Correction TP N° 1

Exercice 1: Ecrire un message

```
#include <stdio.h>
int main()
{
   printf("*************************************************");
   printf("\n*     BIENVENUE DANS LE MONDE          *") ;
   printf("\n*     MAGIQUE DE PROGRAMMATION     *") ;
   printf("\n*************************************************");
   return 0;
}
#include <stdio.h>
int main()
{
   printf("*************************************************");
   printf("\n*     BIENVENUE DANS LE MONDE          *") ;
   printf("\n*     MAGIQUE DE PROGRAMMATION     *") ;
   printf("\n*   Programme écrit par « FLEN »     *") ;
   printf("\n*     Etudiant(e) ISET NABEUL, A.U 17/18    *") ;
   printf("\n*************************************************");
   return 0;
}
```

Exercice 3:

```
#include <stdio.h>
int main()
{
      int x, y, z;
      printf ("donner un entier :");
           scanf("%d",&x) ;
      printf ("donner un deuxième entier :");
      scanf("%d",&y) ;
            z = x+y;
         printf("%d",z);
         return 0;
}
```

Correction TP N° 2

Exercice 1

```
#include <stdio.h>
int main()
{
  int x, y, z, w;
  printf ("donner un entier:\n") ;
  scanf("%d",&x) ;
  printf ("donner un deuxième entier:\n") ;
  scanf("%d",&y) ;
  z = x/y ;
   w = x % y ;
  printf("Le quotient de la division de %d par %d =  %d",x,y,z);
  printf("Le reste de la division de %d par %d =  %d",x,y,w);
  return 0;
}
```

Exercice 2

```
#include <stdio.h>
int main()
{
  int nbr, S ;
  S=0 ;
  printf ("donner le nombre 1:\n") ;
  scanf("%d",&nbr) ;
  S = S + nbr ;
  printf ("donner le nombre 2:\n") ;
  scanf("%d",&nbr) ;
  S = S + nbr ;
  printf ("donner le nombre 3:\n") ;
  scanf("%d",&nbr) ;
  S = S + nbr ;
  printf ("\n S = %d",S) ;
 return 0 ;
}
```

Exercice 3

```
#include <stdio.h>
int main()
{
  int x, y, S, P, D ;
  float M ;
  S=0 ;
  P = 1 ;
  printf ("donner deux entiers:\n") ;
  scanf("%d%d",&x,&y) ;
  S = x + y ;
  P = x * y ;
  D = x - y ;
  M = (float) S / 2 ;
```

```
  printf ("\n S = %d \n P = %d \n D = %d \n M = %f ",S,P,D,M) ;
 return 0 ;
}
```

Exercice 4

```
#include <stdio.h>
int main()
{
  int c, P, S ;
  printf ("donner le coté d'un carré:\n") ;
  scanf("%d",&c) ;
  P = c * 4 ;
  S = c * c ;
  printf ("\n Le périmètre P = %d \n La surface S = %d ",P,S) ;
 return 0 ;
}
```

Exercice 5

```
#include <stdio.h>
#include <math.h>
int main()
{
  int a, b, R ;
  R=0 , S = 0;
  printf ("donner deux entiers:\n") ;
  scanf("%d%d",&a,&b) ;
  S = a+ b ;
  R = pow(S, 2) ;
  printf ("\n (a+b)2= %d ",R) ;
 return 0 ;
}
```

Exercice 6

```
#include <stdio.h>
#include <math.h>
int main()
{
  int a ;
  float R=0.0 ;
  printf ("donner un entier:\n") ;
  scanf("%d",&a) ;
  R = sqrt(a) ;
  printf ("\n Racine carré de %d = %f ",a, R) ;
 return 0 ;
}
```

Exercice 7

```
#include <stdio.h>
int main()
{
  int x, y;
  int aux ;
  x = 3;
  y = 2;
  printf("le continue de x et y avant la permutation:\n");
```

```
  printf("x = %d \ny = %d\n",x,y);
  aux = x ;
  x = y;
  y = aux;
  printf("le continue de x et y apres la permutation:\n");
  printf("x = %d \ny = %d",x,y);
  return 0;
}
```

Exercice 8

```
#include <stdio.h>
int main()
{
  int x, y, z ;
  int aux ;
  printf("donner 3 entiers :\n");
  scanf("%d%d%d",&x,&y,&z);
  aux = x ;
  x = y;
  y = z ;
  z = aux;
  printf("le continue de x , y et z après la permutation:\n");
  printf("x = %d \n y = %d \n z = %d",x,y,z);
  return 0;
}
```

Exercice 9

```
#include <stdio.h>
int main()
{
  int jour, annee ;
  char mois[10] ;
  printf("donner le jour :\n");
  scanf("%d",&jour);
  printf("donner le mois :\n");
  scanf("%s",mois);
  printf("donner l'année :\n");
  scanf("%d",&annee);
  printf("La date du jour est : %d %s %d.",jour, mois, annee);
  return 0;
      }
```

Exercice 10

```
#include <stdio.h>
int main()
{
  int T, h, m, s ;
  printf("donner une durée T :\n");
  scanf("%d",&T);
 h = T / 60 ;
 m = (T % 60) / 60 ;
 s = (T % 60) % 60 ;
  printf("\n %d : h  %d : mn  %d : s",h,m,s);
  return 0;
        }
```

Exercice 11

```
#include <stdio.h>
int main()
{
  int larg, long, S ;
  printf("donner la largeur d'un rectangle :\n");
  scanf("%d",&larg);
  printf("donner la longueur d'un rectangle :\n");
  scanf("%d",&long);
 S = larg * long ;
 printf("\n Le rectangle dont la longueur mesure %d mètres et la
largeur mesure %d      mètres, a une surface égale à %d mètres
carrés.",long,larg,S);
  return 0;
 }
```

Exercice 13

```
#include <stdio.h>
int main()
{     int a=3,b,c;
      a=a*5 ;
      printf ("a=%d",a ) ;
      a++ ;
      b=a ;
      printf ("\n b vaut %d et a vaut %d",b,a );
      c=b;
      b++;
      printf ("\n c vaut %d et b vaut %d",c,b);
      return 0;
}
```

Exercice 14

```
#include <stdio.h>
int main()
{     int PNET, TVA ;
      double PBRUT ;
      printf ("\n Donner le prix net :" ) ;
      scanf("%d",&PNET);
printf ("\n Donner le TVA :" ) ;
      scanf("%d",&TVA);
      PBRUT = (double)PNET+PNET*TVA/100 ;
      printf ("\n Le prix brut est %f",PBRUT );
      return 0;
}
```

Exercice 15

```
#include <stdio.h>
int main()
{     int nb ;
float O, K, M ;
      printf ("\n Donner un nombre en bit :" ) ;
      scanf("%d",&nb);
      O = (float) nb / 8 ;
```

```
      K = O / 1000 ;
      M = K / 1000 ;
      printf ("\n %d bits = %f octets \n %f kilo octets \n %f Mega
octets",O,K,M );
      return 0;
}
```

Correction TP N° 3

Exercice 1

```
#include <stdio.h>
int main()
{     int nbr ;
float O, K, M ;
      printf ("\n Donner un entier :" ) ;
      scanf("%d",&nbr);
      if(nbr<0)
printf ("\n Le nbr %d est négatif",nbr );
            else
                  if(nbr>0)
      printf ("\n Le nbr %d est positif",nbr );
else
      printf ("\n Le nbr %d est nul",nbr );
      return 0;
}
```

Exercice 2

```
#include<stdio.h>
int main ()
{
      int a,b;
      printf("Donner un entier A \n");
      scanf("%d",&a);
      printf("Donner un entier B \n");
      scanf("%d",&b);
      if (a>b)
         printf ("A = %d est supérieur à B = %d",a,b);
         else if(a<b)
                  printf ("A = %d est inférieur à B = %d",a,b);
              else
                  printf ("A = %d est égale à B = %d",a,b);
      return 0;
    }
```

Exercice 3

```
#include <stdio.h>
#include <conio.h>
void main()
{
      int a,b,c, min, max;
      printf("donner trios entiers : ");
      scanf("%d%d%d",&a,&b,&c);
if (a>b)
{     max=a;
min=b;
}
else
```

```
{     max=b;
min=a;
}
if (max<c)
 max=c;
if (min<c)
min=c;
      printf("le min est %d, le max est %d", min, max) ;
      getch() ;
}
```

Exercice 4

```
#include <stdio.h>
#include <conio.h>
void main()
{
      int X ;
      printf("introduire un entier: ");
      scanf("%d",&X);
      if(X%2==0)
            printf(" %d est pair",X);
      else
            printf("%d est impair",X);
      getch() ;
    }
```

Exercice 5

```
#include <stdio.h>
#include <conio.h>
void main()
{
      float a, b, x ;
      printf("introduire deux réels a et b :");
      scanf("%f%f ",&a,&b) ;
      if(a !=0)
      {     x= -b/a ;
            printf("la solution est x = %f ",x) ;
      }
      else
            printf("ERREUR !! ") ;
      getch() ;
}
```

