UNIVERSITE D'ANTANANARIVO

MENTION MATHEMATIQUES ET INFORMATIUQUE

TD COMMUN

ALGORITHME ET PROGRAMMATION EN C

RANDRIANOMENJANAHARY Rado Abraham

Exercice 01 : Écrire un programme permettant d'afficher le message "Bonjour".

Algorithme

```
1   Algorithme Bonjour
2   Debut
3       Ecrire("Bonjour Les  Matheux!")
4   Fin
```

Programme Avec C et Python

Langage C

Bienvenue | exercice_01.cpp

exercice_01.cpp

```
1   #include <stdio.h>
2   int main() {
3       printf("Bonjour Les  Matheux!"); return 0 ;
4   }
5
```

Langage Python

test.py

```
1   print("Bonjour Les  Matheux")
```

Exercice 02 : Écrire un programme en C permettant de saisir deux nombres et d'afficher leur produit

```
1   Algorithme produit_deux_nombres
2   Variables m,n,p :réels
3   Debut
4       Ecrire(" Entrer un nombre:")
5       Lire(m)
6       Ecrire(" Entrer un nombre:")
7       Lire(n) p ← m * n
8       Ecrire("Le produit de ",m,"et",n,"=",p)
9   Fin
```

Langage C

exercice_01.cpp

```c
#include <stdio.h>
int main() {
    float m,n,p ;
    printf(" Entrer un nombre:");
    scanf("%f",&m);
    printf(" Entrer un nombre:");
    scanf("%f",&n);
    p = m*n ;
    printf("le produit de %f et %f = %f",m,n,p);

    return 0 ;
}
```

Langage PYTHON

test.py

```python
m = float(input(" Entrer un nombre:"))
n = float(input(" Entrer un nombre:"))
p = m*n
print("le produit de",m,"et",n,"=",p)

```

Exercice 03 : Écrire un programme en C qui permet d'échanger le contenu de deux entiers A et B saisis par l'utilisateur. et afficher ces entiers après l'échange

```
Algorithme Echange
Variables A, B, auxilaire : entiers
Debut
    Ecrire("Entrer un entier A:")
    Lire (A)
    Ecrire("Entrer un entier B:")
    Lire (B)
    auxilaire ← A
    A ← B
    B ← auxilaire
    Ecrire("le contenu de A est:",A)
    Ecrire("le contenu de B est:",B)
Fin
```

Langage C

exercice_01.cpp

```c
#include <stdio.h>
int main()
{
    int A,B ,auxilaire;
    printf("Entrer un entier A:");
    scanf("%d",&A);
    printf("Entrer un entier B:");
    scanf("%d",&B);
    auxilaire = A ;
    A = B ;
    B = auxilaire ;
    printf("le contenu de A est:%d\n",A);
    printf("le contenu de B est:%d",B);
    return 0 ;
}
```

LANGAGE PYTHON

test.py

```python
m = int(input(" Entrer un Entier m :"))
n = int(input(" Entrer un Entrer n :"))
print("Avant l'echange: m =",m,"et n =",n)
m,n=n,m
print("Apres l'echange: m =",m,"et n =",n)
```

Exercice 04 : Écrire un programme qui permet d'afficher si un nombre entier saisi au clavier est pair ou impair.

```
Algorithme  Parité
Variables n :entier
Debut
  Ecrire("Entrer un entier:")
  Lire(n)
  Si(n mod 2 = 0) alors
     Ecrire(n,"est pair")
    SiNon
     Ecrire(n,"est impair")
  FinSi
Fin
```

LANGAGE C

exercice_01.cpp

```c
#include <stdio.h>
int main() {
    int n ;
    printf("Entrer un entier:");
    scanf("%d",&n);
    if (n % 2 ==  0) {
        printf("%d est pair",n);
    }else{
        printf("%d est impair",n);
    }
    return 0;
}

```

LANGAGE PYTHON

test.py

```python
N = int(input("Entrer un Entier :"))
if N%2==0 :
    print(N,"est pair")
else :
    print(N,"est impair")

```

Exercice5 : Écrire un programme en C qui permet d'afficher le plus grand de trois entiers saisis au clavier.

```
Algorithme  Maximum_Trois_nombres
Variables A,B,C,Max :entiers
Debut
  Ecrire(" Entrer A:")  Lire(A)
  Ecrire(" Entrer B:")  Lire(B)
  Ecrire(" Entrer C:")  Lire(C)
   Max ← A
  Si (B >= Max) alors
     Max ← B
  FinSi

  Si(C >= Max) alors
     Max ← C
  FinSi
 Ecrire("Le Max est",Max)
Fin
```

Langage C

exercice_01.cpp

```c
#include <stdio.h>
int main() {
int A,B,C,Max;
printf(" Entrer A:"); scanf("%d",&A);
printf(" Entrer B:"); scanf("%d",&B);
printf(" Entrer C:"); scanf("%d",&C);
    Max = A ;
   if (B >= Max)
       Max = B ;
   if (C >= Max)
     Max = C ;

 printf("le Max est %d",Max);

return 0;
}

