RÉPUBLIQUE TUNISIENNE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

DOSSIER DE CANDIDATURE

Soumis à

La commission Nationale de Recrutement des Maîtres assistants

Dans la discipline

Informatique

C3M

Travaux Dirigés Corrigés
Algorithmique et Programmation 1

Elaboré par :
Dr. Ghassen HAMDI
Laboratoire MARS, Université de Sousse

République Tunisienne
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale d'Ingénieurs de Sousse

Travaux Dirigés

Algorithmique et Programmation1

Niveau : 1ère Mécatronique, 1ère Génie Mécanique et Productique

Préparé Par : *HAMDI Ghassen*

Docteur en Sciences Informatique.

Année universitaire: 2021/2022

Email **: hamdighassan@gmail.com**

Travaux dirigés
Algorithmique et Programmation1

Objectif(s) :

- S'habituer à résoudre des exercices algorithmiques en utilisant le langage C
- Manipuler des structures de données comme les tableaux, les chaines de caractères, les pointeurs, les fonctions etc., en utilisant le langage C

Bibliographie :

- https://www.examanet.net/universite-de-tunis-el-manar/fseg-tunis/137-exercices-corriges-langage-c

Sommaire

TD 1 : Les fondamentaux de la programmation dans le langage C

Exercice 1

En **langage C, élaborez un programme** qui, après avoir saisi les valeurs de trois variables entières (X, Y, Z) depuis le clavier, les transpose et les affiche :

X ==> Y, Y ==> Z, Z ==> X

Correction

```c
#include <stdio.h>
main()
{
        int X, Y, Z, AUX;
        printf("Entrer trois valeurs (X, Y, Z) : ");
        scanf("%d %d %d", &X, &Y, &Z);
        /* Affichage à l'aide de tabulations */
        printf("X = %d\tY = %d\tZ = %d\n", X, Y, Z);
        AUX=X;
        X=Z;
        Z=Y;
        Y=AUX;
        printf("X = %d\tY = %d\tZ = %d\n", X,Y,Z);
        return 0;
}
```

Exercice 2

En **langage C, élaborer un programme** capable d'afficher simultanément le quotient et le reste résultant d'une division entière entre deux nombres entiers donnés, tout en présentant également leurs quotients sous forme rationnelle.

Correction

```c
#include <stdio.h>
main()
{
        int X,Y;
        printf("Donner deux nombres entiers : ");
        scanf("%d %d", &X, &Y);
        printf("Division entière    :  %d\n", X/Y);
        printf("Reste               :  %d\n", X%Y);
        printf("Quotient rationnel  :  %f\n", (float)X/Y
            );
        return 0;
}
```

Exercice 3

Élaborez un programme en langage C pour calculer et afficher la résistance équivalente de trois résistances R1, R2, R3 (de type double), en tenant compte de leur arrangement :

- En cas de connexion en série des résistances : Rsér = R1 + R2 + R3
- En cas de connexion en parallèle des résistances : Rpar = (R1 * R2 * R3) / (R1 * R2 + R1 * R3 + R2 * R3)

Correction

```c
#include <stdio.h>
main()
{
   double R1, R2, R3, RRES;
   printf("Donner les valeurs de résistances R1, R2 et R3 : ");
   scanf("%lf %lf %lf", &R1, &R2, &R3);
   RRES=R1+R2+R3;
   printf("Résistance résultante sérielle   : %f\n", RRES);
   RRES=(R1*R2*R3)/(R1*R2+R1*R3+R2*R3);
   printf("Résistance résultante parallèle  : %f\n", RRES);
   return 0;
}
```

Exercice 4

En **langage C,** élaborez **un programme** qui, en prenant en compte les longueurs des trois côtés d'un triangle fournies en entrée, calcule et affiche l'aire du triangle en utilisant la formule :

$S^2 = P(P-X)\ (P-Y)\ (P-Z)$

Où X, Y, Z représentent les dimensions des trois côtés du triangle (de type int), où P symbolise La moitié du périmètre du triangle.

