Université Moulay Ismail

Faculté des sciences et techniques d'Errachidia

Département d'informatique

Module : Structure de données et programmation avancée en C

Compte rendu des Travaux Pratique :

Réalise par :

-BAHIDA ANWAR

Responsable :

- Prof : FRHAWI YOUSSEF

- Prof : M. BOUTAHIR

Année universitaire: 2023/2024

Sommaire :

Serie1 :

Exercice 1 :

```
#include<stdio.h>
int main(){
    float a,b;
    int m;
    char n ;
    do{
        printf("Menu: \n +: Addition\n -: Soustraction\n *: Multiplication\n /: Division\ choisir une operation: \n");

        scanf(" %c",&n);
        printf("Donner a: ");
        scanf("%f",&a);
        printf("Donner b: ");
        scanf("%f",&b);
        switch(n){
            case '+':
            printf("%.2f + %.2f = %.2f",a,b,a+b);
            break;
            case '-':
            printf("%.2f - %.2f = %.2f",a,b,a-b);
            break;
            case '*':
            printf("%.2f * %.2f = %.2f",a,b,a*b);
            break;
            case '/':
            printf("%.2f / %.2f = %.2f",a,b,a/b);
```

```
            break;
            default:
            printf("Ereur");
            break;
        }
        printf("\nVoulez-vous continuer ?\n 1:Oui\n 0:Non\n ");
        scanf("%d",&m);
    }while(m==1);
}
```

Exercice 2 :

```
#include<stdio.h>
int main()
{
   int age;
   char sexe;
   printf("entrer votre age:");
   scanf("%d",&age);
   printf("entrer 'H' si vous etes homme et 'F' si vous etes famme :");
   scanf(" %c",&sexe);
   if(sexe == 'H' || sexe == 'F'){
      if((sexe == 'H') &&(age > 20)||(sexe == 'F')&&(age >18 && age <35)){
         printf("vous etes imposable !!!");
      }
      else{
         printf("vous n\'est pas imposable");
      }
   }
}
```

Exercice 3 :

```
#include<stdio.h>
int main(){
  int NbrPho;
  float Prix;
  printf("Donner le nombre de photocopie: ");
  scanf("%d",&NbrPho);
  if(NbrPho <= 10){
    printf("Vous devez paier %d DH",NbrPho);
  }
  if(NbrPho >=10 && NbrPho <=30){
    Prix = 10 + 0.6 * (NbrPho - 10);
    printf("Vous devez paier %.2f DH",Prix);
  }
  if(NbrPho > 30){
    Prix = 10 + 0.6*20 + 0.4*(NbrPho - 30);
    printf("Vous devez paier %.2f DH",Prix);
  }
}
```

Exercice 4 :

```
#include<stdio.h>
int main(){
    float PrixIni, PrixFin, Rem;
    printf("Donner le prix initial du produit: ");
    scanf("%f",&PrixIni);
    if(PrixIni<100){
       Rem = (30*PrixIni)/100;
    }
    if(PrixIni>=100 && PrixIni<=200){
       Rem = (40*PrixIni)/100;
    }
    if(PrixIni>200){
```

```
        Rem = (50*PrixIni)/100;
    }
    PrixFin = PrixIni - Rem;
    printf(" Prix Initial: %.2f DH\n Remise: %.2f DH\n Prix Final: %.2f
DH",PrixIni,Rem,PrixFin);
}
```

Exercice 5 :

```
#include<stdio.h>
int main(){
   int n,i,cmp=0;
   printf("entrer un nombre : ");
   scanf("%d",&n);
   for(i=1;i<=n;i++){
      if(n%i ==0){
         cmp++;
      }
   }
   if(cmp==2)
      printf("le nombre %d est premier ",n);
   else
      printf("le nombre %d n'est pas premier ",n);

}
```

Exercice 6

```
#include<stdio.h>
int main(){
   int N,i;
   printf("entrer un nombre : ");
   scanf("%d",&N);
```

```
    if(N % 2 == 0){
      printf("les nombres pairs qui lui sont inferieur a %d sont: \n",N);
        for(i=N;i>0;i--){
           if(i % 2 == 0){
              printf("%d \n",i);
           }
        }
    }
}
```

Exercice 7 :

```
#include<stdio.h>
int main(){
// La reponse du question 1
    printf("Question 1\n");
    int n,i;
    for(i=1;i<=5;i++){
    printf("donner le nombre numero %d: ",i);
    scanf("%d",&n);
    if(n%2==0){
    printf("Le carre de %d est %d\n",n,n*n);
    }
}
// La reponse du question 2
    printf("\nQuestion 2\n");
    int S=0,SP=0;
    do{
       printf("Donner un nombre: ");
       scanf("%d",&n);
       S = S + n;
       if(n%2==0){
           printf("Le carre de %d est %d\n",n,n*n);
```

```
            SP = SP + n;
        }
    }while(n!=100);
    printf("Le nombre total d'entrees est: %d\n",S);
    printf("Le nombre d'entrees paires est: %d",SP);
}
```

Exercice 8 :

```
#include<stdio.h>
int main(){
// La reponse du question a
    printf("Question a\n\n");
    int i,j;
    for(j=6;j>=1;j--){
        for(i=j;i>=1;i--){
            printf("%d",i);
        }
        printf("\n");
    }
// La reponse du question b
    printf("\nQuestion b\n\n");
    int k;
    for(i=0;i<10;i++){
        for(j=0;j<i;j++){
            printf(" ");
        }
        for(k=10;k>i;k--){
            printf("%d",i);
         }
         printf("\n");
    }
}
```

Exercice 9 :

```
#include<stdio.h>
int main(){
    int n, cmp=1, S, SP, min, minPo;
    printf("Donner un nombre: ");
    scanf("%d",&n);
    min=n;
    S=n;
    if(n>0){
       minPo=n;
       SP=n;
    }
    if(n<0) minPo=999;
    if(n==999) printf("Suite vide");
    else{
       do{
            printf("Donner un nombre: ");
            scanf("%d",&n);
            if(n==999) break;
            else{
              cmp++;
              S = S + n;
              if(n < min)
                  min=n;
              if(n>0)
                  SP = SP + n;
              if(n>0 && n < minPo)
                  minPo=n;
            }
        }while(n != 999);
        printf("Le nombre total de valeurs de la suite: %d\n",cmp);
```

```
        printf("La somme des valeurs lues: %d\n",S);
        printf("Le minimum: %d\n",min);
        printf("La somme des valeurs strictement positives: %d\n",SP);
        printf("Le minimum des valeurs strictement positives: %d\n",minPo);
    }
}
```