```

Langage Python

test.py

```python
A = int(input("Entrer un Entier A :"))
B = int(input("Entrer un Entier B :"))
C = int(input("Entrer un Entier C :"))
Max = A
if B >= Max :
    Max = B
if C >= Max :
    Max = C
print("Le maximum est ",Max)
```

Exercice6 : Écrire un programme en C qui permet d'évaluer une note saisi au clavier (si la note est supérieur à 10 alors il affiche validé sinon non validé (NB : la note comprise entre 0 et 20).

test.algo

```
Algorithme
Variables note :réel
Debut
    Ecrire(" Entrer la note : ")
    Lire (note)

    Si(note < 0  ou  note > 20) alors
      Ecrire("Erreur !!" )
    FinSi
    Si( note >= 0  et note < 10) alors
      Ecrire(" non Validé" )
    FinSi
    Si( note >= 10  et note <= 20) alors
      Ecrire("Validé" )
Fin
```

Langage C

exercice_01.cpp

```c
#include <stdio.h>
int main() {
float note ;
printf(" Entrer la note : ");
scanf("%f",&note);

if ( note < 0  || note > 20)
     printf("Erreur !!" );
if ( note >= 0  && note < 10)
     printf(" non Valide" );
if ( note >= 10  && note <= 20)
     printf("Valide" );
return 0;
}
```

Langage Python

test.py

```python
note = float(input("Entrer la note:"))
if note <= 20 and note >= 10 :
    print("Validé ")
elif note < 10 and note >= 0 :
    print("non Validé ")
else :
    print("La note incorrecte !!! ")
```

Exercice 7 : Une boutique propose à ces clients, une réduction de 15% pour les montants d'achat supérieurs à 200.000.000 Ar. Écrire un programme en C permettant de saisir le prix total HT et de calculer le montant TTC en prenant en compte la réduction et la TVA=20%.

test.algo

```
Algorithme
Variables Prix_HT,Prix_TTC: réels
Debut
   Ecrire(" Entrer le montant HT:")
   Lire(Prix_HT)

   Prix_TTC ← Prix_HT + Prix_HT*0.2

    Si( Prix_TTC > 200 ) alors
       Prix_TTC ← Prix_TTC - Prix_TTC*0.15
       Ecrire("le montant TTC est: ",Prix_TTC)
     SiNon
       Ecrire("le montant TTC est:",Prix_TTC)
   FinSi
Fin

```

Langage C

exercice_01.cpp

```c
#include <stdio.h>
int main() {
float Prix_HT , Prix_TTC ;
printf(" Entrer le montant HT:");
scanf("%f", &Prix_HT);
Prix_TTC = Prix_HT + Prix_HT*0.2 ;
  if ( Prix_TTC > 200 ) {
     Prix_TTC = Prix_TTC - Prix_TTC*0.15;
     printf("le montant TTC est: %f ",Prix_TTC);
  }else{
    printf("le montant TTC est: %f ", Prix_TTC);
  }
return 0;
}

```

Langage PYTHON

test.py

```python
Prix_HT = float(input("Entrer le montant HT:"))
Prix_TTC = Prix_HT + Prix_HT*0.2
if  Prix_TTC > 200 :
    Prix_TTC = Prix_TTC - Prix_TTC*0.15
    print("le montant TTC est: ",Prix_TTC)
else:
    print("le montant TTC est: ",Prix_TTC)

```

Exercice 08 : Écrire un programme en C qui demande l'âge d'un enfant et permet d'informer de sa catégorie sachant que les catégories sont les suivantes:

"poussin de 6 a 7 ans"

"pupille de 8 a 9 ans "

"minime de 10 a 11 ans "

"cadet après 12 ans ".

test.algo

```
Algorithme
Variables age :réel
Debut
   Ecrire("Entrer votre age :")
   Lire(age)

    Si( age >= 5  et  age <= 7) alors
      Ecrire(" Vous etes poussin ")
    FinSi
    Si( age >= 8  et  age <= 9 ) alors
       Ecrire(" Vous etes pupille ")
    FinSi
    Si( age >= 10  et  age <= 11) alors
       Ecrire(" Vous etes minime ")
    FinSi
    Si ( age >= 12 ) alors
       Ecrire(" Vous etes Cadet ")
    FinSi
Fin
```

Langage C

exercice_01.cpp

```
#include <stdio.h>
int main() {
float age;
printf(" Entrer votre age :");
scanf("%f",&age);

  if( age >= 6  &&  age <= 7)
  printf(" Vous etes poussin ");
  if( age >= 8  &&  age <= 9 )
  printf(" Vous etes pupille ");

  if( age >= 10  &&  age <= 11)
  printf(" Vous etes minime ");

  if( age >= 12 )
  printf(" Vous etes Cadet ");
return 0;
}