Exercice 6

```
#include <stdio.h>
#include <math.h>
 main()
{
 /* Calcul des solutions réelles d'une équation du second degré */
 int A, B, C;
 double D; /* Discriminant */
 printf("Calcul des solutions réelles d'une équation du second \n");
 printf("degré de la forme   ax^2 + bx + c = 0 \n\n");
 printf("Introduisez les valeurs pour a, b, et c : ");
```

```
 scanf("%i %i %i", &A, &B, &C);

 /* Calcul du discriminant b^2-4ac */
 D = pow(B,2) - 4.0*A*C;

 /* Distinction des différents cas */
 if (A==0 && B==0 && C==0) /* 0x = 0 */
          printf("Tout   réel   est   une   solution   de   cette
équation.\n");
 else if (A==0 && B==0)  /* Contradiction: c # 0 et c = 0 */
          printf("Cette équation ne possède pas de solutions.\n");
 else if (A==0) /* bx + c = 0 */
    {
     printf("La  solution  de  cette  équation  du  premier  degré  est
:\n");
     printf(" x = %.4f\n", (double)C/B);
    }
 else if (D<0)  /* b^2-4ac < 0 */
     printf("Cette équation n'a pas de solutions réelles.\n");
 else if (D==0) /* b^2-4ac = 0 */
    {
     printf("Cette équation a une seule solution réelle :\n");
     printf(" x =  %.4f\n", (double)-B/(2*A));
    }
 else /* b^2-4ac > 0 */
    {
     printf("Les solutions réelles de cette équation sont :\n");
     printf(" x1 = %.4f\n", (-B+sqrt(D))/(2*A));
     printf(" x2 = %.4f\n", (-B-sqrt(D))/(2*A));
    }
  return 0;
}
```

Exercice 7

```
#include <stdio.h>
#include <conio.h>
void main()
{
      int annee ;
      printf("donner une année:");
      scanf("%d",&annee);
      if((annee%4==0 && annee%100!=0)|| (annee%400)==0)
            printf("%d est une année bissextile!! \n");
      else
            printf ("%d n'est pas une année bissextile !! \n") ;
      getch() ;
}
```

Exercice 8

```
#include <stdio.h>
#include <conio.h>
void main()
{
      int jour, mois, annee;
      printf("introduire la date:");
      scanf("%d%d%d",&jour,&mois,&annee);
```

```
      printf(" \nLa date est  %d ",jour) ;
      switch(mois)
      {case 1 : printf(" Janvier ") ; break ;
       case 2 : printf(" février ") ; break ;
       case 3 : printf(" Mars ") ; break ;
       case 4 : printf(" Avril ") ; break ;
       case 5 : printf(" Mai ") ; break ;
       case 6 : printf(" Juin ") ; break ;
       case 7 : printf(" Juillet ") ; break ;
       case 8 : printf(" Aout ") ; break ;
       case 9 : printf(" Septembre ") ; break ;
       case 10 : printf(" octobre ") ; break ;
       case 11 : printf(" Novembre ") ; break ;
       case 12 : printf(" Décembre ") ; break ;
       default : printf("ERREUR !! ") ;
      }
```

Exercice 9

```
#include <stdio.h>
#include <conio.h>
void main()
{
      int jj, mm, aa;
      printf("introduire la date du jour: ") ;
      scanf("%d%d%d" ,&jj,&mm,&aa) ;
      switch(mm)
      {case 1 : nbj=31 ; break ;
        case 2 : {if((aa%4==0 && aa%100 !=0) ||(aa%400==0))
                  nbj=29;
                 else
                  nbj=28;
                }break;
       case 3 : nbj =31; break;
       case 4 : nbj=30 ; break ;
       case 5 : nbj=31 ; break ;
       case 6 : nbj=3 ; break ;
       case 7 : nbj=31 ; break ;
       case 8 : nbj=31 ; break ;
       case 9 : nbj=30 ; break ;
       case 10 : nbj=31 ; break ;
       case 11 : nbj=30 ; break ;
       case 12 : nbj=31 ; break ;
       default : printf(« ERREUR !! ») ;
      }
if(jj==nbj)
{   if(mm==12)
      { jj=1;
         mm=1;
         aa++;
       }
     else
       {  jj=1;
          mm++;
        }
}
else
```

```
     jj++;
if(jj>31 || mm>12)
     printf("ERREUR!!");
else     printf("\nLa date du lendemain est %d \ % \ %d",jj,mm,aa);
getch() ;
}
```

Exercice 10

```
#include <stdio.h>
#include <conio.h>
void main()
{
     int jour;
     printf("introduire le jour de la semaine :");
     scanf(«%d »,&jour) ;
     switch(jour)
     {case 1 : printf(« On se repose ») ; break ;
      case 2 : printf(« Il y a cours ») ; break ;
      case 3 : printf(« Il y a cours ») ; break ;
      case 4 : printf(« Il y a cours ») ; break ;
      case 5 : printf(« Il y a cours ») ; break ;
      case 6 : printf(« Il y a cours ») ; break ;
      case 7 : printf(« Il y a devoir surveillé ») ; break ;
      default : printf(« ERREUR ») ;
     }
     getch( );
}
```

Exercice 11

```
#include <stdio.h>
#include <conio.h>
void main()
{
       float moy;
       printf("introduire la moyenne: ") ;
       scanf("%f" ,&moy) ;
       if(moy>=16)
              printf(« Mention Très bien ») ;
       else if(moy>=14)
              printf(« Mention Bien ») ;
             else  if(moy>=12)
                     printf(« Mention Assez bien ») ;
                        else (moy>=10)
                     printf(« Passable ») ;
       getch() ;
}
```

Exercice 12

```
#include <stdio.h>
#include <conio.h>
void main()
{
       int taille, poids;
       float PI,BMI ;
       char s ;
```

```
printf("donner le sexe de la personne: ") ;
scanf("%c" ,&s) ;
printf("donner la taille de la personne: ") ;
scanf("%d" ,&taille) ;
printf("donner le poids de la personne: ") ;
scanf("%d" ,&poids) ;
if(s=='H')
       PI= (taille-100) - (taille-150)/4 ;
else
       PI= (taille-100) - (taille-150)/2 ;
printf(« le poids idéal de la personne est : %f »,PI) ;
BMI = poids/(taille/100) * 2 ;
if(BMI<27)
       printf(« Personne normale ») ;
else if(BMI<32)
       printf(« Personne obèse ») ;
else
       printf(« Personne malade ») ;
getch() ;
}
```

Correction TP N° 4

Exercice 1

```
#include<stdio.h>
int main ()
{
     int N, i ;
     printf("Entrer un nombre \n");
     scanf("%d",&N);
    for(i=1 ; i<=10 ; i++)
        printf("\n %d * %d = %d",N,i,N*i);
    return 0;
}
```

Exercice 2

a) en utilisant while,

```
#include <stdio.h>
main()
{
 int N;        /* nombre de données */
 int NOMB;     /* nombre courant    */
 int I;        /* compteur */
 long SOM;     /* la somme   des nombres entrés */
 double PROD; /* le produit des nombres entrés */
printf("Nombre de données : ");
 scanf("%d", &N);
SOM=0;
 PROD=1;
 I=1;
 while(I<=N)
    {
     printf("%d. nombre : ", I);
     scanf("%d", &NOMB);
     SOM  += NOMB;
     PROD *= NOMB;
     I++;
    }
 printf("La somme   des %d nombres est %ld \n", N, SOM);
 printf("Le produit des %d nombres est %.0f\n", N, PROD);
 printf("La moyenne des %d nombres est %.4f\n", N, (float)SOM/N);
 return 0;
}
```

b) en utilisant do – while : Remplacez le bloc de traitement (en gras) de (a) par :

```
 SOM=0;
 PROD=1;
 I=1;
 do
    {
     printf("%d. nombre : ", I);
     scanf("%d", &NOMB);
```

```
     SOM  += NOMB;
     PROD *= NOMB;
     I++;
    }
 while(I<=N);
```

c) en utilisant for : Remplacez le bloc de traitement (en gras) de (a) par :

```
 for (SOM=0, PROD=1, I=1 ; I<=N ; I++)
    {
     printf("%d. nombre : ", I);
     scanf("%d", &NOMB);
     SOM  += NOMB;
     PROD *= NOMB;
    }
```

Exercice 3

```
#include<stdio.h>
int main ()
{
     int x, y, i , M=0 ;
     printf("Entrer deux entiers : \n");
     scanf("%d%d",&x,&y);
    for(i=1 ; i<=y ; i++)
        M += x ;
   printf("\n La multiplication de  %d * %d = %d",x,y,M);
    return 0;
}
```

Exercice 4

```
#include<stdio.h>
int main ()
{
     int x, Max, Min, S, i ;
     float M ;
     printf("\n donner un entier : ");
     scanf("%d",&x) ;
     Max = x ; Min = x ;
     for(i=1 ; i<20 ; i++)
     {  printf("\n donner un entier : ");
        scanf("%d",&x) ;
        S += x ;
        if( Max < x)
            Max = x ;
        if(Min > x)
            Min = x ;
    }
    M = (float) S / 20 ;
    printf("\n La somme est %d \n La moyenne est %f \n Max = %d \n
Min = %d .", S, M, Max, Min);
    return 0;
}
```

Exercice 5

```
#include<stdio.h>
int main ()
```

```
{
     int i, N ;
     printf("Entrer un entier : \n");
     scanf("%d",&N);
     for(i=0 ; i<=N ; i++)
       printf("\n %d * %d = %d ",N,i,N*i);
     return 0;
}
```