Exercice 10 :

```
#include<stdio.h>
int main(){
    int i,k,n;
    printf("Saisir le nombre des lignes: ");
    scanf("%d",&n);
    for(i=1;i<=n;i++){
       for(k=1;k<2*n;k++){
          if(k==n-(i-1)|| k==n+(i-1)|| i==n)
                printf("*");
          else printf(" ");
       }
       printf("\n");
    }
}
```

Exercice 11 :

```
#include<stdio.h>
int main(){
    int n=0,choix;
    printf("Donner un entier: ");
    scanf("%d",&n);
    printf("La valeur de l'entier est: %d\n",n);
```

```
    do{
        printf("\n 1.Ajouter 2\n 2.Multiplier par 3\n 3.Soustraire 5\n 4.Quitter\nChoisir entre 1 et
4: ");
        scanf("%d",&choix);
        switch(choix){
            case 1:
            printf("La nouvelle valeur est: %d\n",n+2);
            break;
            case 2:
            printf("La nouvelle valeur est: %d\n\n",n*3);
            break;
            case 3:
            printf("La nouvelle valeur esr: %d\n\n",n-5);
            break;
            case 4:
            break;
        }
    }while(choix!=4);
}
```

Exercice 12 :

```
#include<stdio.h>
int main(){
    int i=1,k,n ;
    printf("Saisir la taille: ");
    scanf("%d",&n);
    while(i<=n){
    for(k=1;k<=n;k++){
        if((i==1 || i==n) || (k==1 || k==n))
                printf("* ");
        else if(k==i ||k==n-i+1 )
                printf("+ ");
        else if(i==(n/2)+1)
```

```
                printf("| ");
        else if (k==(n/2)+1)
                printf("- ");
            else
                printf(" ");
    }
    printf("\n");
    i++;
    }
}
```

Serie2 :

EXE 1 :

```
#include <stdio.h>

#define Nmax 20

void saisie(int tableau[], int *nbElements) {
    int i;
    printf("Entrez le nombre d'éléments (max %d): ", Nmax);
    scanf("%d", nbElements);

    printf("Entrez les %d entiers:\n", *nbElements);
    for (i = 0; i < *nbElements; i++) {
        printf("Element %d: ", i + 1);
        scanf("%d", &tableau[i]);
    }
}
```

```
void affichage(int tableau[], int nbElements) {
    int i;
    printf("Tableau: ");
    for (i = 0; i < nbElements; i++) {
        printf("%d ", tableau[i]);
    }
    printf("\n");
}

float moyenne(int tableau[], int nbElements) {
    int i;
    float somme = 0;
    for (i = 0; i < nbElements; i++) {
        somme += tableau[i];
    }
    return somme / nbElements;
}

int max_elem(int tableau[], int nbElements) {
    int i, max = tableau[0], pos = 0;

    for (i = 1; i < nbElements; i++) {
        if (tableau[i] > max) {
            max = tableau[i];
            pos = i;
        }
    }

    return pos;
}

void supprimer_max(int tableau[], int *nbElements) {
    if (*nbElements == 0) {
        printf("Le tableau est vide.\n");
```

```
        return;
    }

    int pos = max_elem(tableau, *nbElements);

    for (int i = pos; i < *nbElements - 1; i++) {
        tableau[i] = tableau[i + 1];
    }
    (*nbElements)--;
}

int min_elem(int tableau[], int nbElements) {
    int i, min = tableau[0], pos = 0;

    for (i = 1; i < nbElements; i++) {
        if (tableau[i] < min) {
            min = tableau[i];
            pos = i;
        }
    }

    return pos;
}

void supprimer_min(int tableau[], int *nbElements) {
    if (*nbElements == 0) {
        printf("Le tableau est vide.\n");
        return;
    }

    int pos = min_elem(tableau, *nbElements);

    for (int i = pos; i < *nbElements - 1; i++) {
        tableau[i] = tableau[i + 1];
```

```
    }

    (*nbElements)--;
}

void ajout(int tableau[], int *nbElements) {
    int entier, position;

    printf("Entrez l'entier à ajouter: ");
    scanf("%d", &entier);

    printf("Entrez la position d'insertion (de 1 à %d): ", *nbElements + 1);
    scanf("%d", &position);

    if (position < 1 || position > *nbElements + 1) {
        printf("Position invalide.\n");
        return;
    }

    for (int i = *nbElements; i >= position; i--) {
        tableau[i] = tableau[i - 1];
    }

    tableau[position - 1] = entier;
    (*nbElements)++;
}

int main() {
    int tableau[Nmax];
    int nbElements = 0;
    char choix;

    do {
        printf("\nMenu de choix:\n");
```

```
printf("A- Saisie et affichage\n");
printf("B- Moyenne\n");
printf("C- Suppression du Max et affichage\n");
printf("D- Suppression du Min et affichage\n");
printf("E- Ajout d'un entier à une position donnée\n");
printf("Q- Quitter\n");

printf("Choisissez une option : ");
scanf(" %c", &choix);

switch (choix) {
    case 'A':
        saisie(tableau, &nbElements);
        affichage(tableau, nbElements);
        break;
    case 'B':
        printf("Moyenne: %.2f\n", moyenne(tableau, nbElements));
        break;
    case 'C':
        supprimer_max(tableau, &nbElements);
        affichage(tableau, nbElements);
        break;
    case 'D':
        supprimer_min(tableau, &nbElements);
        affichage(tableau, nbElements);
        break;
    case 'E':
        ajout(tableau, &nbElements);
        affichage(tableau, nbElements);
        break;
    case 'Q':
        printf("Programme terminé.\n");
        break;
    default:
```

```
            printf("Option invalide. Veuillez réessayer.\n");
        }
    } while (choix != 'Q');

    return 0;
}
```