```

test.py

```
age = float(input("Entrer votre age :"))

if age >=12 :
    print("Vous etes cadet")
elif age >=10 and age <=11 :
    print("Vous etes minime")
elif age >=8 and age <=9 :
    print("Vous etes pupille")
elif age >=6 and age <=7 :
    print("Vous etes poussin")
else:
    print("votre age inférieur à 6 ans")
```

Exercice 09 : Écrire un programme en C qui permet d'afficher le message "bonjour" 10 fois . Utilisant la boucle for

test.algo

```
Algorithme Bonjour_10_ fois
Variables i: entier
Debut
  pour i de 1 jusqu'à  10 faire
    Ecrire("Bonjour")
  FinPour
Fin
```

exercice_01.cpp

```c
#include <stdio.h>
int main(){
int i ;
 for (i= 0; i < 10 ; i++ ) {
    printf("Bonjour \n");
   }
return 0;
}


```

test.py

```python
for i in range(10) :
    print("Bonjour")

```

Exercice 10 Écrire un programme en C qui permet de calculer la somme S=1+2+3+4+….+ N. où N saisi au clavier par l'utilisateur. Utilisant la boucle for.

test.algo

```
Algorithme Somme_de_1_jusqu'au_N
Variables i,S,N: entiers
Debut
    S ← 0
   Ecrire("Donner un entier")
   Lire (N)
   pour i de 1 jusqu'à  N faire
      S ← S  + i
   FinPour
   Ecrire("La somme est:" , S)
Fin

```

exercice_01.cpp

```cpp
#include <stdio.h>
int main(){
int i, S=0,N;
  printf ("Entrer un entier positif:");
  scanf("%d",&N);
   for (i=1; i<=N ; i++){
     S = S  + i    ;
   }
printf ("La somme 1+2+...+%d = %d" ,N,S);
return 0;
}

```

test.py

```python
S = 0
n=int(input("Donner un entier :"))
for i in range (1,n+1):
    S = S  + i
print("La somme est:",S)

```

Exercice 11 : Écrire un programme en C qui permet d'afficher la table de multiplication de 5. Utilisant la boucle For.

test.algo

```
Algorithme Table_Multiplication_de_5
Variables i:entier
Debut
   pour i de 0 jusqu'à 10 faire
     Ecrire("5*",i,"=",i*5)
   FinPour
Fin

```

exercice_01.cpp

```cpp
#include <stdio.h>
int main(){
int i;

   for (i=0; i<=10 ; i++){
    printf ("5 x %d = %d \n",i,i*5);
   }
return 0;
}

```

test.py

```python
for i in range (1,11):
    print("5 *",i,"=",i*5)
```

Exercice 12 : **Écrire un programme qui permet d'afficher le message "Bonsoir" 10 fois. Utilisant la boucle while.**

test.algo

```
Algorithme
Variables  Tableau[10]:reéls
           i : entier
Debut
    pour i de 1 jusqu'à 10 faire
        lire(Tableau[i])
    FinPour

  pour i de 1 jusqu'à 10 faire
     Ecrire (Tableau[i])
  FinPour
Fin
```

exercice_01.cpp

```
#include <stdio.h>
int main(){
int Tableau[10],i;

for( i=0; i < 10; i++){
printf("Entrer un entier:");
scanf("%d",&Tableau[i]);
}

for( i=0 ; i < 10; i++)
printf("%d ",Tableau[i]);
return 0;
}

```

Exercice 13 : Écrire un programme en C qui permet d'afficher la table de multiplication de 5. Utilisant la boucle For.

test.algo

```
Algorithme   Bonsoir_10_fois
Variables i: entier
Debut
  i ← 1
   TantQue ( i <= 10  ) faire
      Ecrire("Bonsoir")
        i ← i+1
   FinTantQue
Fin

```

test.cpp

```cpp
#include <stdio.h>
int main(){
    int i=1;
    while ( i <= 10  ) {
        printf("Bonsoir\n");
        i++ ;
    }
    return 0;
}
```

test.py

```python
i=1;
while i <= 10:
    print("Bonsoir")
    i=i+1

```

Exercice 14 Écrire un programme en C qui permet de calculer la somme S=1+2+3+4+….+ N. où N saisi au clavier par l'utilisateur. Utilisant la boucle for

test.algo

```
Algorithme Somme_de_1_jusqu'au_N
Variables i,S,N: entiers
Debut
    S ← 0
   Ecrire("Donner un entier")
   Lire (N)
   pour i de 1 jusqu'à  N faire
      S ← S  + i
   FinPour
   Ecrire("La somme est:" , S)
Fin

```

test.cpp

```cpp
#include <stdio.h>
int main(){
    int i, S=0,N;
    printf ("Entrer un entier positif:");
    scanf("%d",&N);
    for (i=1; i<=N ; i++){
        S = S  + i    ;
    }
    printf ("La somme 1+2+...+%d = %d" ,N,S);
    return 0;
}
```

test.py > ...

```python
S = 0
n=int(input("Donner un entier :"))
for i in range (1,n+1):
    S = S  + i
print("La somme est:",S)