Exercice 6

```
#include<stdio.h>
int main ()
{
     int i, N, F ;
     printf("Entrer un entier : \n");
    scanf("%d",&N);
    F = 1 ;
    for(i=1 ; i<=N ; i++)
       F *= i ;
    printf("\n Le factoriel de %d est égale à %d ",N, F);
    return 0;
}
```

Exercice 7

```
#include <stdio.h>
main()
{
 int LIG;    /* nombre de lignes         */
 int L;      /* compteur des lignes      */
 int ESP;    /* nombre d'espaces         */
 int I;      /* compteur des caractères */
 do
   {
     printf("Nombres de lignes : ");
    scanf("%d", &LIG);
   }
 while (LIG<1 || LIG>20);
 for (L=0 ; L<LIG ; L++)
   {
    ESP = LIG-L-1;
    for (I=0 ; I<ESP ; I++)
        putchar(' ');
    for (I=0 ; I<2*L+1  ; I++)
        putchar('*');
    putchar('\n');
   }
  return 0;
}
```

Exercice 8

```
#include <stdio.h>
main()
{
 int N, X, I, somme=0 ;
```

```
 printf ("\n Les entiers compris entre 1 et 100 (somme des
chiffres=11) : ");
 for(I=1 ;I<=100 ;I++)
   {    X=I ;
        do {
            somme+=X%10;
            X/=10;
        }while (X!=0);
        if(somme==11)
               printf ("\n %d ",I);
   }
  }
```

Exercice 9

```
#include <stdio.h>
main()
{
 int N, X;      /* entier N donné */
 int I, Max=0, Min=0,Np=0,Nn=0,NbPair=0,NbImp=0;
 do
   {
    printf ("Donner un nombre N : ");
    scanf ("%d", &N);
   }
 while (N<1);

 for (I=1 ; I<=N ; I++)
 {      printf ("Donner un entier X : ");
        scanf ("%d", &X);
        if(X > 0)
            Np++ ;
        else if(X<0)
            Nn++ ;
        if(X %2== 0)
            NbPair++ ;
        else
            NbImp++ ;
        if(X>Max)
            Max=X ;
        if(X<Min)1
            Min=X ;
 }
 printf("\n Le nombre des éléments positifs %d", Np);
 printf("\n Le nombre des éléments négatifs %d", Nn);
 printf("\n Le nombre des éléments pairs %d", NbPair);
 printf("\n Le nombre des éléments impairs %d", NbImp);
 printf("\n Le maximum est %d", Max);
 printf("\n Le minimum est %d", Min);

}
```

Exercice 10

```
#include <stdio.h>
main()
{
 int N;      /* entier N */
```

```
 int I;     /* compteur pour la boucle */
 int SomDiv; /* somme des diviseurs.   */
 do
   {
    printf ("Donner un nombre N : ");
    scanf ("%d", &N);
   }
 while (N<1);
 SomDiv = 0 ;
 for (I=1 ; I<=N ; I++)
 {     if(N%I==0)
           SomDiv+=I ;
 }
 if(SomDiv==N)
       printf("\n %d est un nombre parfait.", N);
 else
       printf("\n %d n'est pas un nombre parfait.", N);
}
```

Exercice 11

```
#include <stdio.h>
main()
{
 int N;     /* nombre de termes à calculer */
 int I;     /* compteur pour la boucle */
 float SOM; /* Type float à cause de la précision du résultat.   */
 do
   {
     printf ("Nombre de termes: ");
    scanf ("%d", &N);
   }
 while (N<1);
 for (SOM=0.0, I=1 ; I<=N ; I++)
    SOM += (float)1/I;
 printf("La somme des %d premiers termes est %f \n", N, SOM);
 return 0;
}
```

Exercice 12

```
#include <stdio.h>
main()
{
 int U1, U2, UN;    /* pour parcourir la suite */
 int N;             /* rang du terme demandé   */
 int I;             /* compteur pour la boucle */
 do
   {
     printf("Rang du terme demandé : ");
    scanf("%d", &N);
   }
 while(N<1);

 U1=1; /* Initialisation des deux premiers termes */
 U2=2 ;
```

```
 if (N==1)
      UN=U1;
 else if (N==2)
      UN=U2;
 else
      {
       for (I=3 ; I<=N ; I++)
           {
            UN = U1+U2;
            U1 = U2;
            U2 = UN;
           }
      }
  printf("Valeur du terme de rang %d : %d\n", N, UN);
 return 0;
}
```

Correction TP N° 5

Exercice 1

Ecrire un programme C qui permet de saisir 10 entiers dans un tableau Tab puis affiche les entiers positifs ensuite les entiers négatifs.

Exercice 2

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int T[50]; /* tableau donné */
 int N;     /* dimension     */
 int I,J;   /* indices courants */
 int AIDE;  /* pour l'échange   */
 int SOM;  /* somme des éléments */
 int PMIN, PMAX;   /* position du minimum et du maximum */

 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &T[I]);
    }
 /* Affichage du tableau */
 printf("Tableau donné : \n");
 for (I=0; I<N; I++)
     printf("%d ", T[I]);
 printf("\n");

/* Calcul de la somme */
 for (SOM=0, I=0; I<N; I++)
     SOM += T[I];
  /* Edition du résultat */
 printf("Somme de éléments : %ld\n", SOM);

/* Recherche du maximum et du minimum */
 PMIN=0;
 PMAX=0;
 for (I=1; I<N; I++)
     {
      if(A[I]>A[PMAX]) PMAX=I;
      if(A[I]<A[PMIN]) PMIN=I;
     }
  /* Edition du résultat */
 printf("Position du minimum : %d\n", PMIN);
```

```
 printf("Position du maximum : %d\n", PMAX);
 printf("Valeur   du minimum : %d\n", A[PMIN]);
 printf("Valeur   du maximum : %d\n", A[PMAX]);

 /* Inverser le tableau */
 for (I=0, J=N-1 ; I<J ; I++,J--)
      /* Echange de T[I] et T[J] */
        {
         AIDE = T[I];
         T[I] = T[J];
         T[J] = AIDE;
        }
  /* Edition des résultats */
 printf("Tableau résultat :\n");
 for (I=0; I<N; I++)
     printf("%d ", T[I]);
 printf("\n");

/* Initialisation des dimensions de TPOS et TNEG */
 NPOS=0;
 NNEG=0;
 /* Transfer des données vers TPOS et TNEG */
 for (I=0; I<N; I++)
      { if (T[I]>0) {
                     TPOS[NPOS]=T[I];
                     NPOS++;
                    }
        if (T[I]<0) {
                     TNEG[NNEG]=T[I];
                     NNEG++;
                    }
      }
  /* Edition du résultat */
 printf("Tableau TPOS :\n");
 for (I=0; I<NPOS; I++)
     printf("%d ", TPOS[I]);
 printf("\n");
 printf("Tableau TNEG :\n");
 for (I=0; I<NNEG; I++)
     printf("%d ", TNEG[I]);

 return 0;
}
```

Exercice 3

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50]; /* tableau donné */
 int VAL;   /* valeur à rechercher   */
 int N;     /* dimension      */
 int I;     /* indice courant */

 /* Saisie des données */
```

```
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &A[I]);
    }
 printf("Elément à rechercher : ");
 scanf("%d", &VAL );
 /* Affichage du tableau */
 printf("Tableau donné : \n");
 for (I=0; I<N; I++)
     printf("%d ", A[I]);
 printf("\n");
 /* Recherche de la position de la valeur */
I=0 ;
While((I<N)&& (A[I]!=VAL))
           I++;

/* Edition du résultat */
 if (A[I]==VAL)
     printf("La valeur %d se trouve à la position %d. \n",VAL, I);
  else
      printf("La valeur recherchée ne se trouve pas dans le
tableau.\n");
 return 0;
}
```

Exercice 4

```
#include <stdio.h>
main()
{
 /* Déclarations */
 /* Les tableaux et leurs dimensions */
 int A[50], B[50], C[100];
 int N ;
 int IA, IB, IC; /* indices courants */

 /* Saisie des données */
 printf("Dimension du tableau A et B (max.50) : ");
 scanf("%d", &N );
 printf("Entrer les éléments de A dans l'ordre croissant :\n");
 for (IA=0; IA<N; IA++)
    {
     printf("Elément A[%d] : ", IA);
     scanf("%d", &A[IA]);
    }
 printf("Entrer les éléments de B dans l'ordre croissant :\n");
 for (IB=0; IB<N; IB++)
    {
     printf("Elément B[%d] : ", IB);
     scanf("%d", &B[IB]);
    }
 /* Affichage des tableaux A et B */
 printf("Tableau A :\n");
 for (IA=0; IA<N; IA++)
```

```
    printf("%d ", A[IA]);
 printf("\n");
 printf("Tableau B :\n");
 for (IB=0; IB<N; IB++)
    printf("%d ", B[IB]);
 printf("\n");

/* Fusion des éléments de A et B dans C */
 IA=0; IB=0; IC=0;
 while ((IA<N) && (IB<M))
        if(A[IA]<B[IB])
            {
             C[C]=A[IA];
             IC++;
             IA++;
            }
        else
            {
             C[IC]=B[IB];
             IC++;
             IB++;
            }
 /* Si IA ou IB sont arrivés à la fin de leur tableau, */
 /* alors copier le reste de l'autre tableau.          */
 while (IA<N)
        {
         C[IC]=A[IA];
         IC++;
         IA++;
        }
 while (IB<M)
        {
         C[IC]=B[IB];
         IC++;
         IB++;
        }
  /* Edition du résultat */
 printf("Tableau C :\n");
 for (IC=0; IC<2*N; IC++)
    printf("%d ", C[IC]);
 printf("\n");
 return 0;
}
```