Correction

```c
#include <stdio.h>
#include <math.h>
 main()
{
                    /* Pour ne pas perdre de précision lors de la
                        division, */
                    /* déclarons P comme rationnel. */
   int X, Y, Z;
   float P;
   printf("Introduisez les valeurs de X, Y et Z : ");
   scanf("%d %d %d", &X, &Y, &Z);
                    /* En forçant la conversion de X, les autres
                        opérandes */
                    /* sont converties automatiquement. */
   P=((float)X+Y+Z)/2;
   printf("Surface du triangle S = %f\n",sqrt(P*(P-X)*(P-Y)*(P-Z)));
    return 0;
}
```

Exercice 5

En **langage C,** élaborer **un programme** destiné à calculer la somme de quatre **nombres entiers** spécifiés.

Correction

```c
#include<stdio.h>
main()
{
int  X;
 long SOM;
 SOM = 0;
  printf("Entrez le premier   nombre : ");
 scanf("%d", &X);
 SOM+=X;
  printf("Entrez le deuxième  nombre : ");
 scanf("%d", &X);
 SOM+=X;
  printf("Entrez le troisième nombre : ");
 scanf("%d", &X);
 SOM+=X;
  printf("Entrez le quatrième nombre : ");
 scanf("%d", &X);
 SOM+=X;
  printf("La somme des nombres entrés est %ld\n", SOM);
 return 0;
}
```

Exercice 6

Implémenter en **langage C** un programme pour le calcul et la présentation de la distance DIST (**de type double**) entre deux points A et B dans le plan, en utilisant les coordonnées (XA, YA) et (XB, YB) fournies en tant que valeurs de type int.

Correction

```c
#include <stdio.h>
#include <math.h>
 main()
{
 int XA, YA, XB,YB;
 double DIST;
 /* Remarqe: La chaîne de format que nous utilisons */
/* s'attend à ce que les données soient séparées par */
 /* une virgule lors de l'entrée. */
 printf("Donnez les coordonnées du point A :  XA,YA  ");
 scanf("%d,%d", &XA, &YA);
 printf("Donnez les coordonnées du point B :  XB,YB  ");
 scanf("%d,%d", &XB, &YB);
 DIST=sqrt(pow(XA-XB,2)+pow(YA-YB,2));
 printf("La distance entre A(%d,% d) et B(%d, %d) est %.2f\n",
XA, YA, XB, YB, DIST);
  return 0;
}
```

Exercice 7

Élaborez **un code écrit en langage C** qui enregistre un caractère depuis le clavier et le capture accompagné de son code numérique.

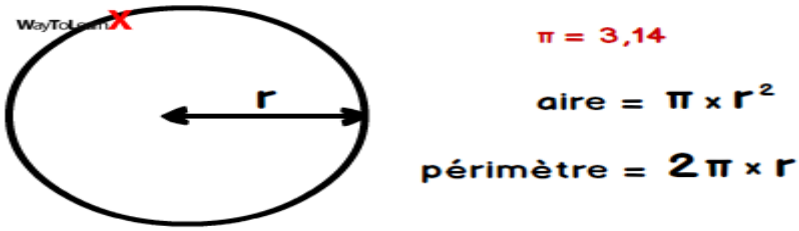
Correction

```c
#include <stdio.h>
main()
{
 int C;
 printf("Saisir un caractère suivi de 'Enter'\n");
 C = getchar();
  printf("Le caractère %c a le code ASCII %d\n", C, C);
 return 0;
}
```

Exercice 8

Élaborez **un programme en langage C** qui reçoit le rayon d'un cercle par l'utilisateur et

identifie l'aire et le périmètre du cercle.



Exemple :

Le rayon = 5

Le périmètre = 31.40 unités

L'aire = 78.50 unités

Correction

```c
#include <stdio.h>
int main()
{
    float r, area, perimeter;

    // Donner le rayon du cercle
    printf("Le rayon du cercle: ");
    scanf("%f", &r);
    //Identifier l'aire et le périmètre du cercle
    perimeter = 2 * 3.14 * r;
    area = 3.14 * (r * r);
    //Afficher le résultat
    printf("Le périmètre du cercle est = %.2f unités \n", perimeter
        );
    printf("L'aire du cercle est = %.2f unités", area);
    return 0;
}
```

TD 2 : Les branches conditionnelles et les boucles en langage C

Les structures alternatives

Exercice 1

En **langage C, élaborer un programme** qui saisit deux nombres entiers (X et Y) et qui affiche le signe du leurs produit sans multiplication.