```

Exercice 15 : Écrire un programme en C permettant de saisir 10 entiers et qui affiche le maximum de ces entiers.

test.algo

```
Algorithme Max_Notes
Variables i,max,Tab[10]:réels
   Ecrire("Entrer un entier:")
   Lire(Tab[0])        Max ← Tab[0]

   pour i de 1 jusqu'à  10 faire
      Ecrire("Entrer un entier:")
     Lire(Tab[i])
       Si ( Tab[i] > Max ) alors
         Max ← Tab[i]
       FinSi
   FinPour
   Ecrire("Le maximum est:%d",Max)
Fin

```

test.cpp

```cpp
#include <stdio.h>
int main(){
   int Tab[10],Max,i;

   printf("Entrer un entier:");
   scanf("%d",&Tab[0]);   Max = Tab[0] ;

   for( i=1; i < 10; i++){
       printf("Entrer un entier:");
       scanf("%d",&Tab[i]);
       if ( Tab[i] > Max )
       Max = Tab[i] ;
   }

   printf("Le maximum est:%d",Max);
   return 0;
}

```

Exercice 16 : Écrire un programme en C qui permet de saisir 10 entiers dans un tableau. Puis compter combien y a-t-il d'éléments pairs et impairs.

test.algo

```
Algorithme
Variables  i,tab[10],Nb_pair, Nb_impair :entiers
Debut
      Nb_pair ← 0 ;  Nb_impair ← 0

    pour i de 1 à 10 faire
      Ecrire("donner  un entier:")
      Lire(tab[i])
    FinPour

    pour i de 1 à 10 faire (
       Si( tab[i] mod 2 = 0 ) alors
         Nb_pair ← Nb_pair + 1
        else    Nb_impair ← Nb_impair + 1
       FinSi
    FinPour

  Ecrire("Nombre de paires:",Nb_pair)
  Ecrire("Nombre d'impaires:",Nb_impair)
Fin
```

test.cpp

```
#include<stdio.h>
int main(){
    int i,tab[10],Nb_pair=0,Nb_impair=0;

    for(i=0;i<10;i++){
        printf("Donner un entier : ");
        scanf("%d",&tab[i]);
    }

    for(i=0;i<10;i++){

        if( tab[i]%2==0 ) {
            Nb_pair ++;
        } else {
            Nb_impair++ ;
        }
    }

    printf("Nombre de paires:%d\n",Nb_pair);
    printf("Nombre d'impaires:%d",Nb_impair);
    return 0;
}
```

Exercice 17 : **Écrire un programme en C permettant de saisir N entiers et de les stocker dans un tableau nommé Tab, puis les afficher. Où N saisi par l'utilisateur.**

test.cpp

```c
#include <stdio.h>
int main() {
    int N = 0, i = 0;
    int * Tab = NULL;
    printf("Entrer la taille :");
    scanf("%d", &N);

    Tab = malloc(N * sizeof(int));

    for (i = 0 ; i < N ; i++){
        printf("Entrer un entier: ");
        scanf("%d", &Tab[i]);
    }

    for (i = 0 ; i < N ; i++){
        printf("%d ", Tab[i]);
    }

    free(Tab);
    return 0;
}

```

Exercice 18 : **Écrire un programme en C permettant de saisir N notes et qui affiche la moyenne de ces notes. Où N saisi par l'utilisateur.**

test.cpp

```c
#include <stdio.h>
int main(){
    float * notes =NULL,Som=0;
    int i,N ;
    printf("Entrer la taille :");
    scanf ("%d",&N);
    notes = malloc(N * sizeof(int));
    for( i=0; i <N; i++){
        printf("Entrer un entier:");
        scanf("%f",&notes[i]);
        Som = Som + notes[i];
    }
    printf("La moyenne est:%f",Som/10);
    return 0;
}

```

Exercice 19 : Écrire un programme en C permettant de saisir N entiers et qui affiche le maximum de ces entiers. Où N saisi par l'utilisateur.

test.cpp

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

    int * Tab,Max,i,N;
    printf("Entrer la taille:");
    scanf("%d",&N);

    Tab = malloc(N * sizeof(int));
    printf("Entrer un entier:");
    scanf("%d",&Tab[0]);

    Max = Tab[0] ;

    for( i=1; i <N; i++){

        printf("Entrer un entier:");
        scanf("%d",&Tab[i]);

        if ( Tab[i] > Max ){
            Max = Tab[i] ;
        }

    }

    printf("Le maximum est:%d",Max);
    return 0;
}

```

Exercice 20 : Écrire un programme en C permettant de saisir N entiers dans un tableau, et de calculer le nombre d'occurrences d'un élément N dans ce tableau. Où N saisi par l'utilisateur. Où N saisi par l'utilisateur.

test.cpp