Exercice 5

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int M[50][50]; /* matrice carrée   */
 int L, C;         /* dimensions de la matrice */
 int I, J;      /* indices courants */
 int S = 0 ;

 /* Saisie des données */
```

```
 printf("Dimensions de la matrice L et C (max.50) : ");
 scanf("%d%d", &L,&C);
 for (I=0; I<L; I++)
    for (J=0; J<C; J++)
        {
         printf("Elément[%d][%d] : ",I,J);
         scanf("%d", &M[I][J]);
         S += M[I][J] ;
        }
 /* Affichage de la matrice */
 printf("Matrice donnée :\n");
 for (I=0; I<L; I++)
    {
     for (J=0; J<C; J++)
          printf("%d ", M[I][J]);
     printf("\n");
    }

 /* Affichage de la somme de la matrice */
 printf("\n La somme de la matrice est %d ",S);
    }
 return 0;
}
```

Exercice 6

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int U[50], V[50]; /* tableaux donnés */
 int N;     /* dimension        */
 int I;     /* indice courant   */
 long PS;   /* produit scalaire */
 /* Saisie des données */
 printf("Dimension des tableaux (max.50) : ");
 scanf("%d", &N );
 printf("** Premier tableau **\n");
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &U[I]);
    }
 printf("** Deuxième tableau **\n");
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &V[I]);
    }
 /* Calcul du produit scalaire */
 for (PS=0, I=0; I<N; I++)
      PS += (long)U[I]*V[I];
  /* Edition du résultat */
 printf("Produit scalaire : %ld\n", PS);
 return 0;
}
```

Exercice 7

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int M[10][10]; /* déclaration de la matrice   */
 int V[100] ;   /* déclaration du tableau   */
 int L, C;      /* dimensions de la matrice */
 int I, J, k;      /* indices courants */

 /* Saisie des données */
 printf("Dimensions de la matrice L et C (max.50) : ");
 scanf("%d%d", &L,&C);
 k = 0 ;
 for (I=0; I<L; I++)
    for (J=0; J<C; J++)
        {
         printf("Elément[%d][%d] : ",I,J);
         scanf("%d", &M[I][J]);
         V[k] = M[I][J] ;
         K++ ;
        }
 /* Affichage du vecteur */
 printf("Vecteur résultat :\n");
 for (I=0; I<L*C; I++)
    printf(" %d ",V[I]);
return 0;
}
```

Exercice 8

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int M[10][10]; /* matrice carrée   */
 int N;         /* dimension de la matrice carrée */
 int I, J;      /* indices courants */
 int x , Nbocc = 0; /* entier à saisir et nombre d'occurrence */

 /* Saisie des données */
 printf("Dimension de la matrice carrée (max.10) : ");
 scanf("%d", &N);
 for (I=0; I<N; I++)
    for (J=0; J<N; J++)
        {
         printf("Elément[%d][%d] : ",I,J);
         scanf("%d", &M[I][J]);
        }

 /* Donner l'entier à rechercher */
 printf("Donner un entier : ");
 scanf("%d", &x);
```

```
 /* Compter le nombre d'occurrence */
 for (I=0; I<N; I++)
    {
     for (J=0; J<N; J++)
          if(M[I][J]==x)
              Nbocc ++ ;
    }
 /* Affichage du résultat */
 printf("\n Le nombre d'occurrence de %d est %d ", x, Nbocc);
 return 0;
}
```

Exercice 9

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50]; /* tableau donné */
 int N;     /* dimension     */
 int I;     /* rang à partir duquel A n'est pas trié */
 int J;     /* indice courant      */
 int AIDE;  /* pour la permutation */
 int PMAX;  /* indique la position de l'élément */
            /* maximal à droite de A[I]         */

 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (J=0; J<N; J++)
    {
     printf("Elément %d : ", J);
     scanf("%d", &A[J]);
    }
 /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (J=0; J<N; J++)
    printf("%d ", A[J]);
 printf("\n");

/* Tri du tableau par sélection directe du maximum. */
 for (I=0; I<N-1; I++)
     {
      /* Recherche du maximum à droite de A[I] */
      PMAX=I;
      for (J=I+1; J<N; J++)
          if (A[J]>A[PMAX]) PMAX=J;
      /* Echange de A[I] avec le maximum */
      AIDE=A[I];
      A[I]=A[PMAX];
      A[PMAX]=AIDE;
     }

  /* Edition du résultat */
 printf("Tableau trié :\n");
 for (J=0; J<N; J++)
     printf("%d ", A[J]);
```

```
 printf("\n");
 return 0;
}
```

Exercice 10

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int T[50]; /* tableau donné */
 int N;     /* dimension     */
 int I,J;   /* indices courants */
 int MIN, MAX;   /* minimum et maximum */
 int choix ;  /* choix du traitement à effectuer */
 int VAL,X ;  /* valeur à rechercher et valeur à supprimer */
 int Y ;  /* valeur à choisir pour continuer le traitement */

 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &T[I]);
    }
/* Affichage du Menu */
Do{
Printf("\n\n ******** MENU ***************") ;
Printf("\n 1 : Affichage du tableau ") ;
Printf("\n 2 : Rechercher l'élément minimum") ;
Printf("\n 3 : Rechercher l'élément maximum") ;
Printf("\n 4 : Rechercher un élément ") ;
Printf("\n 5 : Supprimer un élément") ;
Printf("\n 6 : Quitter le programme") ;

Printf("\n Choisissez le traitement à effectuer : ") ;
Scanf("%d",&choix) ;

Switch(choix)
{   case 1: /* Affichage du tableau */
          printf("Tableau donné : \n");
          for (I=0; I<N; I++)
             printf("%d ", T[I]);
          printf("\n");
          break;

   Case 2: /*recherche du maximum */
         MAX=T[0];
         for (I=1; I<N; I++)
         {     if(T[I]> MAX)
                  MAX=T[I];
         }
         /* Edition du résultat */
         printf("Valeur   du maximum : %d\n", MAX);
```

```
        break ;
    Case 3: /*recherche du minimum */
        MIN=T[0];
        for (I=1; I<N; I++)
        {     if(T[I]<MIN)
                 MIN=T[I];
        }
        /* Edition du résultat */
        printf("Valeur   du minimum : %d\n", MIN);
        break ;

   Case 4:/* Recherche d'un element*/
        printf("Elément à rechercher : ");
        scanf("%d", &VAL );
        I=0 ;
        While((I<N)&& (A[I]!=VAL))
          I++;

        /* Edition du résultat */
         if (A[I]==VAL)
             printf("La valeur %d se trouve à la position %d.
\n",VAL, I);
         else
             printf("La valeur recherchée ne se trouve pas dans le
tableau.\n");
         break ;

   case 5:/*Suppression d'un element */
          printf("Elément à supprimer : ");
          scanf("%d", &X);
         for (I=0; I<N; I++)
         {     if(T[I]== X)
            {for (J=I; J<N; J++)
                 T[J]=T[J+1];
             }
          }
          printf("\n Suppression terminée avec sucées ") ;
          break ;

   case 6: printf("Fin du programme !!! ");
         break;

   default: printf("ERRREUR !! ") ;
}
printf("\n Voulez-vous effectuer un autre traitement (1 si
oui) : ") ;
scanf("%d ",&Y) ;
}while(y ==1) ;

return 0 ;
}
```

Correction TP N° 6

Exercise 2

```
#include <stdio.h>
main()
{ /* Déclarations */
 char CH[20]; /* chaîne donnée       */
 int I;  /* indice courant         */
 int L = 0 ;    /* longueur de la chaîne    */

 /* Saisie des données */
 printf("Entrez une chaine (max.20 caractères) :\n");
 gets(TXT);

 /* a) Compter les caractères */
 for (I=0; TXT[I]!= '\0'; I++)
      L++ ;
 return 0;
}
```

Exercice 3

```
#include <stdio.h>
main()
{ /* Déclarations */
 char S1[20], S2[20]; /* chaînes données     */
 char T[40] ;      /* chaine résultat        */
 int I, J, K;        /* indices courants        */
 int L1, L2 ;    /* longueurs des chaînes   */

 /* Saisie des données */
 printf("Entrez une chaine S1 (max.20 caractères) :\n");
 gets(S1);
 printf("Entrez une deuxième chaine S2 (max.20 caractères)n");
 gets(S2);

 /* a) Compter les caractères */
 L1 = strlen(S1) ;
 L2 = strlen(S2) ;
 /* Concaténation de deux chaines */
 for(I=0 ; I<L1 ; I++)
      T[I]=S1[I] ;
 for(J=I,K=0 ; K<=L2 ; K++)
 {
      T[J]=S2[K] ;
      J++ ;
 }
```

```
/* Affichage du chaine résultat  */
 printf("\nLa chaine concaténée est %s .\n",T);

 return 0;
}
```