EXE 2 :

```
#include <stdio.h>

// Définition de la fonction affiche_matrice
void affiche_matrice(int matrice[5][5]) {
    printf("Matrice:\n");
    for (int i = 0; i < 5; i++) {
        for (int j = 0; j < 5; j++) {
            printf("%d ", matrice[i][j]);
        }
        printf("\n");
    }
}

int main() {
    // Appel de la fonction affiche_matrice pour la matrice iMat

int iMat[5][5] ;
  printf(‘’ remplir la matrice : ’’);
  for (int i = 0; i < 5; i++) {
          for (int j = 0; j < 5; j++) {
                printf(‘’ iMat [%d][ %d]’’,i,j);
    }
}
    affiche_matrice(iMat);
```

```
    return 0;
}
```

EXE 3 :

```
#include <stdio.h>

// Déclaration du tableau iNb_jours
int iNb_jours[12];

// Prototypes des fonctions
void initialiser_iNb_jours();
void afficher_iNb_jours();
void afficher_nb_jours_mois(int mois);

int main() {
    // Appel des procédures d'initialisation et d'impression
    initialiser_iNb_jours();
    afficher_iNb_jours();

    // Variantes possibles :
    // afficher_nb_jours_mois(3); // Affiche le nombre de jours du mois 3
    // Affichage pour un mois saisi par l'utilisateur
    int mois_utilisateur;
    printf("Entrez un mois  : ");
    scanf("%d", &mois_utilisateur);
    do {
        afficher_nb_jours_mois(mois_utilisateur);
    }while (mois_utilisateur <1 || mois_utilisateur >12) ;

    return 0;
}
```

```
// Définition de la procédure d'initialisation
void initialiser_iNb_jours() {
    for (int i = 0; i < 12; i++) {
        if (i == 1) {
            iNb_jours[i] = 28;
        } else if ((i % 2 == 0 && i <= 7) || (i % 2 != 0 && i > 7)) {
            iNb_jours[i] = 30;
        } else {
            iNb_jours[i] = 31;
        }
    }
}

// Définition de la procédure d'impression des 12 valeurs utiles de iNb_jours

void afficher_iNb_jours() {
    printf("Nombre de jours par mois :\n");
    for (int i = 0; i < 12; i++) {
        printf("Mois %d : %d jours\n", i + 1, iNb_jours[i]);
    }
}

// Définition de la procédure d'impression pour un mois donné
void afficher_nb_jours_mois(int mois) {
    if (mois >= 1 && mois <= 12) {
        printf("Le mois %d a %d jours.\n", mois, iNb_jours[mois - 1]);
    } else {
        printf("Mois invalide.\n");
    }
}
```

EXE 4 :

```
#include <stdio.h>
// Définition de la fonction longueur_chaine
int longueur_chaine(char chaine[]) {
    int longueur = 0;
    // Boucle pour compter les caractères jusqu'au null
    while (chaine[longueur] != '\0') {
        longueur++;
    }
    return longueur;
}

int main() {
    char cTab1[10];
    char cTab2[10] ;
    int longueur1, longueur2 ;
    // Appel de la fonction longueur_chaine pour cTab1 et cTab2
    longueur1 = longueur_chaine(cTab1);
    longueur2 = longueur_chaine(cTab2);

    // Affichage des résultats
    printf("Longueur de cTab1 : %d\n", longueur1);
    printf("Longueur de cTab2 : %d\n", longueur2);

    return 0;
}
```

EXE 5 :

```
#include <stdio.h>

// Définition de la fonction de cryptage
void crypt(char *caractere) {
    // Vérification si le caractère est une lettre majuscule
    if ( *caractere >= 'A' &&  *caractere <= 'Z') {
```

```
        // Cryptage des lettres majuscules
         *caractere = ((*caractere - 'A' + 5) % 26) + 'A';
    }
    // Vérification si le caractère est une lettre minuscule
    else if (*caractere >= 'a' && *caractere <= 'z') {
        // Cryptage des lettres minuscules
        *caractere = ((*caractere - 'a' + 5) % 26) + 'a';
    }
    // Ne rien faire pour les autres caractères (ponctuation, etc.)



int main() {

    char cMessage[100] ;
    // Boucle pour crypter chaque caractère du message
    for (int i = 0; cMessage[i] != '\0'; i++) {
        crypt(&cMessage[i]);
    }

    // Affichage du message crypté
    printf("Message crypté : %s\n", cMessage);

    return 0;
}
```

EXE 6 :

```
#include <stdio.h>

int main() {
    char caractere;
    int chiffres = 0, espacements = 0, autres = 0;
    printf("Entrez une séquence de caractères :\n");
    while ((caractere = getchar()) != EOF) {
```

```
        if (caractere >= '0' && caractere <= '9') {
            chiffres++;
        } else if (caractere == ' ' || caractere == '\t' || caractere == '\n') {
            espacements++;
        } else {
            autres++;
        }
    }

    printf("Chiffres : %d\n", chiffres);
    printf("Espacements : %d\n", espacements);
    printf("Autres : %d\n", autres);

    return 0;
}
```