```cpp
#include <stdio.h>
int main(){
    int * Tab,N,nb_occ=0,i,nb_rech;
    printf("Entrer la taille :");
    scanf("%d",&N);
    Tab = malloc(N * sizeof(int));

    for( i=0; i < N; i++){
        printf("Entrer un entier:");
        scanf("%d",&Tab[i]);
    }

    printf("Entrer Le nombre recherché:");
    scanf("%d",&nb_rech);

    for( i=0; i <N; i++){
        if ( Tab[i] == nb_rech )
        nb_occ ++  ;
    }

    printf("Nombre d'occurences de %d est %d",nb_rech,nb_occ);
    return 0;
}

```

Exercice 21 : Écrire un programme en C permettant de saisir N entiers dans un tableau et de trier ce tableau par ordre croissante. puis affiche ce tableau après le tri. Où N saisi par l'utilisateur.

test.cpp

```c
#include <stdio.h>
int main(){
    int * Tab,N,auxilaire,i,j;
    printf("Entrer la taille :");
    scanf("%d", &N);
    Tab = malloc(N * sizeof(int));

    for( i=0; i <N; i++){
        printf("Entrer un entier:");
        scanf("%d",&Tab[i]);
    }

    for( i=0; i <N-1; i++){
        for( j=i+1; j <N; j++){
            if ( Tab[j] < Tab[i]){
                auxilaire = Tab[i] ;
                Tab[i] = Tab[j] ;
                Tab[j] = auxilaire ;
            }
        }
    }

    for( i=0; i <N; i++){
        printf("%d\t",Tab[i]);
    }

    return 0;
}

```

Exercice 20 : Écrire une fonction qui permet de calculer le prix TTC , cette fonction va recevoir un paramètre de type Réel dont le nom est "prixHT" et un second paramètre de type Réel dont le nom est "tva".

test.cpp

```c
#include <stdio.h>
float calculPrixTTC(float prixHT, float tva) {
    float prixTTC;
    prixTTC = prixHT * (1 + tva / 100);
    return prixTTC;
}

int main()
{
    float tauxTva,prixHT,prixTTC;
    printf("Entrer le prix HT:");
    scanf("%f",&prixHT);
    printf("Entrer la TVA:");
    scanf("%f",&tauxTva);
    prixTTC= calculPrixTTC(prixHT, tauxTva);
    printf("Prix TTC : %f", prixTTC);
    return 0;
}

```

Exercice 21 : Écrire une fonction qui permet d'afficher si un nombre entier passé en paramètre est pair ou impair

test.cpp

```c
#include <stdio.h>
void parite (int nb) {
    if(nb % 2 == 0){
        printf("%d est impair\n", nb);
    }else{
        printf("%d est pair\n", nb);
    }
}

int main()
{
    int n;
    printf("Entrer un Entier :");
    scanf("%d",&n);
    parite (n);
    return 0;
}
```

Exercice 22 : Écrire une fonction qui remplace les voyelles (minuscules et majuscules) par des espaces dans une chaîne passée en paramètre.

test.cpp

```c
#include <stdio.h>
int estUneVoyelle(char c) {
    char tVoyelle[] = "aAeEiIoOUuYy";
    for(int i=0 ; i<12 ; i++) {
        if(c == tVoyelle[i])
            return 1;
    }
    return 0;
}
int remplaceVoyellesParEspaces(char text[]) {
    int i=0, nbRemplacement = 0;
    while(text[i] != '\0') {
        if(estUneVoyelle(text[i])) {
            text[i] = ' ';
            nbRemplacement++;
        }
        i++;
    }
    return nbRemplacement;
}
int main() {
    char leTexte[50];
    printf("Entrez un  mot : ");
    scanf("%s", leTexte);
    int n = remplaceVoyellesParEspaces(leTexte);
    printf("%d caractères ont été remplacés   %s", n, leTexte);
    return 0;
}

```

Exercice 23 : Écrire une fonction qui permet de retourner le nombre de caractères d'une chaîne de caractères passée en paramètre.

test.cpp

```cpp
#include <stdio.h>

int longueurChaine(char texte[]) {

    int longueur = 0;
    while (texte[longueur] != '\0') {
        longueur++;
    }
    return longueur;
}

int main() {
    char tNom[30];
    printf("Saisissez votre nom : ");
    scanf("%s", tNom);
    printf("\nVotre nom compte %d caracteres", longueurChaine(tNom));
    return 0;
}


```

Exercice 24 : Écrire une fonction qui permet d'inverser une chaîne de caractères passée en paramètre

test.cpp

```cpp
#include <stdio.h>
void inverseChaine(char texte[]) {
  int nbCar, i;
  char temp;
  nbCar = longueurChaine(texte);
  for(i = 0; i < nbCar / 2; i++) {
    echange(&texte[i], &texte[nbCar - 1 - i]);
  }
}

void echange(char* c1, char* c2) {
  char temp = *c1;
  *c1 = *c2;
  *c2 = temp;
}

int longueurChaine(char texte[]) {
  int longueur = 0;
  while(texte[longueur] != '\0') longueur++;
  return longueur;
}

int main() {
  int nbCar;
  char tMot[30];
  printf("Saisissez un mot : ");
  scanf("%s", tMot);
  inverseChaine(tMot);
  printf("\n%s", tMot);
}
```

Exercice 25 : Écrire une fonction qui cherche combien de fois un caractère est présent dans une chaîne de caractères. Le caractère à chercher et la chaîne seront passés en paramètres

test.cpp