Exercice 4

```
#include <stdio.h>
#include <string.h>
main()
{ /* Déclarations */
 char TXT[201]; /* chaîne donnée       */
 int I,J;  /* indices courants         */
 int L;    /* longueur de la chaîne    */
 int C;    /* compteur des lettres 'e' */
 int AIDE; /* pour l'échange des caractères */

 /* Saisie des données */
 printf("Entrez une ligne de texte (max.200 caractères) :\n");
 gets(TXT); /* L'utilisation de scanf est impossible pour */
 /* lire une phrase contenant un nombre variable de mots. */

 /* a) Compter les caractères */
 L=strlen(TXT) ;
 printf("\nLe texte est composé de %d caractères.\n",L);

 /* b) Compter les lettres 'e' dans le texte */
 C=0;
 for (I=0; I<L I++)
     if (TXT[I]=='a') C++;
 printf("Le texte contient %d lettres \'a\'.\n",C);

 /* c) Afficher la phrase à l'envers */
 for (I=L-1; I>=0; I--)
     putchar(TXT[I]);  /* ou printf("%c",TXT[I]); */
 putchar('\n');        /* ou printf("\n"); */

 /* d) Inverser l'ordre des caractères */
 for (I=0,J=L-1 ; I<J ; I++,J--)
    {
     AIDE=TXT[I];
     TXT[I]=TXT[J];
     TXT[J]=AIDE;
    }
 puts(TXT);  /* ou printf("%s\n",TXT); */
  return 0;
}
```

Exercice 5

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

```
#include <string.h>
main()
{ /* Déclarations */
 char CH[201]; /* chaîne donnée       */
 int I,J;  /* indices courants         */
 int L;    /* longueur de la chaîne    */
 int NB;    /* compteur des nombre des mots */

 /* Saisie des données */
 printf("Entrez une ligne de texte (max.200 caractères) :\n");
 gets(CH); /* L'utilisation de scanf est impossible pour */
 /* lire une phrase contenant un nombre variable de mots. */
 /* a) Compter les caractères */
 L=strlen(CH) ;
 printf("\nLe texte est composé de %d caractères.\n",L);

 /* b) Compter le nombre des mots dans le texte */
 NB=0;
 for (I=0; I<L I++)
  {   if (isspace(CH[I]!=0)
         NB++;
   }
 printf("Le texte contient %d mots \'a\'.\n",NB);
return 0;
}
```

Exercice 6

```
#include <stdio.h>
main()
{ /* Déclarations */
 char TXT[201]; /* chaîne donnée    */
 int  I,J;      /* indices courants */
                        int L;    /* longueur de la chaîne    */
 /* Saisie des données */
 printf("Entrez une ligne de texte (max.200 caractères) :\n");
 gets(TXT);
                                  /* a) Compter les caractères */
                                              L=strlen(TXT) ;
 /* Eliminer les lettres 'e' et comprimer :  */
 /* Copier les caractères de I vers J et incrémenter J */
 /* seulement pour les caractères différents de 'e'.   */
 for (J=0,I=0 ; I<L ; I++)
    {
     TXT[J] = TXT[I];
     if (TXT[I] != 'e')
         J++;
    }
 /* Terminer la chaîne !! */
 TXT[J]='\0';
  /* Edition du résultat */
 puts(TXT);
  return 0;
}
```

Exercice 7

```
#include <stdio.h>
#include <string.h>
 main()
{ /* Déclarations */
 char VERB[20]; /* chaîne contenant le verbe */
 char AFFI[30]; /* chaîne pour l'affichage   */
 int L;         /* longueur de la chaîne     */
 /* Saisie des données */
/* Contrôler s'il s'agit d'un verbe en 'er' */
do
{
 printf("Verbe : ");
 gets(VERB);
 L=strlen(VERB);
}while((VERB[L-2]!='e') || (VERB[L-1]!='r'));

      /* Couper la terminaison 'er'. */
      VERB[L-2]='\0';
      /* Conjuguer ... */
      AFFI[0]='\0';
      strcat(AFFI, "je ");
      strcat(AFFI, VERB);
      strcat(AFFI, "e");
      puts(AFFI);
      /**********/
      /* Couper la terminaison 'er'. */
      VERB[L-2]='\0';
      ……..……..

      AFFI[0]='\0';
      strcat(AFFI, "ils ");
      strcat(AFFI, VERB);
      strcat(AFFI, "ent");
      puts(AFFI);
     }
  return 0;
}
```

Exercice 8

```
#include <stdio.h>
main()
{
 /* Déclarations */
 char CH[101]; /* chaîne donnée */
 int ABC[26];  /* compteurs des différents caractères */
  int i,j ;    /* pointeur d'aide dans ABC */

 /* Saisie des données */
 printf("Entrez une ligne de texte (max.100 caractères) :\n");
 gets(CH);

 /* Initialiser le tableau ABC */
 for (i=0; i<26; i++)
```

```
    ABC[i] =0;
 /* Compter les lettres */
 L=strlen(CH) ;
 for (j=0; j<L; j++)
     {
      if (CH[j]=='A' && CH[j]<='Z')
         (ABC[i+(CH[j]-'A')])++; /* Attention aux parenthèses! */
      if (CH[j]>='a' && CH[j]<='z')
         (*(ABC[i+(CH[j]-'a')])++;
     }
 /* Affichage des résultats */
 /* (PABC-ABC) est le numéro de la lettre de l'alphabet. */
 printf("La chaîne \"%s\" contient :\n", CH);
 for (i=0; i<26; i++)
     if (ABC[i])
        printf(" %d\tfois la lettre '%c' \n",
                                          ABC[i], 'A'+i);
  return 0;
}
```

Exercice 9

```
#include <stdio.h>
main()
{ /* Déclarations */
 char CH1[100]; /* chaîne à transformer        */
 char CH2[100]; /* chaîne à supprimer dans CH1 */
 int I;         /* indice courant dans CH1     */
 int J;         /* indice courant dans CH2     */
 int TROUVE;    /* indicateur logique qui précise */
                /* si la chaîne CH2 a été trouvée */

 /* Saisie des données */
 printf("Introduisez la chaîne à supprimer   : ");
 gets(CH2);
 L2=strlen(CH2) ;
 printf("Introduisez la chaîne à transformer : ");
 gets(CH1);
 L1=strlen(CH1) ;
 /* Recherche de CH2 dans CH1 */
 TROUVE=0;
 for (I=0; I<L1 && !TROUVE; I++)
    /* Si la première lettre est identique, */
    if (CH1[I]==CH2[0])
        { /* alors comparer le reste de la chaîne */
         for (J=1; J<L2 && (CH2[J]==CH1[I+J]); J++)
            ;
         if (CH2[J]=='\0')
              TROUVE=1;
        }
 /* Si la position de départ de CH2 dans CH1 a été trouvée */
 /* alors déplacer le reste de CH1 à cette position. */
 if (TROUVE)
    { I--;
     /* Maintenant  I indique la position de CH2 */
     /* dans CH1 et J indique la longueur de CH2 */
```

```
     for (I; CH1[I+J]; I++)
          CH1[I]=CH1[I+J];
     CH1[I]='\0';
    }
  /* Affichage du résultat */
 printf("Chaîne résultat : \"%s\"\n", CH1);
 return 0;
}
```

Exercice 10

```
#include <stdio.h>
main()
{
 /* Déclarations */
 char TABCH[5][50]; /* tableau pour les 5 mots */
 int I,J;              /* indices courants */
 char aux ;
/* Saisie des mots */
 printf("Entrez 5 mots, séparés par des espaces :\n");

 for (I=0; I<5; I++)
     scanf("%s",TABCH[I]);
 /* Inverser les mots  */
 for(I=0 ; I<5 ;I++)
 { L=strlen (TABCH[I]) ;
       for(J=0 ;J<L/2 ; J++)
       {      aux = TABCH[I][J] ;
              TABCH[I][J]= TABCH[I][L-J-1] ;
              TABCH[I][L-J-1] = aux ;
       }
/* Affichage des 5 mots */
 for (I=0; I<5; I++)
    printf("%s ", TABCH[I]);

 printf("\n");
 return 0;
}
```

Correction TP N° 7

Exercice 1

```
/* Définition de la fonction MAX */
float MAX(float N1, float N2)
{
 if (N1>N2)
     return N1;
 else
     return N2;
}

/* Définition de la fonction MIN */
float MIN(float N1, float N2)
{
 if (N1>N2)
     return N2;
 else
     return N1;
}

/* Définition de la fonction Impaire */
int Impaire(int N)
{
 if (N%2 == 0)
     return 1;
 else
     return 0;
}

/* Définition de la fonction Absolue */
int Absolue(int N)
{
 if (N > 0)
     return N;
 else
     return -N;
}
```

Exercice 2

```
float Puissance(float X, int N)
{
 float RES=1.0;
 int i ;
 for (i=0; i<N; i++)
     RES *= X;
 return RES;
}
```

Exercice 3

```
int Fact(int N)
{
 int F=1;
 int i ;
 for (i=1; i<=N; i++)
     F *= i;
 return F;
}
```

Exercice 4

```
float fonc(float X, int N)
{
 float S=0;
 int i ;
 for (i=0; i<N; i++)
     S+= (float)Puisance(X,i) / Fact(i) ;
 return S;
}
```