EXE 7 :

```
#include <stdio.h>
// Définition des fonctions d'opérations
int addition(int a, int b) {
    return a + b;
}
int soustraction(int a, int b) {
    return a - b;
}
int multiplication(int a, int b) {
    return a * b;
}
int division(int a, int b) {
    if (b != 0) {
        return a / b;
    } else {
```

```
        printf("Division par zéro impossible.\n");
        return 0; // Retourne 0 en cas de division par zéro
    }
}
int modulo(int a, int b) {
    if (b != 0) {
        return a % b;
    } else {
        printf("Opération modulo avec zéro impossible.\n");
        return 0; // Retourne 0 en cas d'opération modulo avec zéro
    }
}
int main() {
    char choix;
    do {

        int nombre1, nombre2;
        char operateur;

        printf("Entrez une expression (ex: 5  +  3) : ");
        scanf("%d %c %d", &nombre1, &operateur, &nombre2);

        // Calcule de la valeur de l'expression
        int resultat;
        switch (operateur) {
            case '+':
                resultat = addition(nombre1, nombre2);
                break;
            case '-':
                resultat = soustraction(nombre1, nombre2);
                break;
            case '*':
                resultat = multiplication(nombre1, nombre2);
                break;
```

```
            case '/':
                resultat = division(nombre1, nombre2);
                break;
            case '%':
                resultat = modulo(nombre1, nombre2);
                break;
            default:
                printf("Opérateur invalide.\n");
                return 1; // Sortie du programme en cas d'opérateur invalide
        }
        // Affichage du résultat
        printf("Résultat : %d\n", resultat);

        printf("Voulez-vous recommencer ? (O/N) : ");
        scanf(" %c", &choix);
    } while (choix == 'O' || choix == 'o');
    return 0;
}
```

EXE 8 :

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

// Définition de la fonction de comparaison de longueur
int compareLongueur(char *nom) {
    return strlen(nom);
}
// Définition de la fonction de lecture d'un nom
void lireNom(char *nom) {
    printf("Entrez un nom (ou 'fin' pour terminer) : ");
    scanf("%s", nom);
}
```

```
int main() {
    char noms[20][21]; // Tableau de 20 noms de 20 caractères maximum

    int nombreNomsPlusDeDixCaracteres = 0;

    // Boucle de lecture des noms
    for (int i = 0; i < 20; i++) {
        lireNom(noms[i]);

        // Vérification de la fin de la saisie
        if (strcmp(noms[i], "fin") == 0) {
            break;
        }

        // Comparaison de la longueur du nom
        if (compareLongueur(noms[i]) > 10) {
            nombreNomsPlusDeDixCaracteres++;
        }
    }

    // Affichage du nombre de noms ayant plus de dix caractères
    printf("Nombre de noms ayant plus de dix caractères : %d\n",
nombreNomsPlusDeDixCaracteres);

    return 0;
}
```

EXE 10 :

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

// Fonction de saisie
```

```
void saisir(char *chaine, int longueur) {
    printf("Saisissez une chaine de caractères : ");
    fgets(chaine, longueur, stdin);
    chaine[strcspn(chaine, "\n")] = '\0';  // Supprimer le saut de ligne
}
// Fonction d'affichage
void afficher(const char *chaine) {
    printf("Chaine saisie : %s\n", chaine);
}
// Fonction d'inversion
void inverser(char *chaine) {
    int longueur = strlen(chaine);
    for (int i = 0, j = longueur - 1; i < j; i++, j--) {
                char temp = chaine[i];
                chaine[i] = chaine[j];
                chaine[j] = temp;
    }
}

// Fonction de comptage de mots
int compterMots(const char *chaine) {
    int nombreMots = 0;
    int enMot = 0;

    for (int i = 0; chaine[i] != '\0'; i++) {
        if (chaine[i] == ' ' || chaine[i] == '\t' || chaine[i] == '\n') {
            enMot = 0;
        }
        else if (enMot == 0) {
                enMot = 1;
                nombreMots++;
            }
    }
    return nombreMots;
```

```
}

int main() {
    char chaine[100];
    int choix;

    do {
        // Affichage du menu
        printf("\nMenu :\n");
        printf("1. Saisir une chaine\n");
        printf("2. Afficher la chaine\n");
        printf("3. Inverser la chaine\n");
        printf("4. Compter les mots\n");
        printf("5. Quitter\n");
        printf("Choix : ");
        scanf("%d", &choix);

        switch (choix) {
            case 1:
                saisir(chaine, sizeof(chaine));
                break;
            case 2:
                afficher(chaine);
                break;
            case 3:
                inverser(chaine);
                printf("La chaine a été inversée.\n");
                break;
            case 4:
                printf("Nombre de mots dans la chaine : %d\n", compterMots(chaine));
                break;
            case 5:
                printf("Programme terminé.\n");
                break;
```

```
            default:
                printf("Choix invalide. Veuillez choisir une option valide.\n");
        }

        // Attendre une touche pour revenir au menu
        printf("Appuyez sur une touche pour revenir au menu...");
        getchar(); // Attendre une touche
        getchar(); // Attendre une autre touche pour continuer

    } while (choix != 5);

    return 0;
}
```

EXE 11 :

Question 1 :

```
#include <stdio.h>
int distanceH(char *S1, char *S2, int length) {
    int hammingDistance = 0;
    for (int i = 0; i < length; i++) {
        if (S1[i] != S2[i]) {
            hammingDistance++;
        }
    }
    return hammingDistance;
}

int main() {
    char str1[100] ;
    char str2[100];
    int longeur ;
```

```
   printf(“entrer str1 : ”);
   gets(str1) ;
  printf(“entrer str2: ”);
  gets(str2) ;
  while(strlen(str1)==strlen(str2)) ;
{
      longeur = strlen(str1);
      int result = distanceH(str1, str2, longeur);
      printf("Distance de Hamming entre \"%s\" et \"%s\" est %d\n", str1, str2, result);
}
   return 0;
}
```

Question 2 :

```
#include <stdio.h>

int distanceH_langage(char **langage, int nbStrings, int length) {
   int minDistance = length;  // Initialisation à la valeur maximale possible
   for (int i = 0; i < nbStrings - 1; i++) {
      for (int j = i + 1; j < nbStrings; j++) {
         int currentDistance = distanceH(langage[i], langage[j], length);
         if (currentDistance < minDistance) {
            minDistance = currentDistance;
         }
      }
   }
   return minDistance;
}

int main() {
   int nbStrings , i , j ;
   int length ;
```

```
    printf(“entrer le nombre de caractere : ”);
    scanf(“%d“,& nbStrings) ;
   printf(“entrer la longeur : ”);
   scanf(“%d“,& length) ;
    char langage[nbStrings][ length]
   for(i=0 ;i< nbStrings ;i++){
         for(j=0 ;j< length ;j++){
                printf(”langage[%d][ %d]”,i,j);
                scanf(“%d”, langage[i][ j]);
         }
}
  int result = distanceH_langage(langage, nbStrings, length);
  printf("Distance de Hamming du langage est %d\n", result);

    return 0;
}
```