Correction

```c
#include <stdio.h>
main()
{
 /* Afficher le signe du produit de deux entiers sans multiplication
     */
 int X, Y;
 printf("Donner deux nombres entiers :");
 scanf("%i %i", &X, &Y);
 if ((X>0 && Y>0) || (X<0 && Y<0))
   printf("Le signe du produit %i * %i est positif\n", X, Y);
 else if ((X<0 && Y>0) || (X>0 && Y<0))
   printf("Le signe du produit %i * %i est négatif\n", X, Y);
 else
   printf("Le produit %i * %i est zéro\n", X, Y);
  return 0;
}
```

Exercice 2

En **langage C, élaborer un programme** qui saisit deux nombres entiers (A et B) et qui capture le signe de leurs sommes sans effectuer l'addition. (Adopter la fonction **fabs** de la bibliothèque<*math.h*>).

Correction

```c
#include <stdio.h>
#include <math.h>
 main()
{
 /* Afficher le signe de la somme de deux entiers sans faire
     l'addition */
 int A, B;
 printf("Donner deux nombres entiers :");
 scanf("%i %i", &A, &B);
 if ((A>0 && B>0) || (A<0 && B>0 && fabs(A)<fabs(B))
                  || (A>0 && B<0 && fabs(A)>fabs(B)))
    printf("Le signe de la somme %i + %i est positif\n",A,B);
 else if ((A<0 && B<0) || (A<0 && B>0 && fabs(A)>fabs(B))
                       || (A>0 && B<0 && fabs(A)<fabs(B)))
    printf("Le signe de la somme %i + %i est négatif\n",A,B);
 else
    printf("La somme %i + %i est zéro\n", A, B);
  return 0;
}
```

Exercice 3

En **langage C, concevoir un programme** qui détermine les valeurs réelles satisfaisant une équation quadratique exprimé de manière $ax^2+bx+c = 0$.

Correction

```c
#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 degré de la
     forme ax^2 + bx + c = 0 \n\n");
 printf("Donner les valeurs de 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("l'équation n'a 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("l'équation n'a pas de solutions réelles.\n");
else if (D==0) /* b^2-4ac = 0 */
   {
    printf("l'équation a une seule solution réelle qui est :\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;
}
```

Les structures répétitives

Exercice 1

Effectuer la division entière de deux nombres et calculer le quotient suivi du reste en utilisant des soustractions successives.

Correction

```
#include <stdio.h>
main()
{
 int NUM;    /* numérateur de la division entière */
 int DEN;    /* dénominateur de la division entière */
 int DIV;    /* résultat de la division entière */
 int RES;    /* reste de la division entière */

 printf("Saisir le numérateur   : ");
 scanf("%d", &NUM);
 printf("Saisir le dénominateur : ");
 scanf("%d", &DEN);

 RES=NUM;
 DIV=0;
 while(RES>=DEN)
    {
     RES-=DEN;
     DIV++;
    }

/* ou mieux encore : */
/*
 for (RES=NUM, DIV=0 ; RES>=DEN ; DIV++)
      RES-=DEN;
 */

 printf(" %d divisé par %d est %d reste %d\n", NUM, DEN, DIV, RES);
 return 0;
}
```

Exercice 2

Déterminez le factoriel d'un nombre entier N! = 1*2*3*...*(N-1)*N en gardant à l'esprit que 0!=1.

a) Adoptez **while**,
b) Adoptez **for**.

Correction

a) en utilisant la boucle **while**,

```c
#include <stdio.h>
main()
{
 int  N;       /* La donnée   */
 int  I;       /* Le compteur */
 double FACT; /* N! - Type double à */

 do
    {
     printf("Donner un entier naturel : ");
     scanf("%d", &N);
    }
 while (N<0);

 /* a */
 /* Pour N=0, le résultat sera automatiquement 0!=1 */
 I=1;
 FACT=1;
 while (I<=N)
       {
        FACT*=I;
        I++;
       }

 printf ("%d! = %f\n", N, FACT);
 return 0;
}
```

b) en utilisant la boucle **for**,

```c
#include <stdio.h>
main()
{
 int  N;       /* La donnée   */
 int  I;       /* Le compteur */a
 double FACT; /* N! - Type double à */

 do
    {
     printf("Donner un entier naturel : ");
     scanf("%d", &N);
    }
 while (N<0);

for (FACT=1.0, I=1 ; I<=N ; I++)
       FACT*=I;

 printf ("%d! = %.0f\n", N, FACT);
 return 0;
}
```

Exercice 3

En utilisant les multiplications successives, calculez A^N de deux nombres entiers naturels spécifiés A et N.