Exercice 5

```
int Position(int TAB[],int x, int N)
 {
  int I=0, pos=0;
  while (TAB[I] !=x && I<N)
       I++ ;
  if(TAB[I]==x)
       pos = I ;
  return pos;
 }
void Supprimer(int TAB[],int k, int N)
 {
  int I, pos=0;
  for (I=k ; I<N ; I++)
       TAB[I]=TAB[I+1] ;
 }
```

Exercice 7

```
void LIRE_TAB(int TAB[], int NMAX, int * N)
 {
  /* Variables locales */
  int I;
  /* Saisie de la dimension du tableau */
  do
    {
     printf("Dimension du tableau (max.%d) : ", NMAX);
     scanf("%d", N); /* Attention: écrire N et non &N ! */
    }
 while (*N<0 || *N>NMAX);
 /* Saisie des composantes du tableau */
 for (I=0; I<*N; I++)
    {
```

```
     printf("Elément[%d] : ", I);
     scanf("%d", TAB+I);
    }
}
void ECRIRE_TAB(int TAB[], int N)
 {
  int I;
  for (I=0; I<N; I++)
     printf("\n TAB[%d] = %d ", I, TAB[I]);
}

 int SOMME_TAB(int TAB[], int N)
 {
  int I, S=0;
  for (I=0; I<N; I++)
       S += TAB[I] ;
  return S ;
 }
```

Exercice 8

```
int strlen(char S[])
{
 int N =0, i ;
 for (i=0; S[i] != '\0'; i++)
         N++;
 return N;
}
```

Exercice 9

```
int Apparition(char CH[], char c)
{
 int Nb =0, i ;
 for (i=0; CH[i] != '\0'; i++)
 {
       If(CH[i]==c)
              Nb++;
 }
 return Nb;
}
```

Exercice 10

```
 int LONG_CH(char CH[])
 {
  int N =0, i ;
  for (i=0; CH[i] != '\0'; i++)
         N++;
  return N;
 }

 Char * AJOUTE_CH(char CH1[10], char CH2[10])
 { char CH[20] ;
  int Nb =0, i, j ;
  for (i=0; CH1[i] != '\0'; i++)
```

```
      Ch[i]=CH1[i] ;
 for (j=0; CH2[j] != '\0'; j++)
 {    Ch[i]=CH2[j] ;
      i++ ;
 }
 CH[i]='\0' ;
 return CH;
}

void PERMUTE_CH(char * CH1, char * CH2)
{
 Char c ;
 c = * CH1 ;
 *CH1 = *CH2 ;
 *CH2= c ;
}

void INVERSER_CH(char CH[])
{
 int i, j , L;
 L= LONG_CH(CH) ;
 for(i=0, j=L-1 ; i<L/2, j>L/2 ;i++,j--)
      PERMUTE_CH(CH[i],CH[j]) ;
}
```

Exercice 11

```
void SuppOcc(char CH1[],char CH2[])
{
  int I;          /* indice courant dans CH1     */
  int J;          /* indice courant dans CH2     */
  int TROUVE;     /* indicateur logique qui précise */
                /* si la chaîne CH2 a été trouvée */

/* Recherche de CH2 dans CH1 */
TROUVE=0;
for (I=0; CH1[I] && !TROUVE; I++)
   /* Si la première lettre est identique, */
   if (CH1[I]==CH2[0])
       {
        /* alors comparer le reste de la chaîne */
        for (J=1; CH2[J] && (CH2[J]==CH1[I+J]); J++)
           ;
        if (CH2[J]=='\0') TROUVE=1;
       }
/* Si la position de départ de CH2 dans CH1 a été trouvée */
/* alors déplacer le reste de CH1 à cette position. */
if (TROUVE)
   {
    I--;
    /* Maintenant  I indique la position de CH2 */
    /* dans CH1 et J indique la longueur de CH2 */
    for (; CH1[I+J]; I++)
         CH1[I]=CH1[I+J];
    CH1[I]='\0';
   }
```

```
   /* Affichage du résultat */
   printf("Chaîne résultat : \"%s\"\n", CH1);
 }
```

Exercice 12

```
void SaisirComptes(int TabCpt[], float TabSoldes[], int N)
 {
  int i, j ;
  for(i=0; i<N ;i++)
  {
       printf("\nIntroduire le numéro de compte: ");
       scanf("%d",&TabCpt[i]);
       printf("\nIntroduire le solde de compte: ");
       scanf("%f",&TabSoldes[i]);
   }
 }

void AffichComptes(int TabCpt[], float TabSoldes[], int N)
 {
  int i, j ;
  for(i=0; i<N ;i++)
  {
       printf("\n Numéro de compte: %d",TabCpt[i]);
       printf("\t Solde de compte: %f",TabSoldes[i]);
    }
 }

int Rechercher(int TabCpt[], int N, int num)
 {
  int i, pos =-1 ;
  i=0;
  while(TabCpt[i] !=num && i<N)
   {
       i++ ;
   }
   if(TabCpt[i]==num)
       pos =i ;
   return pos ;
 }

 void Ajouter(int TabCpt[], float TabSoldes[],int num,float s,int *N)
 {
  TabCpt[*N]=num ;
  TabSoldes[*N]=s ;
  (*N)++ ;
 }

 void Deposer(int TabCpt[],float TabSoldes[],int num,float montant
,int N)
 {
  int i =0 ;
  while(TabCpt[i] != num  && i<N)
       i++ ;
  if(TabCpt[i]==num)
       TabSoldes[i] += montant ;
```

```
  else
       printf("\n ERREUR ! Compte Inexixtant !! ") ;
 }

 void Retirer(int TabCpt[],float TabSoldes[],int num,float montant
,int N)
 {
  int i =0 ;
  while(TabCpt[i] != num  && i<N)
       i++ ;
  if(TabCpt[i]==num)
       if(TabSoldes[i]>=montant)
              TabSoldes[i] -= montant ;
       else
              printf("\n ERREUR ! Solde Insuffisant !! ") ;
  else
       printf("\n ERREUR ! Compte Inexixtant !! ") ;
 }
```

Correction TP N° 8

Exercice 1

```
#include <stdio.h>
main()
{
 struct Date
 {     int jour ;
       int mois ;
       int annee ;
 } ;
 struct Personne
 {     char Nom[10] ;
       char Prenom[10] ;
       struct Date DN ;
       char Matricule[20] ;
 } ;
 struct Personne P ;
 strcpy(P.Nom, "Amer") ;
 strcpy(P.Prenom, "Salem") ;
 P.DN.jour=3 ;
 P.DN.mois=4 ;
 P.DN.annee=2005 ;
 strcpy(P.Matricule, "A32") ;
}
```

Exercice 2

```
1) struct Point
 {     float abs ;
       float ord ;
} ;
void Saisir(struct Point *P)
{     printf("\n Donner l'abscisse : ") ;
      scanf("%f",P.abs) ;
      printf("\n Donner l'ordonné : ") ;
      scanf("%f",P.ord) ;
}
void Afficher(struct Point P)
{     printf("\nle  point  est  d'abscisse :  %f  ,  et  d'ordonné
%f",P.abs,P.ord) ;
}
void Deplacer (struct Point *P, float dx, float dy)
{     *P.x += dx ;
      *P.y += dy ;
}
struct Point Milieu(struct Point P1, struct Point P2)
{     struct Point PM ;
      PM.x = (P1.x + P2.x)/2 ;
      PM.y = (P1.y + P2.y)/2 ;
      return PM ;
```

```
}
Main()
{
 Struct Point A ={6.5,2.3) ;
 Struct Point B ={-9,8.5) ;
 Deplacer(&B, 3.0, 8.0) ;
 M=Milieu(A,B) ;
 Afficher(M) ;
}
```

```
2)#include <stdio.h>
main()
{
 struct Date
 {     int jour ;
       int mois ;
       int annee ;
 } ;
 struct Personne
 {     char Nom[10] ;
       char Prenom[10] ;
       struct Date DN ;
       char Matricule[20] ;
 } ;
 struct Personne P ;
 strcpy(P.Nom, "Amer") ;
 strcpy(P.Prenom, "Salem") ;
 P.DN.jour=3 ;
 P.DN.mois=4 ;
 P.DN.annee=2005 ;
 strcpy(P.Matricule, "A32") ;
```

Exercice 3

```
struct Complexe
 {     float reel ;
       float img ;
} ;
void SaisirComplexe(struct Complexe *C)
{     printf("\n Donner la partie réelle : ") ;
      scanf("%f",C.reel) ;
      printf("\n Donner la partie imaginaire : ") ;
      scanf("%f",C.img) ;
}
struct Complexe SommeComplexe(struct Complexe C1,struct Complexe C2)
{     struct Complexe CS ;
      CS.reel = C1.reel+ C2.reel ;
      CS.img = C1.img + C2.img ;
      return CS ;
}
struct Complexe ProduitComplexe(struct Complexe C1,struct Complexe
C2)
{     struct Complexe CP ;
      CP.reel = C1.reel * C2.reel ;
```

```
      CP.img = C1.img * C2.img ;
      return CP ;
}
void AfficherComplexe(struct Complexe C)
{     printf("\n Z= %f + %f *i ", C.reel, C.img) ;
}

main()
{     struct Complexe Z1, Z2, Z3, Z4 ;
      SaisirComplexe(&Z1) ;
      SaisirComplexe(&Z2) ;
      Z3 = SommeComplexe(Z1, Z2) ;
      Z4 = ProduitComplexe(Z1, Z2) ;
      AfficherComplexe(Z3) ;
      AfficherComplexe(Z4) ;
}
```