Question 3 :

```
#include <stdio.h>

char* binaire(int N) {
    char* binary = malloc(9);  // 8 bits + '\0'
    for (int i = 7; i >= 0; i--) {
        binary[7 - i] = ((N > i) & 1) + '0';
    }
    binary[8] = '\0';
    return binary;
}

int distanceNombre(int A, int B) {
    char* binaryA = binaire(A);
    char* binaryB = binaire(B);
```

```
    int hammingDistance = 0;
    for (int i = 0; i < 8; i++) {
        if (binaryA[i] != binaryB[i]) {
            hammingDistance++;
        }
    }
    free(binaryA);
    free(binaryB);
    return hammingDistance;
}

int main() {
    int num1;
    int num2;
    printf("entrer le premier num :") ;
    scanf("%d",&num1) ;
    printf("entrer le deuxieme num :") ;
    scanf("%d",&num2) ;

    int result = distanceNombre(num1, num2);
    printf("Distance de Hamming entre %d et %d est %d\n", num1, num2, result);

    return 0;
}
```

Serie3 :

Exercice 1 :

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

```
#include<stdlib.h>
#include<string.h>
int main(){
                char *nom, *d, *f; char tmp; int n;
                nom = (char*)calloc(30,sizeof(char));
                printf("Donner un nom : ");
                gets(p);
                n = strlen(p);
                d = nom;
                f = nom+n-1;
                 while( d <= f ){
                          tmp = *d;
                         *d = *f;
                         *f = tmp;
                          d++;
                          f--;
                 }
                printf("Le nom inverse est: %s",nom); }
```

Exercice 2 :

```
#include<stdio.h>
 #include<stdlib.h>
 int longeur(char *p){
          int cmp=0;
         while(*p != '\0'){
                    cmp++;
                    p++;
          }
         return cmp;
 }
char *inverse(char *p){
          char *n;
          int i,j;
          n = (char*)malloc(longeur(p)*sizeof(char));
          for(i=longeur(p); i>0; i--) {
               for(j=0;j<longeur(p); j++){
                   *(n+j) = *(p+i-1);
                   *(n+longeur(p))='\0';
               }
           }

          return n;
}
int main(){
          char *mot1, *mot2;
          mot1 = (char*)malloc(30*sizeof(char));
           printf("Donner un nom: ");
           mot2 = inverse(mot1);
           if(*mot1 == *mot2)
              printf("Le mot saisis est palindrome");
           else
                printf("Le mot saisis n'est pas palindroma");
}
```

Exercice 3 :

```
#include<stdio.h>
#include<stdlib.h>
int i,j;
float Somme_form_usuel(float A[3][4]){
            float Som=0;
            for(i=0; i<3; i++){
                    for(j=0; j<4; j++) {
                            Som += A[i][j];
                    }
            }
            return Som;
}
float Somme_form_poin(float **A, int l, int c){
             float Som=0;
             for(i=0; i<l; i++)
                    for(j=0; j<c; j++)
                          Som += *(*(A+i)+j);
 return Som;
}
int main(){
        float t[3][4];
        float somme;
        int i,j ;
        // Declaration dynamique de la matrice float **T;
        T= (float**)malloc(3*sizeof(float*));
        for( i=0; i<4; i++)
          *(C+i) = (float*)malloc(4*sizeof(float));
        // initilisation du matrice par lecture au clavier
        for( i=0; i<3; i++)
             for(j=0; j<4; j++){
                 printf("t[%d][%d] = ",i,j);
                 scanf("%f",&t[i][j]);
        }
        // Remplissage du tableau dynamique par les pointeurs
        for(i=0; i<3; i++)
             for(j=0; j<4; j++)
                    *(C+j) = *(t+i);
        // Affchage de la matrice
        for(i=0; i<3; i++){
              for(j=0; j<4; j++)
                    printf("%.2f \t",*(*(t+i)+j));
              printf("\n");
         }
       //Calcule de la somme en utilisant le formalisme usuel des tabeaux
       somme = Somme_form_usuel(t);
       printf("En utilisant le formalisme usuel des tabeaux: \nLa somme des elements du
tableau est: %.2f   \n",somme);
       //Calcule de la somme en utilisant le formalisme de pointeurs
       somme = Somme_form_poin(T,3,4);
       printf("En utilisant le formalisme de pointeurs: \nLa somme des elements du tableau est:
%.2f        \n",somme);
}
```

Exercice 4 :

```
#include<stdio.h>

int i,j;

void afficher(int A[3][5]){
        for(i=0; i<3; i++){
                for(j=0; j<5; j++)
                        printf("%d\t",A[i][j]);
        } printf("\n");
}
int main(){
        int b[3][5];
        int *a=*b;
        for(i=0 ; i<15 ; *a++ = i++){
                afficher(b);
                **b=15;
                afficher(b);
                 **(b+1)=16;
                 afficher(b);
                *(b[0]+1)=17;
                 afficher(b);
                *(*b+8)=18;
                  afficher(b);
                 *(b[1]+2)=19;
                 afficher(b);
                *(*(b+1)+5)=20;
                afficher(b);
                *(b[2]+3)=21;
                afficher(b);
                *(*(b+2)+2)=22;
                 afficher(b);
        }
```

Exercice 5 :

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int nbr_occ(char *c, char *mot){
    int resultat =0;
    char *p=c;
    while( (p=strstr(p,mot)) != NULL){
        resultat++;
        p += strlen(mot);
     }
     return resultat;
}
int main(){
      char *ph, *mot;
      int resultat;
      c = (char*)malloc(250*sizeof(char));
      mot = (char*)malloc(15*sizeof(char));
      printf("Donner une phrase: ");
      gets(c);
```

