
   * Demander à l'utilisateur de saisir la taille et les éléments des deux tableaux.
   * Appeler la fonction **FusionTrie**
   * Afficher le tableau fusionné et trié.

Exemple d'exécution:
Taille du tableau A: 4
Éléments du tableau A: 5 4 9 1
Taille du tableau B: 5
Éléments du tableau B: 8 7 2 3 6
Tableau fusionné et trié : 9 8 7 6 5 4 3 2 1

Solution

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_TAILLE 100

void fusionTrie(int *A, int a, int *B, int b, int *C, int *c) {
   *c = a + b;
   for (int i=0 ; i<a ; i++) C[i] = A[i];
   for (int i=0 ; i<b ; i++) C[a+i] = B[i];
   for (int i=0 ; i<*c ; i++)
      for (int j=i+1 ; j<*c ; j++)
         if (C[i] < C[j])
           {  int temp = C[i];
              C[i] = C[j];
              C[j] = temp;  }
 }

void afficherTableau(int *tableau, int taille) {
   for (int i=0 ; i<taille ; i++)   printf("%d ", tableau[i]);
 }

int main() {
   int A[MAX_TAILLE], B[MAX_TAILLE], C[2*MAX_TAILLE];
   int sizeA,sizeB,sizeC;
   printf("La taille du tableau A: ");scanf("%d", &sizeA);
   printf("Les éléments du tableau A:\n");
   for (int i=0 ; i<sizeA ; i++)   scanf("%d", &A[i]);
   printf("La taille du tableau B: ");scanf("%d", &sizeB);
   printf("Les éléments du tableau B:\n");
   for (int i=0 ; i<sizeB ; i++) scanf("%d", &B[i]);
   fusionTrie(A,sizeA,B,sizeB,C,&sizeC);
   printf("Tableau fusionné et trié : ");
   afficherTableau(C,sizeC);
   return 0;
}
```

3 Gestion des Employés d'une Entreprise (7 pts)

Une entreprise souhaite informatiser la gestion de ses employés. Pour ce faire, elle a besoin d'un programme qui permet de saisir et de gérer les informations de chaque employé. Les informations requises sur chaque employé comprennent :

* Nom de l'employé
* Identifiant de l'employé
* Poste occupé par l'employé
* Salaire de l'employé
* Ancienneté de l'employé en mois

1. Créer la structure **Employe**.

```
typedef struct {
    char nom[100];
    int id;
    char poste[100];
    float salaire;
    int anciennete; // Ancienneté en mois
} Employe;
```

2. Écrire les deux fonctions **SaisirEmploye** et **AfficherEmploye**.

```
void SaisirEmploye (Employe *employe) {
    printf("Entrez le nom de l'employe : ");
    scanf("%s", employe->nom);
    printf("Entrez le numero d'identification de l'employe : ");
    scanf("%d", &employe->id);
    printf("Entrez le poste de l'employe : ");
    scanf("%s", employe->poste);
    printf("Entrez le salaire de l'employe : ");
    scanf("%f", &employe->salaire);
    printf("Entrez l'anciennete de l'employe en mois : ");
    scanf("%d", &employe->anciennete);
}
```

```
void AfficherEmploye(Employe employe) {
    printf("Nom : %s\n", employe.nom);
    printf("ID : %d\n", employe.id);
    printf("Poste : %s\n", employe.poste);
    printf("Salaire : %.2f\n", employe.salaire);
    printf("Anciennete en mois : %d\n", employe.anciennete);
}
```

3. Écrire la fonction **Augmentation** qui permet d'augmenter le salaire d'un employé si son salaire actuel est inférieur à une certaine valeur donnée. L'augmentation de salaire doit être appliquée en pourcentage (*Exp*. 15%).

```
void Augmentation (Employe *employe, float pourcentage, float salaire_max) {
   if (employe->salaire < salaire_max) {
      employe->salaire *= (1 + pourcentage / 100);
      printf("Le salaire de l'employe a ete augmente de %.2f pourcent.\n", pourcentage);
                                                }
          else
      printf("Le salaire de l'employe est deja superieur ou egal a %.2f. Aucune augmentation n'a
                ete effectuee.\n", salaire_max);
   }
```

4. Écrire la fonction **Changement** qui permet de changer le poste d'un employé. Le changement de poste ne doit être autorisé que si l'employé a une ancienneté d'au moins 12 mois et que son poste actuel est différent du nouveau poste spécifié (*Exp*. Manager). Afficher un message approprié dans le cas où aucun changement n'a été effectué.

```
void Changement (Employe *employe,  char nouveau_poste[]) {
   if (strcmp(employe->poste, nouveau_poste) != 0 && employe->anciennete >= 6) {
        strcpy(employe->poste, nouveau_poste);
        printf("Le poste de l'employe a ete change pour : %s\n", nouveau_poste);      }
      else if (strcmp(employe->poste, nouveau_poste) == 0)
           printf("L'employe occupe deja le poste de %s. Aucun changement n'a ete effectue.\n",
                      nouveau_poste);
             else
           printf("L'employe doit avoir une anciennete d'au moins 6 mois pour changer de poste.
                      Aucun changement n'a ete effectue.\n");
}
```

5. Écrire la fonction **main**() qui permet de tester toutes les fonctions ci-dessus, en affichant à la fin les informations mises à jour sur l'employé.