```c
#include <stdio.h>
int nombreOccurence(char texte[], char car) {
    int nbCar = 0;
    int i;

    i = 0;
    while(texte[i] != '\0') {

        if(texte[i] == car) nbCar++;

        i++;
    }

    return nbCar;
}

int main()
{
    char leTexte[40],c;
    printf("Entrer la lettre :");
    scanf("%c",&c);
    printf("Entrer une chaîne :");
    scanf("%s",leTexte);


    printf("\n\nLe caractère %c est présent %d fois",c, nombreOccurence(leTexte, c));

    return 0;
}
```

Exercice 26 : Écrire une fonction récursive qui permet d'afficher les éléments d'une matrice passée en paramètre.

test.cpp

```c
#include <stdio.h>
void showItemOfMatrix(int tab[3][4],int a,int b){

  if( a < 3 )
    if (b < 4){
        printf("%d",tab[a][b]);
        showItemOfMatrix(tab,a,b+1);
    }else{
        printf("\n");
        showItemOfMatrix(tab,a+1,0);
    }
 }

int main() {
    int T[3][4]={0,1,2,3,4,5,6,7,0,1,2,3};
    showItemOfMatrix(T,0,0);
    return 0;
}
```

Exercice27 : Écrire une fonction récursive qui permet de calculer le PGCD de deux entiers passés en paramètres, (utiliser l'algorithme d'Euclide)

test.cpp

```cpp
#include <stdio.h>

int  algoEuclideRecursif(int a,int b) {
    if( b == 0) {
        return  a ;
    }else{
        return  algoEuclideRecursif(b,a%b);
    }
}
int main() {

    int a,b;
    printf("Entrer un entier a :");
    scanf("%d",&a);
    printf("Entrer un entier b :");
    scanf("%d",&b);
    printf("PGCD(%d,%d)=%d",a,b, algoEuclideRecursif(a,b));
    return 0;
}

```

Exercice 28 : Écrire une fonction récursive qui permet de calculer l'image d'un entier (passé en paramètre) par une suite de Fibonacci. Suite de Fibonacci est une suite d'entiers dans laquelle chaque terme est la somme des deux termes qui le précèdent. F(0)=F(1)=1

test.cpp

```cpp
#include <stdio.h>

int suiteDeFibonacci(int n){
    if(n ==0 || n == 1)
    return 1;
    else
    return suiteDeFibonacci(n-2)+suiteDeFibonacci(n-1);
}

int main() {

    int n;
    printf("Entrer un entier : ");
    scanf ("%d",&n);
    printf("%d",suiteDeFibonacci(n));
    return 0;

}

```

Problème 01 : (Cahier de Charge)

Dans le département d'informatique de la faculté des sciences, il existe un laboratoire, appelé « Traitement de l'information et Aide à la Décision », qui regroupe un ensemble de professeurs, chercheurs et doctorants.

Ce laboratoire produit chaque année un ensemble de productions scientifiques. L'objectif de ce sujet est la gestion automatique de cette production en utilisant un tableau de structures, où chaque production est une structure.

Une production est caractérisée par les informations suivantes : un code de référence (ref), le titre de la production (titre), l'auteur principal (aut1), le directeur de la thèse (aut2), l'encadrant (aut3), le type de production (type : thèse, publication ou communication), et la date de production (date).

On considère les structures suivantes :

test.algo

```
typedef struct  {
   int  jours ;
   int  mois ;
   int  annee ;
}

Date;

typedef struct  {
   int      ref  ;
   char     titre [100]  ;
   char     aut1[40] ;
   char     aut2[40];
   char     aut3[40] ;
   char     type[40]   ;
   Date     date  ;
}

Production;

#define max 100
```

Les fonctions à rédiger sont les suivantes:

1) void lecture (Production P[max], int N) qui permet de saisir les informations de N productions.

2) void Affichage (Production P[max] , int N) qui permet d'afficher les informations de N productions.

3) void NombreProductions (Production P[max] , int N) qui permet de calculer et afficher le nombre de production pour chaque type (thèse, publication et communication) sachant que P un tableau contenant les productions et N représente le nombre total de productions.

4) Production * RechType (Production P[max] , int N , char type[30] , int * k)

qui permet de rechercher et retourne un tableau des productions (un pointeur sur Production) contenant les Productions d'un type donné (type). P représente un tableau de productions et N représente le nombre total de productions, et le troisième argument k sert à mémoriser le nombre de productions enregistrés dans le tableau retourné.

5) void SuppProd (Production P[max] , int * N, char auteur[30]) qui permet de

supprimer les productions d'un auteur (qui peut être un auteur principal, un directeur de thèse ou bien un encadrant). Et après la suppression, afficher les productions supprimées.

6) void AffichOrdre (Production P[max] , int N) qui permet d'afficher

les productions par ordre croissant selon la date de production. P représente un tableau de productions et N représente le nombre total de productions.

Fonction 01 :

test.cpp