Correction

```c
#include <stdio.h>
main()
{
 int  A, N;   /* Les données */
 int  i;      /* Le compteur */
 double RESU; /* Type double à cause de la */
 Do
    {
     printf("Donner la valeur A : ");
    scanf("%d", &A);
    }
 While (A<0);
 Do
    {
     printf("Donner l'exposant N : ");
    scanf("%d", &N);
    }
 While (N<0);

 /* Pour N=0, le résultat sera automatiquement A^0=1 */
 for (RESU=1.0, i=1 ; i<=N ; i++)
        RESU*=A;

 /* faire attention: Pour A=0 et N=0 , 0^0 n'est pas défini */
 if (N==0 && A==0)
    printf("zéro exposant zéro n'est pas défini !\n");
 else
    printf("Résultat : %d ^ %d = %.0f\n", A, N, RESU);
  return 0;
}
```

Exercice 4

Soit 1 + 1/2 + 1/3 + ... + 1/N, une série harmonique. Trouvez la somme des N premiers termes

Correction

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

Exercice 5

Employez l'algorithme d'Euclide pour trouver le PGCD de deux entiers naturels.

Correction

```c
#include <stdio.h>
main()
{
 int A, B;          /* données */
 int X, Y, RESTE;  /* var. d'aide pour l'algorithme d'Euclide */

 do
    {
     printf("Entrer A (non nul) : ");
     scanf("%d", &A);
    }
 while(!A);
 do
    {
     printf("Entrer B (non nul) : ");
     scanf("%d", &B);
    }
 while(!B);

 for (RESTE=A, X=A, Y=B ; RESTE ; X=Y, Y=RESTE)
     RESTE = X%Y;

 printf("Le PGCD de %d et de %d est %d\n", A, B, X);
 return 0;
}
```

TD 3 : Les tableaux en langage C

Exercice 1

Elaborer un **programme C** qui reçoit la taille M d'un tableau A du type **int** (50 composantes au maximum), charge le tableau par des entiers et qui l'affiche. Puis effectuer la somme de ses composants et présenter le résultat.

Correction

```c
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50]; /* tableau donné  */
 int M;     /* dimension    */
 int i;     /* indice courant */
 long Sum;  /* somme des éléments - type long à cause */
            /* de la grandeur prévisible du résultat. */

 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &M );
 for (i=0; i<M; i++)
    {
     printf("Elément %d : ", i);
     scanf("%d", &A[i]);
    }
  /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (i=0; i<M; i++)
     printf("%d ", A[i]);
 printf("\n");
  /* Calcul de la somme */
 for (Sum=0, i=0; i<M; i++)
     Sum += A[i];
  /* Edition du résultat */
 printf("Somme de éléments : %ld\n", Sum);
 return 0;
}
```

Exercice 2

En recevant la taille M d'un tableau A ayant comme dimension maximale 50, élaborer un programme C qui charge le tableau par des entiers et qui l'affiche. Puis effacer chaque apparition de 0 et préserver le reste en capturant le tableau résultant.

Correction

```c
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50]; /* tableau donné    */
 int M;     /* dimension    */
 int I,J;   /* indices courants */

 /* lit les données */
 printf("taille du tableau (max.50) : ");
 scanf("%d", &M );
 for (I=0; I<M; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &A[I]);
    }
 /* Affichage du tableau */
 printf("Tableau donné : \n");
 for (I=0; I<M; I++)
     printf("%d ", A[I]);
 printf("\n");
 /* Effacer les zéros et comprimer :        */
 /* Copier tous les éléments de I vers J et */
 /* augmenter J pour les éléments non nuls. */
```

```c
 for (I=0, J=0 ; I<M ; I++)
     {
      if (A[I])
      {
      A[J] = A[I];
      J++;
      }
     }
 /* Nouvelle dimension du tableau ! */
 M = J;
  /* Edition des résultats */
 printf("Tableau résultat :\n");
 for (I=0; I<M; I++)
     printf("%d ", A[I]);
 printf("\n");
 return 0;
}
```

Exercice 3

En **langage C, élaborer un programme** qui reçoit la taille M d'un tableau A du type **int** (50 composantes au maximum), charge le tableau par des entiers et qui l'affiche.

