Annexe2 : Correction des Travaux Dirigés

Correction TD N°1 : Familiarisation avec langage C et notions de bases

Exercice 1 :

```
#include <stdio.h>
int main()
{
      int x ;
      printf ("donner un entier :");
      scanf("%d",&x) ;
      printf("\n L'opposé de %d est %d ",x,-x);
      return 0;
}
```

Exercice 2 :

```
#include <stdio.h>
int main()
{
      int x, y, z, s, p ;
      float M ;
      printf ("donner un entier x :");
      scanf("%d",&x) ;
      printf ("donner un deuxième entier y :");
      scanf("%d",&y) ;
      printf ("donner un trosième entier z :");
      scanf("%d",&z) ;
      s = x+y+z ;
      p = x*y*z ;
      M = (float)s/3 ;
      printf("\n la somme est %d",s);
      printf("\n le produit est %d",p);
      printf("\n la moyenne est %f",M);
      return 0;
}
```

Exercice 3 :

```
#include <stdio.h>
int main()
{
      int x , j, m, a ;
      printf ("donner un entier de 6 chiffres :");
      scanf("%d",&x) ;
      a = x % 100 ;
      j = x/ 10000 ;
      m = (x%10000)/100 ;
      printf("\n Jour : %d ",j);
```

```
      printf("\n Mois : %d ",m);
      printf("\n Année : %d ",a+2000);
      return 0;
}
```

Exercice 4 :

```
#include <stdio.h>
int main()
{
      int r , p, s;
      printf ("donner le rayon :");
      scanf("%d",&r) ;
      printf("\n Le périmètre du cercle est %d ",r*r);
      printf("\n La surface du cercle est %d ",2*3.14*r);
      return 0;
}
```

Exercice 5 :

```
#include <stdio.h>
int main()
{
      float TF, TK, TC ;
      printf ("\n donner la température TF :");
      scanf("%f",&TF) ;
      TC = (TF - 32)/1.8 ;
      TK = 273 + TC ;
      printf("\n La température en degré Kelvin TK est %f ",TK);
      return 0;
}
```

Correction TD N°2 : Structures conditionnelles

Exercice 1 :

```
#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 le maximum",a);
        else if(a<b)
                printf ("B = %d est le maximum",b);

     return 0;
   }
```

Exercice 2 :

```
#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 3 :

```
#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 4 :

```
#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 5 :

```
#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 6 :

```
#include <stdio.h>
#include <conio.h>
void main()
{
   float moy;
   printf("\n Introduire la moyenne: ") ;
   scanf("%f" ,&moy) ;
   if(moy==20)
      printf("\n Parfait") ;
   else if(moy<20 && moy>=18)
             printf("\n Excellent") ;
        else if(moy<18 && moy>=17)
                  printf(" \n Très bien ") ;
             else if(moy<17 && moy>=14)
                     printf(" \n Bien") ;
                  else if(moy<14 && moy>=12)
                           printf(" \n Assez bien") ;
                       else if(moy<12 && moy>=10)
                                 printf(" \n Passable") ;
                            else if(moy<10)
                                 printf(" \n Non admis") ;

    getch() ;
}
```

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 op1, op2 ;
      char op ;
      printf("\n donner opérande1:");
      scanf("%d",&op1);
      printf("\n donner opérande2:");
      scanf("%d",&op2);
      printf("\n donner l'opérateur :");
      scanf("%c",&op);
      switch(op)
      {
  case '+' : printf ("\n %d%c%d=%d",op1,op,op2,op1+op2) ;break ;
  case '-' : printf ("\n %d%c%d=%d",op1,op,op2,op1-op2) ;break ;
  case '*' : printf ("\n %d%c%d=%d",op1,op,op2,op1*op2) ;break ;
  case '/' : printf ("\n %d%c%d=%f",op1,op,op2,float(op1/op2)) ;break ;
  default : printf("\n ERREUR !!") ;
     }
      getch() ;
}
```

Exercice 10 :