Exercice 4

```
struct Horaire
 {      int heure ;
        Int min ;
        Int sec ;
} ;
struct Bus
{     int numéro ;
      char LieuDep[20] ;
      char LieuArr[20] ;
      struct Horaire HDep ;
      struct Horaire HArr ;
} ;
void SaisirBus(struct Bus Tbus[], int N)
{ int i ;
 for(i=0 ;i<N ;i++)
 {
      printf("\n Donner le numéro de bus: ") ;
      scanf("%d",Tbus[i].numero) ;
      printf("\n Donner le lieu de départ de bus: ") ;
      gets(Tbus[i].LieuDep) ;
      printf("\n Donner le lieu d'arrivée de bus: ") ;
      gets(Tbus[i].LieuArr) ;
      printf("\nDonner l'horaire de départ (heure/min/sec): ") ;
      scanf("%d",Tbus[i].HDep.heure) ;
      scanf("%d",Tbus[i].HDep.min) ;
      scanf("%d",Tbus[i].HDep.sec) ;
  }
}
void AffichBus(struct Bus Tbus[],char V1[],char V2[],struct Horaire
H1,struct Horaire H2,int N)
{ int i ;
  for(i=0 ;i<N ;i++)
  {   if(strcmp(Tbus[i].LieuDep,V1)==1                                  &&
strcmp(Tbus[i].LieuArr,V2)==1 )
           If(Tbus[i].HDep==H1 && Tbus[i].HArr==H2)
```

```
                printf("\n     Numéro     de     bus     trouvé:     %d
          ",Tbus[i].numero) ;

  }
}

void  AffichHeureArriv(struct  Bus  Tbus[],int  num,char  V1[],char
V2[],struct Horaire H1,int N)
{ int i ;
  for(i=0 ;i<N ;i++)
  {   if(Tbus[i].numero==num   &&   strcmp(Tbus[i].LieuDep,V1)==1   &&
strcmp(Tbus[i].LieuArr,V2)==1 )
          If(Tbus[i].HDep==H1)
               printf("\n L'horaire d'arrivée du bus trouvé: %d %d
          %d
          ",Tbus[i].HArr.heure,Tbus[i].HArr.min,Tbus[i].HArr.sec) ;

  }
}
main()
{     struct Bus Tbus[10] ;
      int i ;
      struct Horaire H1={12,15,0} ;
      struct Horaire H2={15,20,0} ;
      struct Horaire H3={15,30,0} ;
      SaisirBus(Tbus,10) ;
      AffchBus(Tbus,"Nabeul","Tunis",H1,H2,10) ;
      AffichHeureArriv(Tbus,"Sousse","Tunis",H3,10) ;
}
```

Exercice 5

```
struct Date
{     int jour ;
      int mois ;
      int annee ;
} ;
struct Sportif
{     char Nom[20] ;
      char Prenom[20] ;
      char Pays[20] ;
      struct Date DN ;
      int performance ;
} ;
void SaisiSport(struct Sportif TSport[], int N)
{ int i ;
 for(i=0 ;i<N ;i++)
 {
      printf("\n Donner le nom du sportif: ") ;
      gets(TSport[i].Nom) ;
      printf("\n Donner le prénom du sportif: ") ;
      gets(TSport[i].Prenom) ;
      printf("\n Donner le pays du sportif: ") ;
      gets(TSport[i].Pays) ;
      printf("\nDonner la date de naissance (jour/mois/annee): ") ;
      scanf("%d",TSport[i].DN.jour) ;
```

```
      scanf("%d",TSport[i].DN.mois) ;
      scanf("%d",TSport[i].DN.annee) ;
  }
}
void TriSport(struct Sportif TSport[], int N)
{
  struct Sportif * P, AIDE ;

  /* Tri du tableau par sélection directe du maximum. */
  for (P=TSport; P<TSport+N-1; P++)
     {
      /* Recherche du maximum à droite */
      PMAX=P-TSport;
      for (Pj=P+1; Pj<TSport+N; Pj++)
          if ((*Pj).performance > (*(TSport+PMAX)).performance)
              PMAX=Pj-TSport ;
      /* Echange de *P avec le maximum */
      AIDE= *P;
      *P =*(TSport+PMAX);
      *(TSport+PMAX)=AIDE;
     }
}

void AffichSport(struct Sportif TSport[],int N)
{
         TriSport(TSport, N) ;
          printf("\nLes trois vaiqueurs sont : ") ;
          printf("\n      Medaille      d'Or :      %s      %s      %s
",TSport[0].Nom,TSport[0].Prénom,TSport[0].pays) ;
         printf("\n     Medaille     d'argent :     %s     %s     %s
",TSport[1].Nom,TSport[1].Prénom,TSport[1].pays) ;
         printf("\n     Medaille     de     Bronze:     %s     %s     %s
",TSport[02].Nom,TSport[2].Prénom,TSport[2].pays) ;
}
```

Exercice 6

```
struct Produit
{     int code ;
      char Nom[20] ;
      int prix ;
} ;
struct Caissier
{     char id[30] ;
      char nom[20] ;
      strcut Produit TProd[100] ;
      int NP ;
      int soldeCaisse ;
} ;
void AfficherProduit(struct Produit P)
{     printf("\nle    produit    code    :    %d    ,nom    %s,    prix
%d",P.code,P.Nom,P.prix) ;
}
void VendreProduit(struct Produit P, struct Caissier *C)
{     *C.TProd[NP].code = P.code ;
```

```
      strcpy(*C.TProd[NP].Nom,P.Nom) ;
      *C.TProd[NP].prix=P.prix ;
      *C.NP ++;
      *C.soldeCaisse+=P.prix ;
}
void ListeProduit(struct Caissier C)
{     int i ;
      for(i=0 ;i<C.NP ;i++)
      {     printf("\nProduit
code:%d,nom :%s,prix%d",C.TProd[i].code,C.TProd[i].Nom,C.TProd[i].pr
ix) ;
      }
}
int SoldeCaisse(struct Caissier C)
{     int i , SC =0 ;
      for(i=0 ;i<C.NP ;i++)
      {     SC+= C.TProd[i].prix ;
      }
      return SC ;
}
void ChangerPrix(struct Produit * P ,int Px)
{
      *P.prix = Px ;
}
struct Produit PlusCher(struct Caissier C)
{     int i , maxp ;
      maxp = C.TProd[0].prix ;
      for(i=1 ;i<C.NP ;i++)
      {     if(maxp< C.TProd[i].prix)
                  { maxp= C.TProd[i].prix ;
                    Pos=i ;
                   }
      }
      return (C.TProd[Pos]) ;
}
int ChercherProduit(struct Caissier C, int codeB)
{     int i ;
      i=0;
      while (C.TProd[i].code!=codeB && i<C.NP )
            i++ ;
      if(C.TProd[i].code==codeB)
            return 1 ;
      else
            return 0 ;
}
Struct Caissier MeilleurCaissier(struct Caissier TCaisses[10])
{     int i , max ,Pos;
      max = TCaisses[0].SoldeCaisse ;
      for(i=1 ;i<10 ;i++)
      {     if(max< TCaisses[i].SoldeCaisse )
                  {     max= TCaisses[i].SoldeCaisse  ;
                        Pos=i ;
                  }
      }
      return (TCaisses[Pos]) ;

}
```

Correction TP N° 9

Exercice 1

	A	B	C	P1	P2
`int A = 1, B = 2, C = 3;` `int *P1, *P2;`	1	2	3	/	/
`P1=&A;`	1	2	3	&A	/
`P2=&C;`	1	2	3	&A	&C
`*P1=(*P2)++ ;`	3	2	4	&A	&C
`P1=P2 ;`	3	2	4	&C	&C
`P2=&B ;`	3	2	4	&C	&B
`*P1-=*P2 ;`	3	2	2	&C	&B
`++*P2 ;`	3	3	2	&C	&B
`*P1*=*P2 ;`	3	3	6	&C	&B
`A=*P2**P1 ;`	18	3	6	&C	&B
`P1=&A ;`	18	3	6	&A	&B
`*P1/=*P2 ;`	6	3	6	&A	&B

Exercice 2

```
a) *P+2   = 14
b) *(P+2) = 34
c) &P+1   = A+1
d) &A[4]-3= &A[1]
e) A+3 =&A[3]
f) &A[7]-P = A+7-A= 7
g) P+(*P-10)= P+2 = &A[2]
h) *(P+*(P+8)-A[7])= *(A+90-89)=A[1]=23
```

Exercice 3

```
#include <stdio.h>
main()
{ int N, M ;
  int *PA, *PB ;
  int A[50],B[50] ;

  printf("\n Donner N et M ") ;
  scanf("%d%d",&N,&M) ;

  for(PA=A ;PA<A+N ;PA++)
  {
     printf("\n Donner A[%d]: ",PA-A) ;
     scanf("%d",PA) ;
  }
```

```
  for(PB=B ;PB<B+M ;PB++)
  {
     printf("\n Donner B[%d]: ",PB-B) ;
     scanf("%d",PB) ;
  }
  for(PA=A+N,PB=B ;PA<A+N+M,PB<B+M ;PA++,PB++)
     *PA=*PB ;
  /*     Affichage du tableau résultat   */
  for(PA=A ;PA<A+N+M ;PA++)
       printf("\n A[%d]= %d ",PA-A,*PA) ;
 }
```