```cpp
void Lecture(Production P[max],int N){

    int i;
    for ( i=0; i<N ; i++){

        printf("Entrer la reference :"); scanf("%d",&P[i].ref);
        printf("Entrer Le titre : "); scanf("%s",P[i].titre);
        printf("Entrer Le nom de l'auteur  : "); scanf("%s",P[i].aut1);
        printf("Entrer Le nom du directeur :");  scanf("%s",P[i].aut2);
        printf("Entrer le nom de l'encadrant  :"); scanf("%s",P[i].aut3);
        printf("Entrer le type de la production  :"); scanf("%s",P[i].type);
        printf("Entrer la date de la prodction: \n");
        printf("  Entrer le jour :"); scanf("%d",&P[i].date.jour);
        printf("  Entrer le mois :"); scanf("%d",&P[i].date.mois);
        printf("  Entrer l'annee :"); scanf("%d",&P[i].date.annee);
        printf("*********************************\n");
    }
}

```

Fonction 02 :

test.cpp

```cpp
void Affichage(Production P[max],int N){
    int i ;

    for ( i=0; i<N ; i++)
    {
        printf(" La reference : %d\n",P[i].ref);
        printf(" Le titre: %s\n",P[i].titre);
        printf(" Le nom de l'auteur: %s\n",P[i].aut1);
        printf(" Le nom du directeur: %s\n",P[i].aut2);
        printf(" Le nom de l'encadrant: %s\n",P[i].aut3);
        printf(" Le type de la production: %s\n",P[i].type);
        printf(" La date de la prodction: %d/%d/%d\n",P[i].date.jour,
        P[i].date.mois,P[i].date.annee);
        printf("*****************************\n");
    }
}

```

Fonction 03 :

test.cpp

```cpp
void  NombreProductions(Production P[max],int N)
{
    int Nb_pro_these=0,Nb_pro_pub=0,Nb_pro_commu=0 ,i;

    for ( i=0; i<N ; i++)
    {
        if(strcmp(P[i].type,"These")==0)
            Nb_pro_these++;
        if(strcmp(P[i].type,"Publication")==0)
            Nb_pro_pub++;
        if(strcmp(P[i].type,"Communication")==0)
            Nb_pro_commu++;
    }

    printf("Le nombre de production Thèse : %d\n",Nb_pro_these);
    printf("Le nombre de production publication : %d\n",Nb_pro_pub);
    printf("Le nombre de production communication : %d\n",Nb_pro_commu);
}

```

Fonction 04 :

test.cpp

```cpp
Production * RechType(Production P[max],int N,char type[30],int *k)
{
    Production tab[max];
    int i;
    *k = 0 ;
    for ( i=0; i<N ; i++)
    {
        if(strcmp(P[i].type,type)==0){
            tab[*k] = P[i];
            (*k)++ ;
        }
    }
    return tab;
}
```

Fonction 06 :

test.cpp

```cpp
void SuppProd (Production P[max],int *N,char auteur[30])
{
    Production supp[max];
    int i,j,k=0;
    for ( i=0; i<*N ; i++)
    {
        if(strcmp(P[i].aut1,auteur)==0 ||
        strcmp(P[i].aut2,auteur)==0 ||
        strcmp(P[i].aut3,auteur)==0){
            supp[i]=P[i];
            for ( j=i ; j<*N ; j++)  {
                P[j] = P[j+1];
                (*N)--; i--;  k++;
            }
        }
    }
    printf("la liste de productions supprimées :");
    Affichage(supp,k);
}
```

Fonction 07 :

test.cpp

```cpp
void AffichOrdre(Production P[max],int N)
{
    Production aide;
    int i,j;

   for( i=0; i <N-1; i++)
   {
       for( j=i+1; j <N; j++){
           if(P[i].date.annee > P[j].date.annee) {

               aide = P[i]; P[i] = P[j] ;  P[j]= aide ;
           }
           if( P[i].date.annee == P[j].date.annee
               && P[i].date.mois > P[j].date.mois) {

               aide = P[i]; P[i] = P[j] ;  P[j]= aide ;
           }

           if( P[i].date.annee == P[j].date.annee &&
               P[i].date.mois  == P[j].date.mois  &&
               P[i].date.jour > P[j].date.jour) {

               aide = P[i]; P[i] = P[j] ;  P[j]= aide ;
           }
       }
    }
    Affichage(P,N);
}
```

Problème 02 (Cahier de Charge)

L'objectif est de créer un programme permettant la gestion d'une bibliothèque, cette bibliothèque contient un grand nombre de livres de différentes spécialités. Chaque livre comporte un ensemble d'informations (code, titre, spécialité, auteur et édition).

On considère les structures suivantes :

test.algo