```
#include <stdio.h>
#include <conio.h>
void main()
{
      int ind1, ind2 , kwh ;
      float MHT, MTTC ;
      printf("\n donner l'ancien index:");
      scanf("%d",&ind1);
      printf("\n donner le nouvel index:");
      scanf("%d",&ind2);
      kwh = ind2 - ind1 ;
      if(kwh<=100)
            MHT = 2500 + (kwh*100) ;
      else if(kwh>100 && kwh <=250)
            MHT = 2500+(100*100)+(kwh-100)*175 ;
      else if(kwh>250)
            MHT = 2500+(100*100)+(150*175)+(kwh-250)*225 ;
      MTTC = (float)MHT * (1+.08) ;
      printf("\n la consommation en kwh est : %d ",kwh);
```

```
      printf("\n le montant à payer hors taxes : %d ",MHT);
      printf("\n le montant à payer TTC : %d ",MTTC);
      getch() ;
}
```

Exercice 11 :

```
#include <stdio.h>
#include <conio.h>
void main()
{
      int N , MI;
      float QF , RI, IMP ;
      printf("\n donner le nombre de personnes de la famille N :");
      scanf("%d",&N);
      printf("\n donner le montant imposable MI:");
      scanf("%d",&MI);
      RI=(float)(MI*0.9)*0.8 ;
      QF=RI/N ;

      if(QF<1800)
            IMP = 0 ;
      else if(QF>=1800 && QF <2500)
               IMP = (RI*0.15)-(140*N) ;
            else if(QF>=2500 && QF<4500)
                     IMP = (RI*0.25)-(420*N) ;
                  else if(QF>=4500 && QF<=6000)
                              IMP = (RI*0.35)-(940*N) ;
                        else if(QF>6000)
                                    IMP = (RI*0.45)-(3040*N) ;

      printf("\n l'impot à payer IMP = %f ",IMP);
      getch() ;
}
```

Correction TD N°3: Structures répétitives

Exercice 1 :

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

Exercice 2 :

```
#include<stdio.h>
int main ()
{
int I, SOM, PROD ;
for (SOM=0, PROD=1, I=1 ; I<=100 ; I++)
    {
     printf("%d. nombre : ", I);
     scanf("%d", &NOMB);
     SOM  += NOMB;
     PROD *= NOMB;
    }
prinf("\ la somme est %d",SOM) ;
printf("\n le produit est %d",PROD) ;
getch() ;
}
```

Exercice 3 :

```
#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 4 :

```
#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 ("\n donner un entier ente 10 et 20: ");
    scanf ("%d", &N);
   }
 while (N<10 || N>20);
 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 8 :

```
#include<stdio.h>
int main ()
{
     int i, N, U0, Un ;
     printf("Entrer un entier : \n");
     scanf("%d",&N);
       U0 = 1 ;
     for(i=1 ; i<=N ; i++)
     { Un = 5*U0+3 ;
       U0=Un ;
     }

     printf("\n Un = %d ",Un);
     return 0;
}
```

Exercice 9 :

```
#include<stdio.h>
int main ()
{
     int i, N, U0, Un, V0, Vn ;
     printf("Entrer un entier : \n");
     scanf("%d",&N);
     U0 = 5 ;
     V0 = 3 ;
     for(i=1 ; i<=N ; i++)
     { Un = 5*V0 - 2*U0 +3 ;
       Vn = 3*V0 + 7 ;
       U0=Un ;
       V0=Vn ;
     }

     printf("\n Un = %d et Vn = %d ",Un,Vn);
     return 0;
}
```

Exercice 10 :

Ecrire un programme C fournissant la moyenne d'un nombre n de notes obtenues par des élèves. Le premier nombre lu sur l'organe d'entrée est n, les suivants étant les notes.