```
        printf("Donner un mot: ");
        gets(mot);
        resultat = nbr_occ(c,mot);
        printf("\nLe nombre d'occurence de (%s) dans la phrase %s est: %d",mot,c,resultat);
}
```

Exercice 6 :

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
char* suprimer_blanc(char *phrase, char *p){
            while(*p == ' ')
                    p++;
             return p;
}
int main(){
        char * phrase;
        ph = (char*)malloc(250*sizeof(char));
        printf("Donner une phrase: ");
        gets(phrase);
        char *q= phrase;
        printf("%s",suprimer_blanc(phrase,q) );
 }
```

Exercice 7 :

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

int search(int T[],int taill,int va_cher){
   int i,cmp=0,pos_valeur;
   for(i=0;i<taill;i++){
          if(T[i]== va_cher){
             pos_valeur=i+1;
             cmp++;
          }
   }
   if (cmp==0)
      return -1;
   else
      return pos_valeur;
}
void saisi(int T[],int taill){
   int i;

   for(i=0;i<taill;i++){
      printf("\nT[%d]=",i);
      scanf("%d",&T[i]);
   }
}
void affiche(int T[],int taill){
   int i;
   for(i=0;i<taill;i++)
```

```
        printf("\n%d",T[i]);

}
int main(){
        int taill;
        int pos_valeur;
        int va_cher;
        printf("entrer le taill de tableau : ");
        scanf("%d",&taill);
        int T[taill];
        printf("Remplir le tableau : ");
        saisi(T,taill);
        printf("\naffichage de tableau : ");
        affiche(T,taill);
        printf("\nentrer la valeur pour le chercher dans le tableau :");
        scanf("%d",&va_cher);
        pos_valeur=search(T,taill,va_cher);
        printf("\nla position de %d est : %d ",va_cher,pos_valeur);
        return 0;
}
```

Exercice 8 :

```
 #include <stdio.h>

void compressRLE(int tableau[], int taille) {
    int i = 0;
    int Valeur, count ;
    while (i < taille) {
        valeur = tableau[i];
        count = 1;
        // Compter le nombre d'occurrences de la valeur
        while (i < taille - 1 && tableau[i] == tableau[i + 1]) {
            count++;
            i++;
        }

        // Afficher l'encodage RLE

        printf("%d%d", count, valeur);
        i++;
    }
    printf("\n");
}

int main() {
    int taille;
    int I;
    printf("Entrez la taille du tableau : ");
    scanf("%d", &taille);

    int tableau[taille];

    // Saisie les valeurs dans tableau
    printf("Entrez les valeurs du tableau d'entiers :\n");
```

```
    for (i = 0; i < taille; i++) {
        printf("Valeur %d : ", i + 1);
        scanf("%d", &tableau[i]);
    }

    // Appel de la fonction de compression RLE
    printf("compression RLE : ");
    compressRLE(tableau, taille);
    return 0;
}
```

Serie4 :

Exercice 1:

```
#include<stdio.h>

 typedef struct Complex{
          int reel;
          int img;
 } Complex;
double imag(Complex z){
           return z.img;
 }
double reel(Complex z){
           return z.reel;
 }
Complex mul(Complex z1, Complex z2){
           Complex Nbr_Comp;
           Nbr_Comp.reel = ( reel(z1)*reel(z2) - imag(z1)*imag(z2) ) ;
           Nbr_Comp.img = ( reel(z1)*imag(z2) + imag(z1)*reel(z2) );
           return Nbr_Comp;
 }
int main(){
       Complex z1, z2, w;
       printf("Donner la partie real de z1: ");
       scanf("%d",&z1.reel);
       printf("Donner la partie imaginaire de z1: ");
       scanf("%d",&z1.img);
       printf("Donner la partie real de z2: ");
       scanf("%d",&z2.reel);
       printf("Donner la partie imaginaire de z2: ");
       scanf("%d",&z2.img);
       printf("z1 = %.2lf + %.2lf i\n", reel(z1), imag(z1) );
       printf("z2 = %.2lf + %.2lf i\n",reel(z2), imag(z2) );
```

```
    w = mul(z1,z2);
    printf("z1 x z2 = %.2lf + %.2lf i", reel(w), imag(w));
}
```

Exercice 2 :

```
#include<stdio.h>
#include<string.h>
struct date{
    int jour;
    char mois[10];
    int annee;
};
struct employe{
        char nom[15], prenom[15];
        struct date date_naissance, date_embauche;
}Employe;
int main(){
    int n,i;
    printf("Donner le nombre des employes: ");
    scanf("%d",&n);
    Employe Em[n];
    printf("Veuillez saisir les informations des employes: \n");
    for(i=1; i<n; i++){
            printf("Employe %d \n",i+1);
            printf("Nom: ");
            scanf(" %s",Em[i].nom);
            printf("Prenom: ");
            scanf(" %s",Em[i].prenom);
            printf("Date de naissance (jour): ");
            scanf("%d",&Em[i].date_naissance.jour);
            printf("Date de naissance (mois): ");
            scanf(" %s",Em[i].date_naissance.mois);
            printf("Date de naissance (anne): ");
            scanf("%d",&Em[i].date_naissance.annee);
            printf("Date d'embauche (jour): ");
            scanf("%d",&Em[i].date_embauche.jour);
            printf("Date d'embauche (mois): ");
            scanf(" %s",Em[i].date_embauche.mois);
            printf("Date d'embauche (anne): ");
            scanf("%d",&Em[i].date_embauche.annee);
            printf("\n");
    }
            printf("Les informations des employes sont: \n");
            for(i=1; i<=n; i++){
                printf("Employe %d \n",i+1);
                printf("Nom: %s\n",Em[i].nom);
                printf("Prenom: %s\n",Em[i].prenom);
                printf("Date de naissance: %d / %s / %d\n", Em[i].date_naissance.jour,
                Em[i].date_naissance.mois, Em[i].date_naissance.annee);
                printf("Date d'embauce: %d / %s / %d\n", Em[i].date_embauche.jour,
                Em[i].date_embauche.mois, Em[i].date_embauche.annee);
                printf("\n");
            }
```

}

Exercice 3 :