Sans utiliser de tableau d'aide, ensuite, organiser les éléments du tableau en ordre inverse et présenter le tableau obtenu.

Idée : changer les composants du tableau en utilisant deux repères qui explorent le tableau en débutant au commencement et à la fin du tableau dans l'ordre et qui s'intersectent en son milieu.

Correction

```c
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50]; /* tableau donné */
 int M;     /* dimension     */
 int I,J;   /* indices courants */
 int AUX;  /* pour l'échange   */

 /* Lit des données */
 printf("La taille du tableau (max.50) : ");
 scanf("%d", &M );
 for (I=0; I<M; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &A[I]);
    }
 /* Affichage du tableau */
 printf("Tableau donné : \n");
 for (I=0; I<M; I++)
     printf("%d ", A[I]);
 printf("\n");
 /* Inverser le tableau */
 for (I=0, J=M-1 ; I<J ; I++,J--)
      /* Permuter A[I] et A[J] */
        {
         AUX = A[I];
         A[I] = A[J];
         A[J] = AUX;
        }
  /* Affichage des résultats */
 printf("Tableau résultat :\n");
 for (I=0; I<M; I++)
     printf("%d ", A[I]);
 printf("\n");
 return 0;
}
```

Exercice 4

En **langage C, élaborer un programme** qui reçoit la taille M d'un tableau A d'entiers (50 composantes au maximum), alimente le tableau avec des valeurs saisies depuis le clavier et le présente.

Ensuite, recréez un second tableau appelé APS, contenant chaque composante strictement positive, et un troisième tableau nommé ANG, comprenant chaque composante strictement négative. Enfin, affichez les tableaux APS et ANG.

Correction

```c
#include <stdio.h>
main()
{
  /* Déclarations */
  /* Les tableaux et leurs dimensions */
  int A[50], APS [50], ANG[50];
  int M,     MPS,     MNG;
  int I; /* indice courant */

  /* Saisie des données */
  printf("Dimension du tableau (max.50) : ");
  scanf("%d", &M );
  for (I=0; I<M; I++)
     {
      printf("Elément %d : ", I);
      scanf("%d", &A[I]);
     }
  /* Affichage du tableau */
  printf("Tableau donné :\n");
  for (I=0; I<M; I++)
      printf("%d ", A[I]);
  printf("\n");
  /* Initialisation des dimensions de APS et ANG */
  MPS=0;
  MNG=0;
  /* Transfer des données */
  for (I=0; I<M; I++)
      { if (A[I]>0) {
                     APS[MPS]=A[I];
                     MPS++;
                    }
        if (A[I]<0) {
                     ANG[MNG]=A[I];
                     MNG++;
                    }
      }
   /* Edition du résultat */
  printf("Tableau APS :\n");
  for (I=0; I<MPS; I++)
      printf("%d ", APS[I]);
  printf("\n");
  printf("Tableau ANG :\n");
  for (I=0; I<MNG; I++)
      printf("%d ", ANG[I]);
  printf("\n");
  return 0;
}
```

Exercice 5

En C, élaborer un programme permettant de concevoir un tableau d'entiers A à deux dimensions L et C. Charger ce tableau et le capturer puis calculer la somme de ses valeurs.

Correction

```c
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50][50]; /* tableau donné */
 int L, C;  /* dimensions        */
 int I, J;  /* indices courants  */
 long som;  /* somme des éléments - type long à cause */
            /* de la grandeur prévisible du résultat. */

 /* Lit des données */
 printf("Nombre de lignes   (max.50) : ");
 scanf("%d", &L );
 printf("Nombre de colonnes (max.50) : ");
 scanf("%d", &C );
 for (I=0; I<L; I++)
    for (J=0; J<C; J++)
        {
         printf("Elément[%d][%d] : ",I,J);
         scanf("%d", &A[I][J]);
        }
  /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (I=0; I<L; I++)
    {
     for (J=0; J<C; J++)
         printf("%7d", A[I][J]);
     printf("\n");
    }
  /* Calcul de la somme  */
 for (SOM=0, I=0; I<L; I++)
     for (J=0; J<C; J++)
         som += A[I][J];
  /* Edition du résultat */
 printf("Somme des éléments : %ld\n", som);
 return 0;
}
```

TD 4 : Les chaînes de caractères

Exercice 1

Développer un programme qui recueille cinq mots, saisis séparément par l'utilisateur, les affiche en une ligne, puis effectue une inversion de leur ordre initial.