```
#include <stdio.h>
main()
{
 int N, X, S ;      /* entier N donné */
 int I ;
 float M ;
 do
   {
    printf ("Donner le nombre des notes N : ");
    scanf ("%d", &N);
   }
 while (N<1);

 for (I=1 ; I<=N ; I++)
 {     printf ("Donner un entier X : ");
       scanf ("%d", &X);
       S+=X ;
 }
 M = (float)S/N ;
 printf("\n La moyenne des notes est %f", M);

}
```

Exercice 11 :

```
#include <stdio.h>
main()
{
 int N, X;      /* entier N donné */
 int I, sp=0, sn=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++ ;
            sp+=X ;
       }
       else if(X<0)
       {    Nn++ ;
            sn+=X ;
       }
       if(X %2== 0)
           NbPair++ ;
       else
           NbImp++ ;

 }
```

```
 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 la somme des nombres positifs est %d", sp);
 printf("\n la somme des nombres négatifs est %d", sn);

}
```

Exercice 12 :

```
#include <stdio.h>
main()
{
 int N, X;
 int I, P=0 ;
 printf ("Donner un entier X : ");
 scanf ("%d", &X);
 for(I=2 ;I<X ;I++)
 {  if(X%I ==0)
       P=1 ;
 }
 if(P==1)
       printf ("\n %d est premier",X);
 else
       printf ("\n %d n'est pas premier",X);

 }
```

Correction TD N°4 : Les Tableaux

Exercice 1 :

Ecrire un programme C qui lit et affiche les éléments d'un tableau de N entiers (max. 50) dans l'ordre des indices décroissants (de droite à gauche).

Exercice 2 :

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int Tab[50]; /* tableau donnée   */
 int N;
 int I ;      /* indice */
 int val , Nbocc = 0; /* entier à saisir et nombre d'occurrence */

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

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

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

Exercice 3 :

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int Tab[50]; /* tableau donnée   */
 int N;
 int I ;      /* indice */
 int MAX, MIN;
```

```
 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N);
 for (I=0; I<N; I++)
 {
        printf("Elément[%d] : ",I) ;
        scanf("%d", &Tab[I]);
  }
/*recherche du maximum */
  MAX=Tab0];
  for (I=1; I<N; I++)
  {     if(Tab[I]> MAX)
           MAX=Tab[I];
  }
  /* Edition du résultat */
  printf("\n Valeur   du maximum : %d\n", MAX);

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

Exercice 4 :

```
#include <stdio.h>
main()
{
 /* Déclarations */
 float Tab[50]; /* tableau donné */
 float X;   /* 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("%f", &Tab[I]);
    }
 printf("Elément à rechercher : ");
 scanf("%f", &X );
 /* Affichage du tableau */
 printf("Tableau donné : \n");
 for (I=0; I<N; I++)
     printf("%f ", Tab[I]);
 printf("\n");
 /* Recherche de la position de la valeur */
I=0 ;
While((I<N)&& (Tab[I]!=X))
```

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

Exercice 5 :

```
#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;
}
```

Correction TD N°5 : Les chaines de caractères

Exercice 1 :

```
#include <stdio.h>
#include <string.h>
main()
{
 /* Déclarations */
 char CH[20], big[2]; /* chaine donnée */
 int N, nbo;      /* dimension       */
 int I, J;      /* indices courants */

 /* Saisie des données */
 printf("Donner une chaine(max.20) : ");
 gets(CH) ;
 printf("Donner un bigramme (chaine de 2 caractères) : ");
 gets(big) ;
 N = strlen(CH) ;
 for(I=0 ;I<N ;I++)
 {
       if(CH[I]==big[0] && CH[I+1]==big[1])
              nbo++ ;
 }
 printf("\n Le nnombre d'occurrence est %d : ",nbo);
 retrun 0 ;
}
```

Exercice 2 :

```
#include <stdio.h>
#include <string.h>
main()
{
 /* Déclarations */
 char Mot[20]; /* chaine donnée */
 int N, Pal=1;      /* dimension       */
 int I, J;      /* indices courants */

 /* Saisie des données */
 printf("Donner un mot(max.20) : ");
 gets(Mot) ;
 N = strlen(Mot) ;