```
typedef struct  {
   int  jour ;
   int  mois ;
   int  annee ;
}
Edition ;

typedef struct   {
   int code  ;
   char titre [30]  ;
   char specialite[30]  ;
   char auteur[30]  ;
   Edition edit  ;
}
Livre  ;

#define max 100
```

Les fonctions à rédiger sont les suivantes :

1) void lecture (Livre L[max], int N) qui permet de saisir les informations de N livres.

2) void afficher (Livre L[max], int N) cette fonction permet d'afficher les informations de N livres.

3) int comparer (Edition x , Edition y) qui permet de comparer deux dates d'éditions, et retourne (-1 si x avant y, 0 si x et y sont identiques , 1 si x après y).

4) void ordre_edition (Livre L[max] , int N) permet de classer les livres du tableau L par ordre,

selon la date d'édition.

5) Livre * livres_auteur (Livre L[max] , int N , char nom[30] , int * k)

cette fonction permet de retourner un tableau contenant les livres d'un auteur passé en paramètre (nom).

L représente un tableau contenant N Livres et k sert à mémoriser la taille du tableau retourné.

6) void supprimer (Livre L[max] , int * N , char nom[30])

cette fonction permet de supprimer les livres d'un auteur passé en paramètre.

N passé par adresse pour modifier la taille du tableau après la suppression.

Fonction 01 :

test.cpp

```cpp
void   lecture (Livre L[max],int N ){
    int i;
    for ( i=0; i<N ; i++)
    {
        printf("Entrer le code:");   scanf("%d",&L[i].code);
        printf("Entrer le titre: "); scanf("%s",L[i].titre);
        printf(" specialite: ");  scanf("%s",L[i].specialite);
        printf(" L'auteur :");    scanf("%s",L[i].auteur);
        printf("Date d'edition \n");
        printf(" Jour:");  scanf("%d",&L[i].edit.jour);
        printf(" Mois:");  scanf("%d",&L[i].edit.mois);
        printf(" annee:"); scanf("%d",&L[i].edit.annee);
        printf("**************************\n");
    }
}
```

Fonction 02 :

test.cpp

```cpp
void  afficher (Livre L[max],int N){
    int i;
    for ( i=0; i<N ; i++)
    {
        printf("le code: %d\n",L[i].code);
        printf("Le titre : %s\n",L[i].titre);
        printf(" specialite :%s\n",L[i].specialite);
        printf(" l'auteur : %s\n",L[i].auteur);
        printf("Date d'edition: %d/%d/%d \n",
        L[i].edit.jour,L[i].edit.mois,L[i].edit.annee);

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

```

Fonction 03

test.cpp

```cpp
int  comparer(Edition x ,Edition y) {

   if(x.annee < y.annee ) return -1;
   if(x.annee > y.annee ) return  1 ;
   if(x.annee == y.annee && x.mois < y.mois ) return -1 ;
   if(x.annee == y.annee && x.mois > y.mois ) return  1 ;
   if(x.annee == y.annee && x.mois == y.mois && x.jour < y.jour) return -1 ;
   if(x.annee == y.annee && x.mois == y.mois && x.jour > y.jour) return  1 ;
   if(x.annee == y.annee && x.mois == y.mois && x.jour == y.jour) return 0 ;
}


```

Fonction 04 :

test.cpp

```cpp
void ordre_edition(Livre L[max],int N){

   Livre aide;   int i,j;

   for( i=0; i <N-1; i++)
   {
       for( j=i+1; j <N; j++)
       {
           if(L[i].edit.annee > L[j].edit.annee) {
               aide = L[i]; L[i] = L[j] ;  L[j]= aide ;
           }
           if( L[i].edit.annee == L[j].edit.annee &&
               L[i].edit.mois > L[j].edit.mois) {

               aide = L[i]; L[i] = L[j] ;  L[j]= aide ;
           }

           if( L[i].edit.annee == L[j].edit.annee &&
               L[i].edit.mois  == L[j].edit.mois  &&
               L[i].edit.jour > L[j].edit.jour) {
               aide = L[i]; L[i] = L[j] ;  L[j]= aide ;
           }
       }
   }
}

```

Fonction 05 :

test.cpp

```cpp
Livre * livres_auteur(Livre L[max],int N ,char nom[30],int *k)
{
    Livre tab[max];
    int i;
    *k = 0 ;
    for ( i=0; i<N ; i++)
    {
        if(strcmp(L[i].auteur,nom)==0){
            tab[*k] = L[i];
            (*k)++ ;
        }
    }
    return tab;
}

```

Fonction 06

test.cpp

```cpp
void   supprimer (Livre L[max],int *N,char nom[30])  {

    int i,j;
    for ( i=0; i<*N ; i++){
        if(strcmp(L[i].auteur,nom)==0){

           for (j=i;j<*N ;j++){
               L[j] = L[j+1];
               (*N)--;
               i--;
           }
        }
    }
 }

```