Exemple :

voici une petite phrase !

! phrase petite une voici

Correction

```c
#include <stdio.h>
main()
{
 char A[30], B[30], C[30], D[30], E[30];
 printf("Tapez 5 mots :\n");
 scanf ("%s %s %s %s %s",  A, B, C, D, E);
 printf("%s %s %s %s %s\n",E, D, C, B, A);
  return 0;
}
```

Exercice 2

Concevoir un programme qui recueille une phrase (limitée à 200 caractères) saisie par l'utilisateur, la stocke dans une variable CH, puis affiche par la suite :

1. La taille L de la chaîne.
2. Le nombre de 'e' dans le texte.
3. Afficher la phrase complète sans modifier la valeur stockée dans la variable CH.
4. Afficher la phrase complète suite au changement de la séquence des caractères dans CH.

Correction

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

 /* Donner les données */
 printf("Donner une phrase :\n");
 gets(CH); /* L'utilisation de scanf est impossible pour */
 /* reçoit une phrase contenant un nombre variable de mots. */
```

```c
/* 1) Compter les caractères */
for (L=0; CH[L]; L++)
    ;
printf("Le texte est composé de %d caractères.\n",L);
/* 2) Compter les lettres 'e' dans le texte */
C=0;
for (i=0; CH[i]; i++)
    if (CH[i]=='e') C++;
printf("Le texte contient %d lettres \'e\'.\n",C);

/* 3) Afficher la phrase à l'envers */
for (i=L-1; i>=0; i--)
    putchar(CH[i]);  /* ou printf("%c",CH[i]); */
putchar('\n');         /* ou printf("\n"); */

/* 4) Inverser l'ordre des caractères */
for (i=0,j=L-1 ; i<j ; i++,j--)
    {
    aux=CH[i];
    CH[i]=CH[j];
    CH[j]=aux;
    }
puts(CH);  /* ou printf("%s\n",CH); */
 return 0;
}
```

Exercice 3

Editer en langage C le nom du jour de la semaine correspond à un nombre entre 1 et 7.

Correction

```c
#include <stdio.h>
main()
{
    int N;
    char jours[8][9]={"Erreur!","Lundi","Mardi","Mercredi","Jeudi"
        ,"Vendredi","Samedi","Dimanch "};
    do
    {
        printf("Entrer un nombre entre 1 et 7 : ");
        scanf("%d",&N);
    }
    while((N>7)&&(N<0));
    if((N>0)&&(N<8))
        printf("Le %d eme jour de la semaine correspond a %s.\n",N,jours[N]
            );
    else
        puts(jours[0]);
  return 0;
}
```

Exercice 4

En utilisant la fonction **strlen**, élaborer un programme C qui reçoit le nom et le prénom d'un utilisateur et opère la longueur totale du nom en excluant les espaces.

Correction

```c
#include <stdio.h>
#include <string.h>
 main()
{
  char NOM[40], PRENOM[40];
  printf("Introduisez votre nom et votre prénom: \n");
  scanf("%s %s", NOM, PRENOM);
  printf("\nBonjour %s %s !\n", NOM, PRENOM);
  printf("Votre nom est composé de %d lettres.\n",
                                  strlen(NOM) + strlen(PRENOM));
 /* ou bien
  printf("Votre nom est composé de %d lettres.\n",
                                  strlen(strcat(NOM,PRENOM)));
 */
 return 0;
}
```

Exercice 5

En introduisant deux chaînes de caractères CH1 et CH2, répliquer la moitié initiale de CH1 et de CH2 dans une troisième chaîne CH3, puis capture le résultat.

1. Appliquer les fonctions de <*string*>.

Correction

```c
#include <stdio.h>
#include <string.h>
 main()
{
 /* Déclarations */
 char CH1[100], CH2[100]; /* chaînes données */
 char CH3[100]="";        /* chaîne résultat */