 I=0 ;
 J=N-1;
 While(I<N/2 && Pal ==1)
 {     if(Mot[I] != Mot[J])
              Pal=0 ;
       I++ ;
       J-- ;
 }
 if(Pal==1)
```

```
       printf("\n Chaine palindrome");
 else
     printf("\n chaine non palindrome");

 return 0;
}
```

Correction TD N°6 : Les Fonctions

Exercice 1 :

```
/* Définition de la fonction PERMUT */
void PERMUT(int * N1, int * N2)
{       int aux ;
        aux = *N1 ;
        *N1=*N2 ;
        *N2=aux ;
}

/* Définition de la fonction LIRE_TAB */
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);
    }
 }

 /* Définition de la fonction ECRIRE_TAB */
 void ECRIRE_TAB(int TAB[], int N)
 {
  int I;
  for (I=0; I<N; I++)
     printf("\n TAB[%d] = %d ", I, TAB[I]);
 }
```

Exercice 2 :

```
/* Définition de la fonction DIV2 */
int DIV2(int N)
 {
  int I, Nb ;
  Nb =0 ;
  while( N%2==0)
  {
       Nb++ ;
       N=N/2 ;
   }
   return Nb ;
```

```
 }
 /* Définition de la fonction main() */
 main()
 {
  int X, Nb =0;
  printf("\n donner un entier : ") ;
  scanf("%d",&X) ;
  Nb = DIV2(X) ;
  printf("\n le nombre de fois de division par 2 est %d ",Nb) ;
 }
```

Exercice 3 :

```
/* Définition de la fonction LIRE_TAB */
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);
    }
 }

/* Définition de la fonction VERIF_TRIE */
 int VERIF_TRIE(int TAB[], int  N)
 {
  /* Variables locales */
  int I=0 ;
  while (i<N && TAB[I] < TAB[I+1])
       I++ ;
  if(I==N)
       return 1 ;
  else
       return 0 ;

 }
/* Définition de la fonction main() */
 main()
 {
  int  N, X ;
  int T[50] ;
  LIRE_TAB(T,&N,50) ;
  X=VERIF_TRIE(T,N) ;
  if(X == 1)
       printf("\n Le tableau est trié.") ;
  else
```

```
        printf("\n Le tableau n'est pas trié.") ;
 }
```

Exercice 4 :

```
/* Définition de la fonction MIN */
 int MIN(int  X, int  Y)
 {
  /* Variables locales */
  int min=0 ;
  if((X-Y)>0)
       min = X+Y-(X-Y)/2 ;
  else if((X-Y)<0)
       min = X+Y-(-X+Y)/2 ;
  return min ;

 }
 /* Définition de la fonction main() */
 main()
 {
  int  N1, N2, MN ;
  printf("\n Donner 2 entiers :") ;
  scanf("%d%d",&N1,&N2) ;
  MN=MIN(N1,N2) ;
  printf("\n Le minimum est %d .",MN) ;
 }
```

Exercice 5 :

```
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 6 :

```
/* Définition de la fonction COUT_KM */
float COUT_KM(int NbKm)
{
 float cout ; ;
 if(NbKm < 100)
       cout = (float)NbKm * 0.120 ;
 else if(NbKm >100 && NbKm <=1000)
              cout = (100 * 0.120)+ ((float)NbKm-100)*0.150 ;
    else if(NbKm >1000)
          cout = (100 * 0.120)+ (900 * 0.150)+ ((float)NbKm-1000)*170 ;

 return cout ;
}
```

```
/* Définition de la fonction COUT_JOURNALIER */
float COUT_JOURNALIER(int NbJour)
{
 float cout ; ;
 cout = (float) NbJour * 75 ;
 return cout ;
}

 /* Définition de la fonction main() */
 main()
 {
  int  km, nj ;
  float c1, c2 ;
  printf("\n Donner le nombre de jour :") ;
  scanf("%d",&nj) ;
  printf("\n Donner le nombre de kilomètres :") ;
  scanf("%d",&km) ;
  c1 = COUT_KM(km) ;
  c2 = COUT_JOURNALIER(nj) ;
  if(c1>c2 )
       printf("\n La tarification par kilomètres est meilleure.") ;
  else
       printf("\n La tarification journalière est meilleure.") ;

 }
```