Exercice 4

```
#include <stdio.h>
main()
{ int N, X ;
  int *P1, *P2 ;
  int A[50] ;

  printf("\n Donner N ") ;
  scanf("%d",&N) ;
  for(P1=A ;P1<A+N ;P1++)
  {
     printf("\n Donner A[%d]: ",P1-A) ;
     scanf("%d",P1) ;
  }
  printf("\n Donner X : ") ;
  scanf("%d",&X) ;

  for(P1=A;P1<A+N ;P1++)
  {  if(*P1== X)
     {     for(P2=P1 ;P2<A+N ;P2++)
                *P2=*(P2+1) ;
           N-- ;
     }
  }
  /*   Affichage du tableau résultat   */
  for(P1=A ;P1<A+N ;P1++)
       printf("\n A[%d]= %d ",P1-A,*P1) ;
 }
```

Exercice 5

```
#include <stdio.h>
main()
{ int N, NA=0, NB=0 ;
  Int *P, *P1, *P2 ;
  int T[100] ;
  int TPOS[100], TNEG[100] ;

  printf("\n Donner N ") ;
  scanf("%d",&N) ;
  for(P=T ;P<T+N ;P++)
  {
```

```
      printf("\n Donner T[%d]: ",P-T) ;
      scanf("%d",P) ;
   }
   P1=TPOS ;
   P2=TNEG ;
   for(P=T;P<T+N ;P++)
   {  if(*P > 0)
      {     *P1=*P ;
             P1++ ;
             NA++ ;
      }
      else if(*P < 0)
      {     *P2=*P ;
             P2++ ;
             NB++ ;
      }
   }
   /*   Affichage des tableaux TPOS et TNEG   */
   for(P1=TPOS ;P1<TPOS+NA ;P1++)
        printf("\n TPOS[%d]= %d ",P1-TPOS,*P1) ;
   for(P2=TNEG ;P2<TNEG+NB ;P2++)
        printf("\n TNEG[%d]= %d ",P2-TNEG,*P2) ;

 }
```

Exercice 6

```
#include <stdio.h>
main()
{ /* Déclarations */
 char CH[20]; /* chaîne donnée       */
 int *PH ;    /* pointeur courant          */
 int L = 0 ;    /* longueur de la chaîne    */

 /* Saisie des données */
 printf("Entrez une chaine (max.20 caractères) :\n");
 gets(CH);

 /* a) Compter les caractères */
 for (PH=CH; *PH!= '\0'; PH++);
 L= PH - CH;
 printf("\nLa longueur de la chaine est %d",L);
}
```

Exercice 7

```
#include <stdio.h>
main()
{
 /* Déclarations */
 char TABCH[5][51];/* tableau de chaînes de caractères */
 char AIDE;        /* pour la permutation des caractères */
 char *P1, *P2;    /* pointeurs d'aide */
 int I;            /* indice courant   */
```

```
 /* TABCH+I est l'adresse de la I-ième chaîne du tableau */
 /* Il vaut mieux convertir TABCH+I en pointeur sur char */
  /* Saisie des données */
 printf("Entrez 5 mots :\n");
 for (I=0; I<5; I++)
     {
      printf("Mot %d (max.50 caractères) : ", I);
      gets((char *)(TABCH+I));
     }

 /* Inverser l'ordre des caractères à l'intérieur des mots */
 for (I=0; I<5; I++)
     {
      P1 = P2 = (char *)(TABCH+I);
      /* Placer P2 à la fin de la chaîne */
      while (*P2)
             P2++;
      P2--; /* sinon '\0' est placé au début de la chaîne */
      while (P1<P2)
           {
            AIDE = *P1;
            *P1  = *P2;
            *P2  = AIDE;
            P1++;
            P2--;
           }
     }

  /* Affichage des mots inversés */
 for (I=0; I<5; I++)
      puts((char *)(TABCH+I));
 return 0;
}
```

Exercice 8

```
#include <stdio.h>
#include <string.h>
 main()
{
 /* Déclarations */
 int T1[50], T2[50]; /* Tableaux donnés */
 int N,M ;     /* Dimensions des tableaux  */
 int *P1, *P2; /* pointeurs d'aide dans T1 et T2 */
 int TROUVE; /* indicateur logique: vrai, si le caractère */
             /* actuel dans T1 a été trouvé dans T2.    */

 /* Saisie des données */
  printf("\n Donner les tailles des tableaux N et M :") ;
  scanf("%d%d",&N,&M) ;

  printf("\n Introduire les éléments du tableau T1 :")
  for(P1=T1 ;P1<T1+N ;P1++)
  {
```

```
      printf("\n Donner T1[%d]: ",P1-T1) ;
      scanf("%d",P1) ;
  }

  printf("\n Introduire les éléments du tableau T : ") ;
  for(P2=T2 ;P2<T2+M ;P2++)
  {
      printf("\n Donner T2[%d]: ",P2-T2) ;
      scanf("%d",P2) ;
  }

 /* Rechercher T2 dans CT1 : */
 /* L'expression P2-T2 est utilisée pour déterminer l'indice */
 /* de P2 dans T2. On pourrait aussi résoudre le problème à  */
 /* l'aide d'un troisième pointeur P3 parcourant T1. */
 TROUVE=0;
 for (P1=T1 ; *P1 && !TROUVE ; P1++)
     {
      for (P2=T2 ; *P2 == *(P1+(P2-T2)) ; P2++)
          ;
      if (!*P2)
          TROUVE = 1;
     }

 /* A la fin de la boucle, P1 est incrémenté, donc */
 P1--;
 /* Si T2 se trouve dans T1, alors P1 indique la position */
 /* de la première occurrence de T2 dans T1 et P2 pointe à */
 /* la fin de T2. (P2-T2) est alors la longueur de T2.   */
 if (TROUVE)
       P1=P1+(P2-T2) ;

 /* Affichage du résultat */
 for(P1=T1 ;P1<T1+N-M ;P1++)
       printf("\nTableau résultat : \"%d \n", *P1);
 return 0;
}
```

Exercice 9

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50][50]; /* matrice donnée */
 int B[50][50]; /* matrice donnée */
 int C[50][50]; /* matrice résultat */
 int N, M, P;   /* dimensions des matrices */
 int I, J, K;   /* indices courants */

  /* Saisie des données */
 printf("*** Matrice A ***\n");
 printf("Nombre de lignes de   A (max.50) : ");
 scanf("%d", &N );
 printf("Nombre de colonnes de A (max.50) : ");
 scanf("%d", &M );
```

```
 for (I=0; I<N; I++)
   for (J=0; J<M; J++)
      {
       printf("Elément[%d][%d] : ",I,J);
       scanf("%d", (int *)A+I*50+J);
      }
 printf("*** Matrice B ***\n");
 printf("Nombre de lignes de   B : %d\n", M);
 printf("Nombre de colonnes de B (max.50) : ");
 scanf("%d", &P );
 for (I=0; I<M; I++)
   for (J=0; J<P; J++)
      {
       printf("Elément[%d][%d] : ",I,J);
       scanf("%d", (int *)B+I*50+J);
      }

  /* Affichage des matrices */
 printf("Matrice donnée A :\n");
 for (I=0; I<N; I++)
   {
    for (J=0; J<M; J++)
        printf("%7d", *((int *)A+I*50+J));
    printf("\n");
   }
 printf("Matrice donnée B :\n");
 for (I=0; I<M; I++)
   {
    for (J=0; J<P; J++)
        printf("%7d", *((int *)B+I*50+J));
    printf("\n");
   }


 /* Affectation du résultat de la multiplication à C */
 for (I=0; I<N; I++)
    for (J=0; J<P; J++)
       {
        *((int *)C+I*50+J)=0;
        for (K=0; K<M; K++)
*((int*)C+I*50+J) += *((int*)A+I*50+K) * *((int*)B+K*50+J);
       }

 /* Edition du résultat */
 printf("Matrice résultat C :\n");
 for (I=0; I<N; I++)
   {
    for (J=0; J<P; J++)
        printf("%7d", *((int *)C+I*50+J));
    printf("\n");
   }
 return 0;
}
```

Exercice 10

```
void ChercherVal (int tab[], int n, int A, int *pos, int *nbOcc)
{       int *P ;
        *pos= -1 ;
        for(P=tab ;P<tab+n ;P++)
        {       if(*P==A)
                { *pos = P-tab ;
                  *nbOcc ++ ;
                }
        }
}
```

Exercice 11

```
int EstVoyelle (char C)
{
        if(C=='a'||C=='e'||C=='i'||C=='u'||C=='o'||C=='y')
                return 1 ;
        else
                return -1 ;
}
void NBVoyelle (char CH[],int *V,int*S)
{ char * PH ;
   for(PH=CH ;*PH !='\0' ;PH++)
        if(EstVoyelle(*PH))
                *V ++ ;
        else
                *S ++ ;
}
```

Exercice 13

```
void SupprimerC(char TXT[], char C)
{
 char *P;  /* pointeur d'aide dans TXT */
 /* Comprimer la chaîne à l'aide de strcpy */
 P = TXT;
 while (*P)
   {
      if (*P==C)
          strcpy(P, P+1);
      else P++;
   }
}
```