 /* Saisie des données */
 printf("Ecrire la première chaîne de caractères : ");
 gets(CH1);
 printf("Ecrire la deuxième chaîne de caractères : ");
 gets(CH2);
 /* Traitements */
 strncpy(CH3, CH1, strlen(CH1)/2);
 strncat(CH3, CH2, strlen(CH2)/2);
  /* Affichage du résultat */
 printf("Un demi \"%s\" plus un demi \"%s\" donne \"%s\"\n",CH1, CH2, CH3);
 return 0;
}
```

TD 5 : Les pointeurs

Exercice 1

Nous disposons d'un pointeur P qui est dirigé vers le tableau T:

int T[] = {12, 23, 34, 45, 56, 67, 78, 89, 90};

int *P;

P = T;

-Pour chaque expression, donner les valeurs ou adresses fournis :

1. ***P+2**
2. ***(P+2)**
3. **&T[4]-3**
4. **T+3**
5. **&T[7]-P**
6. **P+(*P-10)**
7. ***(P+*(P+8)-T[7])**

Correction

```
1.  *P+2              => la valeur 14
2.  *(P+2)            => la valeur 34
3.  &T[4]-3           => l'adresse de la composante T[1]
4.  T+3               => l'adresse de la composante T[3]
5.  &T[7]-P           => la valeur (indice) 7
6.  P+(*P-10)         => l'adresse de la composante T[2]
7.  *(P+*(P+8)-T[7]) => la valeur 23
```

Exercice 2

Elaborer un programme C qui saisit un **nombre entier X** et un tableau d'**entiers** A et extermine toutes les instances de X dans A en écrasant tous les composants qui demeurent. Afin d'explorer le tableau, adopter les **pointeurs** P1 et P2.

Correction

```c
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50];    /* tableau donné   */
 int N;        /* dimension du tableau */
 int X;        /* valeur à éliminer  */
 int *P1, *P2; /* pointeurs aideur  */
```

```
/* Saisie des données */
 printf("taille du tableau (maximum 50) : ");
 scanf("%d", &N );
 for (P1=A; P1<A+N; P1++)
     {printf("Elément %d : ", P1-A);
      scanf("%d", P1);}
  printf("Introduire l'élément X à éliminer du tableau : ");
 scanf("%d", &X );
 /* Affichage du tableau */
 for (P1=A; P1<A+N; P1++)
     printf("%d ", *P1);
  printf("\n");
 /* Supprimer toutes les occurrences de X et comprimer :  */
 /* Copier tous les éléments de P1 vers P2 et augmenter */
 /* P2 pour tous les éléments différents de X. */
 for (P1=P2=A; P1<A+N; P1++)
     {
      *P2 = *P1;
      if (*P2 != X)
          P2++;
     }
  /* Nouvelle taille de A */
 N = P2-A;
  /* Edition du résultat */
 for (P1=A; P1<A+N; P1++)
     printf("%d ", *P1);
  printf("\n");
 return 0;
}
```

Exercice 3

En **langage C, inverser les séquences des composants d'un tableau d'entiers**. Pour transposer les composants, adopter des pointeurs P1 et P2 et une variable de type numérique nommée AUX.

Correction

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int B[50];    /* tableau donné */
 int M;        /* dimension du tableau */
 int AUX;     /* pour la permutation  */
 int *P1, *P2; /* pointeurs d'aide     */
 /* Saisie des données */
```

```
printf("Dimension du tableau (max.50) : ");
 scanf("%d", &M );
 for (P1=B; P1<B+M; P1++)
     {
      printf("Elément %d : ", P1-B);
      scanf("%d", P1);
     }
  /* Affichage du tableau */
 for (P1=B; P1<B+M; P1++)
     printf("%d ", *P1);
  printf("\n");
 /* Inverser le tableau */
 for (P1=B,P2=B+(M-1); P1<P2; P1++,P2--)
     {
      AUX = *P1;
      *P1  = *P2;
      *P2  = AUX;
     }
  /* Edition du résultat */
 for (P1=B; P1<B+M; P1++)
      printf("%d ", *P1);
  printf("\n");
 return 0;
}
```

Exercice 4

En utilisant les pointeurs p1 et p2, taper la fonction echange(int *p1, int *p2) qui permet d'échanger la valeur des deux paramètres effectifs de l'appel.
Exemple d'appel : echange(&i, &j).

Correction

```
void echange(int *p1, int *p2)
{
    int temp;
    temp=*p1;
    *p1=*p2;
    *p2=temp;
}

#include <stdio.h>
int main()
{
        int a=5,b=10;
    printf("valeurs de a et b avant la permutation : %d %d\n", a, b);
    echange(&a, &b);
    printf("valeurs de a et b après la permutation : %d %d\n", a, b);
return 0;
}
```

Exercice 5

Pour un tableau T de dimension n, taper une fonction int* max(int T[taille], int m) qui donne comme résultat un pointeur sur le plus grand élément de ce tableau.