```
#include<stdio.h>
 #include<string.h>
typedef struct etudiant{
        char nom[15], prenom[15];
        int CNE;
        float notes[4], moyenne;
 };
 int main(){
                int n, i, j,tmp;
                float S, moy, NOTES[4]; int cne;
                char NOM[15], PRENOM[15];
                printf("Donner le nombre des etudiants: ");
                scanf("%d",&n);
                etudiant T[n];
// Lecture des informations des etudiants
                 printf("\n\n");
                 printf("Veuillez saisir les informations des etudiants: \n");
                 for(i=0; i<n; i++){
                        S = 0;
                        printf("Etudiant %d \n",i+1);
                        printf("Nom: ");
                        scanf(" %s",T[i].nom);
                        printf("Prenom: ");
                        scanf(" %s",T[i].prenom);
                        printf("CNE: ");
                        scanf("%d",&T[i].CNE);
                        for(j=0; j<4; j++){
                                printf("Note[%d] = ",j+1);
                                scanf("%f",&T[i].notes[j]);
                                S = S + T[i].notes[j];
                         }
                        T[i].moyenne = S / 4;
                         printf("\n");
                }
// Affichage des informations des etudiants
                printf("\n\n");
                printf("Les informations saisis sont: \n");
                for(i=0; i<n; i++){
                        printf("Etudiant %d \n",i+1);
                        printf("Nom: %s \n",T[i].nom);
                        printf("Prenom: %s \n",T[i].prenom);
                        printf("CNE: %d \n",T[i].CNE);
                        for(j=0; j<4; j++) {
                                printf("Note[%d] = %.2f \n", j+1, T[i].notes[j]);
                        }
                        printf("Moyenne = %.2f \n", T[i].moyenne);
                        printf("\n");
                }
                for(i=0; i<n; i++){
                        if( T[i].moyenne > moy ){
                                for(j=0; j<15; j++){
```

```
                                        NOM[j] = T[i].nom[j];
                                        PRENOM[j] = T[i].prenom[j];
                                    }
                                cne = T[i].CNE;
                                for(j=0; j<4; j++)
                                        NOTES[j] = T[i].notes[j];
                                        moy = T[i].moyenne;
                        }
                }

                printf("\n\n");
                printf("la plus grand moyenne est de l'etudiant %s, ces informations sont: \n",
NOM);
                printf("Nom: %s \n",NOM);
                printf("Prenom: %s \n",PRENOM);
                printf("CNE: %d \n",cne);
                for(i=0; i<4; i++)
                        printf("Note[%d] = %.2f \n",i+1,NOTES[i]);
                        printf("Moyenne = %.2f \n", moy);
// Trie du Tableau selon la moyenne
                for(i=0; i<n; i++) {
                        for(j=i; j<n; j++){
                                if(T[j].moyenne < T[i].moyenne ){
                                        tmp= T[i].moyenne;

                                        T[i].moyenne = T[j].moyenne;

                                        T[j].moyenne = tmp;

                                    }
                        }
                }
```

Exercice 4 :

```
#include<stdio.h>
typedef struct Repertoire{
        char nom[15], prenom[15];
        int Tele;
}Repertoire ;
void Afficher(Repertoire A[], int t){
         int i, j;
         printf("Les informations saisis sont: \n");
         for(i=0; i<t; i++){
                  printf(" Enregistrement  %d \n",i+1);
                  printf("Nom: %s \n",A[i].nom);
                  printf("Prenom: %s \n",A[i].prenom);
                  printf("Tele: %d \n",A[i].Tele);
         }
}
int main(){
         int i, j, n;
         printf("Donner le nombre des enregistrements : ");
         scanf("%d",&n);
```

```
        Repertoire T[n];
        // Lecture des donnees
        printf("\n\nVeuillez saisir les donnees: \n");
        for(i=0; i<n; i++){
                printf("Enregistrement %d  \n",i+1);
                printf("Nom: ");
                scanf(" %s",T[i].nom);
                printf("Prenom: ");
                scanf(" %s",T[i].prenom);
                printf("Tele: ");
                scanf("%d",&T[i].Tele);
                printf("\n");
         }
        Afficher(T,n);
}
```

Exercice 5 :

```
#include<stdio.h>
#include<string.h>
int nb_occurrence(char T[], int n, char c){
        int i, cmp=0, p_occ, d_occ;
        for(i=0; i<n; i++){
                if(T[i] == c){
                        p_occ = i;
                        break;
                }

          }
          printf("p_occ = %d \n", p_occ);
          for(i=n; i>0; i--){
               if(T[i] == c){
                      d_occ = i;
                      break;
               }
           }
          printf("d_occ = %d \n", d_occ);
          for(i=0; i<n; i++){
                   if(T[i] == c)
                        cmp++;
                  return cmp;
          }
}
int main(){
        int t;
        char h;
        printf("Donner la taille du tableau: ");
        scanf("%d",&t);
        char A[t];
        printf("Saisir les caracteres: ");
        scanf(" %s",A);
        printf("Donner le caracter a cherche: ");
        scanf(" %c",&h);
        printf("Le nombre des occuences est: %d",nb_occurrence(A, t, h));
}
```

Exercice 6 :

```
#include<stdio.h>
typedef struct Panneau{
        float lar, lon, epa;
        int type;
}Panneau;
void Saisie(Panneau P[], int t){
        int i;
       printf("\n\nVeuillez saisir les informations des panneaux: \n");
       for(i=0; i<t; i++){
              printf("Panneau %d \n",i+1);
              printf("Largeur: ");
              scanf("%f",&P[i].lar);
              printf("Longeur: ");
              scanf("%f",&P[i].lon);
              printf("Epaisseur: ");
              scanf("%f",&P[i].epa);
              do{
                     printf("Type du bois: \n 0: Pin \n 1: Chene \n 2: Hetre \n ");
                     scanf("%d",&P[i].type);
               }while(P[i].type != 0 && P[i].type != 1 && P[i].type != 2);
                printf("\n");
        }
}
void Afficher(Panneau P[], int t){
       int i;
       printf("\n\nLes informations saisis sont: \n");
       for(i=0; i<t; i++){
              printf(" Panneau %d  \n",i+1);
              printf("Largeur: %.2f mm \n",P[i].lar);
              printf("Longeur: %.2f mm \n",P[i].lon);
              printf("Epaisseur: %.2f mm \n",P[i].epa);
              printf("Type du bois : ");
              if( P[i].type == 0)
                     printf("Pin \n");
              else
                    if( P[i].type == 1)
                           printf("Chene \n");
               else
                     if( P[i].type == 2)
                            printf("Hetre \n");
                      printf("\n");
          }
}
float Calcul_Volume(Panneau P){
        float Volume;
        Volume = P.lar * P.lon * P.epa;
        return Volume;
}
int main(){
       int n,i;
       printf("Donner le nombre des panneaux: ");
       scanf("%d",&n);
```