```
int main() {
   Employe employe1;
   saisir_employe(&employe1);
   printf("\nInformations sur l'employe :\n");
   afficher_employe(employe1);
   augmenter_salaire(&employe1, 10, 30000);
   changer_poste(&employe1, "Manager");
   printf("\nInformations mises a jour sur l'employe :\n");
   afficher_employe(employe1);
return 0;
}
```

Exemple d'exécution :

Informations sur l'employé :
Nom : Mohamed
ID : 32716
Poste : Assistant
Salaire : 35000 DA
Ancienneté en mois : 16

Le salaire de l'employé est déjà supérieur ou égal à 30000 DA. Aucune augmentation n'a été effectuée.
Le poste de l'employé a été changé pour : Manager

Informations mises à jour sur l'employé :
Nom : Mohamed
ID : 32716
Poste : Manager
Salaire : 35000 DA
Ancienneté en mois : 16

Programming and Data Structures II
Date: May 22, 2024
Examination time : 2h
Semester 2

Abou Bakr Belkaïd University - Tlemcen
Faculty of Science
Mathematics Department
L1-Maths-MI

Final Exam

Solutions should be written in **C**
Portable devices should be turned off and placed on the supervisor's desk
CC = (Display1 + Ex2) *2
FE = Display (1 + 2) + Ex2 + Ex3

1 Display (9 pts)

What do the following two programs display :

(6pts)

```
void tab(int *T , int n){
   for (int i=0 ; i <8 ; i++)
    { *T= *(T+1);
      printf ("T[%d] = %d\n" , i , *T);
      T++; }
}
void mat(int n , int m , int T[n] , int A[][m])
{ int k=0;
  for(int i=0 ; i<n ; i++){
    for(int j=0 ; j<m ; j++)
       { A[i][j]=T[k++];
         printf("%d " , A[i][j]); }
         printf("\n"); }
}
void main(){
  int T[10]={ 1, 2, 3, 4, 5, 6};
  int A[2][4] = {0};
  mat(2,4,T,A);
  printf ("%d  %d  " , **A , *(*(A +1)));
  printf ("%d  " , *(*(A +1)+2));
  printf ("%d\n" , *(*A +1)+2);
  int  *p = T;
  printf ("*p = %d \n" , *p);
  tab(T,8);   }
```

(3pts)

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

void  MOT( char *T , char c){
  while (*T) {
  if (*T==c)     strcpy (T,T+1) ;
     else  T++;}
 }

 int main ()
 { char T[20]= "Hello!", S[20]= "Cool";
    int i;
  puts(strchr(T,'l'));
  MOT(T,'l');
  puts(T);
  char *P=S;
  int n=strlen(S);
  for ( i=n ; i <2*n ;  i++)
   *(P+i) = *(P+2*n-i-1) ;
  puts(S);
 return 0;
}
```

2 Merge and Sort (4 pts)

1. Write the function **MergeSort** that takes two arrays of integers as input, merges them into a single array, then sorts this array in descending order.

2. The function **main()** should :
   - Ask the user to enter the size and elements of the two tables.
   - Call the **MergeSort** function.
   - Display the merged and sorted table.

Execution Example:
Size of table A: 4
Elements of table A: 5 4 9 1
Size of table B: 5
Elements of table B: 8 7 2 3 6
Merged and sorted table: 9 8 7 6 5 4 3 2 1

3 Management of a Company's Employees (7 pts)

A company wants to computerise the management of its employees. To do this, it needs a program that captures and manages each employee's information. The information required on each employee includes:

* Employee Name
* Employee ID
* Position Held by the Employee
* Employee Salary
* Employee Seniority in Months

1. Create the **Employee** Structure.
2. Write the two functions **EnterEmployee** and **DisplayEmployee**.
3. Write the function **Increase** which increases an employee's salary if their current salary is less than a certain value. The salary increase must be applied as a percentage (*Exp*. 15%).
4. Write the function **Change** which allows you to change an employee's position. The change of position should only be authorised if the employee has at least 12 months' a seniority and their current position is different from the new position specified (*Exp*. Manager).
   Display an appropriate message if no changes have been made.
5. Write the function **main()** that tests all of the above functions, displaying the updated employee information at the end.

Execution Example :

```
Employee information :
  Name : Mohamed
  ID : 32716
  Position : Assistant
  Salary : 35000 DA
  Seniority in months : 16

The employee's salary is already greater than or equal to 30000 DA.
No increase has been made.
The employee's position has been changed to: Manager

Updated employee information:
  Name : Mohamed
  ID : 32716
  Position : Manager
  Salary : 35000 DA
  Seniority in months: 16
```

« Good luck »