Correction

```
int* max(int T[taille], int m)
{
    int i;
    int* resultat=T;
    for(i=0;i<m;i++)
    {
        if(T[i]>*resultat)
            resultat=&T[i];
    }
    return resultat;
}

#include <stdio.h>
#define taille 10
int main()
{
       int nbElts,i;
    do
    {
        printf("Donner le nombre d'élements \n");
        scanf("%d",& nbElts);
    }
    while ((nbElts >taille) || (nbElts <0));

    int A[taille];
    for(i=0;i< nbElts;i++)
    {
        printf("Donner l'élement num %d: \n",i+1);
        scanf("%d",&A[i]);
    }

    int* resultat=max(A, nbElts);
    printf("l'élément le plus grand est %d.\n", *resultat);
return 0;
}
```

TD 6 : Les fonctions

Exercice 1

Donnez une fonction puissance pour le processus de détermination de la puissance d'un nombre réel p par un nombre entier n. (n et p : paramètres).

Correction

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

int puissance(int a,int b)
{
    return(pow(a,b));
}


void main()
{
    int n,p,res;
    do{
    printf("Donner un entier une puissance");
    scanf("%d %d",&n,&p);
    }while(p<0);

    res=puissance(n,p);
    printf("%d",res);

}
```

Exercice 2

Trouver, dans l'ordre, le **minimum** et le **maximum** de deux entiers, en employant deux fonctions **min** et **max**.
En recourant à ces deux fonctions, élaborer un programme qui calcule le maximum et le minimum de 4 entiers.

Correction

```c
#include <stdio.h>

float fmin(float x,float y)
{
    if(x<y) return x;
    return y;
}

float fmax(float x,float y)
{
    if(x>y) return x;
    return y;
}
void main()
{

    float a,b,c,d,min,max;
    printf("Donner 4 réel \n");
    scanf("%f %f %f %f",&a,&b,&c,&d);

    min=fmin(fmin(a,b),fmin(c,d));

    max=fmax(fmax(a,b),fmax(c,d));

    printf("le max est %f \n",max);
    printf("le min est %f \n",min);
}
```

Exercice 3 :

1. Donner une fonction **premier** d'un paramètre entier m et de retour TRUE si le nombre est premier et FALSE dans le cas contraire.
2. Donner une fonction **prochain_premier** en recevant un paramètre entier n et retourne le plus petit nombre premier plus grand ou égal à n.
3. Elaborer un programme qui donne un entier n d'un utilisateur et capture le premier nombre premier plus grand ou égal à n.

Correction

```c
#include <stdlib.h>
#include <stdio.h>
#define TRUE 1
#define FALSE 0
/* Fonction testant si un nombre est premier */
int premier(int m)
{
int i ;
for(i = 2; i < m; i = i + 1)
if ((m % i) == 0) /* divisible */
return FALSE;
return TRUE;
}
/* Fonction cherchant le premier nb premier plus grand que n */
int prochain premier(int n)
{
while(! premier(n))
n = n + 1;
return n;
}
/* La fonction principale */
void main(void)
{
int k;
printf ("Entrez un entier positif ");
scanf("%d", &k);
printf ("Le prcohain nombre premier de %d est %d\n", k, prochain premier(k));
}
```

Exercice 4 :

Donnez une fonction utilisée pour obtenir la somme suivante :

$$S=\sum_{i=0}^{n} x^i/i!$$ (x et n étant des valeurs passés en paramètres).

Utiliser la fonction puissance et celle de factoriel.

Correction

```c
#include <math.h>
int puissance(int a,int b)
{
    return(pow(a,b));
}
int factorielle(int a)
    {
        int fact=1,i=1;
        if(a==0) return 1;
        while(i<=a)
        {
            fact=fact*i;
            i++;
        }
        return fact;
    }
    float somme(int n,int x)
    {
        int i=0;
        float somm=0;
        while(i<=n)
        {
            somm=somm+((float)puissance(x,i)/(float)factorielle(i));
            i++;
        }
        return somm;
    }
void main()
{ int n,x;
    float som;
    do{
        printf("Donner n et x \n");
        scanf("%d %d",&n,&x);
    }while(n<0);
    som=somme(n,x);
    printf("SOMME= %f ",som); }
```