Exercice 7 :

```
/* Définition de la fonction LIRE_TAB */
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);
    }
 }
/* Définition de la fonction SYMETRIQUE */
int SYMETRIQUE(int TAB[], int N)
 {     int I , J , Sm ;
       I=0 ;
       J=N-1 ;
       Sm=1 ;
       while(I<N/2 && Sm ==1)
       {      if(TAB[I] !=TAB[J])
```

```
                    Sm=0 ;
               I++ ;
               J-- ;
        }
        return Sm ;
 }
```

Exercice 9 :

```
/* Définition de la fonction LIRE_MATRICE */
void LIRE_MATRICE (int *MAT, int L, int C, int CMAX)
{
 /* Variables locales */
 int I,J;
 /* Saisie des composantes de la matrice */
 for (I=0; I<L; I++)
      for (J=0; J<C; J++)
          {
           printf("Elément[%d][%d] : ", I, J);
           scanf("%d", MAT + I*CMAX + J);
          }
}

/* Définition de la fonction TRANSPO_MATRICE*/
int TRANSPO_MATRICE (int *MAT, int *L, int LMAX, int *C, int CMAX)
{
  /* Prototypes des fonctions appelées */
 void PERMUTER(int *A, int *B);
 /* Variables locales */
 int I,J;
 int DMAX; /* la plus grande des deux dimensions */
 /* Transposition de la matrice */
 if (*L>CMAX || *C>LMAX)
      return 0;
 else
      {
       DMAX = (*L>*C) ? *L : *C;
       for (I=0; I<DMAX; I++)
            for (J=0; J<I; J++)
                 PERMUTER (MAT+I*CMAX+J, MAT+J*CMAX+I);
       PERMUTER(L,C);  /* échanger les dimensions */
       return 1;
      }
}

/* Définition de la fonction ECRIRE_MATRICE*/
void ECRIRE_MATRICE (int *MAT, int L, int C, int CMAX)
{
 /* Variables locales */
 int I,J;
 /* Affichage des composantes de la matrice */
 for (I=0; I<L; I++)
     {
      for (J=0; J<C; J++)
           printf("%7d", *(MAT + I*CMAX + J));
```

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

Correction TD N°7 : Les Structures

Exercice 1 :

```
#include <stdio.h>
main()
{
 struct Article
 {      int Ref ;
        char NOM[10] ;
        int Prix ;
 } ;
 struct Article T[100] ;
 int i, n, p1, p2 , px;
 int r ;
 printf("\n donner le nombre des articles :") ;
 scanf("%d",&n) ;

 for(i=0 ;i<n ;i++)
 {      printf("\n donner la référence de l'article :") ;
        scanf("%d",&T[i].Ref) ;
        printf("\n donner e nom de l'article :") ;
        gets(T[i].Nom) ;
        printf("\n donner le prix de l'article :") ;
        scanf("%d",&T[i].Prix) ;
  }
  printf("\n donner 2 prix :") ;
  scanf("%d%d",&p1,&p2) ;
  printf("\n les articles sont :") ;
  for(i=0 ;i<n ;i++)
  {     if(p1<T[i].Prix && T[i].Prix>p2)
                printf("\n %d",T[i].Ref) ;
   }
   printf("\n donner la référence d'un article :") ;
   scanf("%d",&r) ;
   for(i=0 ;i<n ;i++)
   {    if(T[i].Ref==r)
           printf("\n le prix de l'article ref %d est %d",r,T[i].Prix) ;
   }
}
```

Correction TD N°8 : Les Pointeurs

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++;
   }
}
```