```
        Panneau T[n];
        Saisie(T,n);
        Afficher(T,n);
        do{
                printf("Donner le numero de panneau souhaiter pour calculer le volume: ");
                scanf("%d",&i);
        }while(i<=0 || i>n);
        printf("Le Volume de cette panneau Numero %d est: %.2f m³ \n",i,
        Calcul_Volume(T[i-1]) );
}
```

Exercice 7 :

```
#include <stdio.h>

int i, j;

typedef struct Produit {
   int Ref, Type, Qte;
   float Prix;
} Produit;

void Saisie_Affichage(Produit P[4]) {
   int i ;
   printf("Veuillez saisir les donnees: \n");
   for (i = 0; i < 4; i++) {
      printf("<------ Produit %d ------> \n", i + 1);
      printf("Reference: ");
      scanf("%d", &P[i].Ref);
      do {
         printf("Type du produit: \n 1: Cartes meres \n 2: Processeurs \n 3: Barettes memoire \n
4: Carte graphique \n----> ");
         scanf("%d", &P[i].Type);
      } while (P[i].Type != 1 && P[i].Type != 2 && P[i].Type != 3 && P[i].Type != 4);
      printf("Quantite: ");
      scanf("%d", &P[i].Qte);
      printf("Prix: ");
      scanf("%f", &P[i].Prix);
      printf("\n");
   }
   printf("\n\nLes informations saisies sont: \n");
   for (i = 0; i < 4; i++) {
      printf("<------ Produit %d ------> \n", i + 1);
      printf("Reference: %d \n", P[i].Ref);
      printf("Type du produit: ");
      if (P[i].Type == 1) printf("Carte mere \n");
      else if (P[i].Type == 2)
            printf("Processeur \n");
      else if (P[i].Type == 3)
            printf("Barrettes memoire \n");
      else if (P[i].Type == 4)
             printf("Cartes graphique \n");
      printf("Quantite: %d \n", P[i].Qte);
      printf("Prix: %.2f DH \n", P[i].Prix);
      printf("\n");
```

```
    }
}

void Commande(Produit T[4]) {
    int type, qte;
    printf("\n\nSaisir votre commande : \n");
    do {
        printf("choisir le produit que vous voulez . \n 1: Cartes meres \n 2: Processeurs \n 3:
Barettes memoire \n 4: Carte graphique \n----> ");
        scanf("%d", &type);
    } while (type != 1 && type != 2 && type != 3 && type != 4);
    printf("Quelle est la quantite ? \n----> ");
    scanf("%d", &qte);
    for (j = 0; j < 4; j++) {
        if (T[j].Type == type) {
            while (T[j].Qte < qte) {
                printf("la quantite disponible est insuffisante ! \n");
                printf("Quelle est la quantite ? \n----> ");
                scanf("%d", &qte);
            }
            printf("<------ Facture ------> \n");
            printf(" (: Notre boutique souhaite la bienvenue :) \n");
            printf("Le produit demandé est: ");
            if (T[j].Type == 1) printf("Carte mere \n");
            else if (T[j].Type == 2)
                    printf("Processeur \n");
            else if (T[j].Type == 3)
                    printf("Barrettes memoire \n");
            else if (T[j].Type == 4)
                    printf("Cartes graphique \n");
            printf("---> La Reference du produit: %d \n", T[j].Ref);
            printf("---> Le prix total est: %.2f DH \n", qte * T[j].Prix);
            T[j].Qte -= qte;  // Mettre à jour la quantité disponible
            break;  // Sortir de la boucle une fois que le produit est trouvé
        }
    }
}

int main() {
    Produit T[4];
    Saisie_Affichage(T);
    printf("Voulez-vous effectuer une commande ? \n 1: Oui \n 0: Non \n----> ");
    scanf("%d", &i);
    if (i == 1)
        Commande(T);
    return 0;

}
```

Exercice 8 :

```
#include<stdio.h>

#include<stdlib.h>

typedef struct TypeTableau {
```

```
    int nb_elem;
    int *tab;
} TypeTableau;
TypeTableau Creation_Tab(int n) {
    TypeTableau T;
    T.nb_elem = n;
    T.tab = (int*)malloc(T.nb_elem * sizeof(int));
    return T;
}

void Simple_Lecture_Tab(TypeTableau T) {
    int i;
    printf("\nVeuillez remplir les elements du tableau: \n");
    for (i = 0; i < T.nb_elem; i++) {
        printf("T[%d] = ", i);
        scanf("%d", &T.tab[i]);
    }
}

void Affichage(TypeTableau T) {
    int i;
    printf("Les elements du tableau sont: \n");
    for (i = 0; i < T.nb_elem; i++)
        printf("%d\t", T.tab[i]);
}

TypeTableau Double_Tab(TypeTableau T) {
    int i;
    TypeTableau M;
    M.nb_elem = T.nb_elem;
    M.tab = (int*)malloc(M.nb_elem * sizeof(int));
    for (i = 0; i < M.nb_elem; i++)
        M.tab[i] = 2 * T.tab[i];
    return M;
```

```
}

void Destruction_Tab(TypeTableau T) {
    free(T.tab);
}

int main() {
    TypeTableau A;
    int n;
    printf("Donner la taille du tableau: ");
    scanf("%d", &n);
    A = Creation_Tab(n);
    Simple_Lecture_Tab(A);
    Affichage(A);
    A = Double_Tab(A);
    printf("\n  Apres le calcule du double: \n");
    Affichage(A);
    Destruction_Tab(A);

    return 0;
}
```