TPC++

TP1:

Enoncé 1

Créer un programme en C qui effectue les opérations arithmétiques de base (multiplication, addition, soustraction, division, modulo) sur deux nombres entiers.

```
#include <stdio.h>

int main() {
   // Déclaration des variables
   int nombre1, nombre2;
   int somme, difference, produit, quotient, reste;

   // Demande à l'utilisateur d'entrer les deux nombres
   printf("Entrez le premier nombre : ");
   scanf("%d", &nombre1);
   printf("Entrez le deuxième nombre : ");
   scanf("%d", &nombre2);

   // Effectue les opérations
   somme = nombre1 + nombre2;
   difference = nombre1 - nombre2;
   produit = nombre1 * nombre2;
   quotient = nombre1 / nombre2;
   reste = nombre1 % nombre2;

   // Affiche les résultats
   printf("Somme : %d + %d = %d\n", nombre1, nombre2, somme);
   printf("Différence : %d - %d = %d\n", nombre1, nombre2, difference);
   printf("Produit : %d * %d = %d\n", nombre1, nombre2, produit);
   printf("Quotient : %d / %d = %d\n", nombre1, nombre2, quotient);
   printf("Reste : %d %% %d = %d\n", nombre1, nombre2, reste);

   return 0; // Indique que le programme s'est exécuté avec succès
}
```

Enoncé 2

Écrire un programme en C qui résout une équation du premier degré (de la forme ax + b = 0).

```
#include <stdio.h>

int main() {
   // Déclaration des variables
   float a, b, x;

   // Demande à l'utilisateur d'entrer les coefficients de l'équation
   printf("Entrez le coefficient a : ");
   scanf("%f", &a);
   printf("Entrez le coefficient b : ");
   scanf("%f", &b);

   // Vérifie si l'équation est linéaire (a différent de 0)
   if (a != 0) {
      // Résout l'équation du premier degré
      x = -b / a;
      printf("La solution de l'équation %.2fx + %.2f = 0 est x = %.2f\n", a, b, x);
```

```
    } else {
 // Si a est égal à 0, l'équation n'est pas du premier degré
        printf("L'équation n'est pas du premier degré (a ne peut pas être égal à 0).\n"); }

    return 0; // Indique que le programme s'est exécuté avec succès
}
```

TP2:

Enoncé

a-Supposons que vous ayez deux nombres complexes, l'un représenté par sa partie réelle a et sa partie imaginaire b, et l'autre par sa partie réelle c et sa partie imaginaire d. Si vous entrez a, b, c et d en tant que parties réelles et imaginaires respectives des deux nombres complexes, ce programme en C calcule et affiche la somme, la différence, le produit, le quotient, le module et l'argument (en radians) des deux nombres complexes.

Formulaire : Somme : (a + c) + (b + d)i

Formulaire : Différence : (a - c) + (b - d)i

Formulaire : Produit : (a * c - b * d) + (a * d + b * c)i

Formulaire : Quotient : (a * c + b * d) / (c * c + d * d) + (b * c - a * d) / (c * c + d * d)i(Assurez-vous de vérifier si le dénominateur est différent de zéro)

Formulaire : Module du nombre complexe : sqrt(a^2 + b^2)

Formulaire : Argument du nombre complexe en radian: atan2(b, a)

Répondez à ces questions en utilisant les valeurs que vous entrez pour a, b, c et d lors de l'exécution du programme.

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

int main() {
   float reel1, imaginaire1, reel2, imaginaire2;
   float sommeReelle, sommeImaginaire, differenceReelle, differenceImaginaire;
   float produitReel, produitImaginaire, quotientReel, quotientImaginaire;
   float module1, module2, argument1, argument2;

   // Saisie des nombres complexes
   printf("Entrez la partie réelle du premier nombre complexe : ");
   scanf("%f", &reel1);
   printf("Entrez la partie imaginaire du premier nombre complexe : ");
   scanf("%f", &imaginaire1);
   printf("Entrez la partie réelle du deuxième nombre complexe : ");
   scanf("%f", &reel2);
   printf("Entrez la partie imaginaire du deuxième nombre complexe : ");
   scanf("%f", &imaginaire2);

   // Calcul des opérations sur les nombres complexes
   sommeReelle = reel1 + reel2;
   sommeImaginaire = imaginaire1 + imaginaire2;
   differenceReelle = reel1 - reel2;
   differenceImaginaire = imaginaire1 - imaginaire2;
   produitReel = reel1 * reel2 - imaginaire1 * imaginaire2;
   produitImaginaire = reel1 * imaginaire2 + imaginaire1 * reel2;

   // Vérification du dénominateur pour la division
   if (reel2 * reel2 + imaginaire2 * imaginaire2 != 0) {
      quotientReel = (reel1 * reel2 + imaginaire1 * imaginaire2) / (reel2 * reel2 + imaginaire2 *
imaginaire2);
      quotientImaginaire = (imaginaire1 * reel2 - reel1 * imaginaire2) / (reel2 * reel2 + imaginaire2 *
imaginaire2);
```

```
    } else {
        printf("Division par zéro impossible.\n");
    }

    // Calcul des modules
    module1 = sqrt(reel1 * reel1 + imaginaire1 * imaginaire1);
    module2 = sqrt(reel2 * reel2 + imaginaire2 * imaginaire2);

    // Calcul des arguments en radians
    argument1 = atan2(imaginaire1, reel1);
    argument2 = atan2(imaginaire2, reel2);

    // Affichage des résultats
    printf("Somme : %f + %fi\n", sommeReelle, sommeImaginaire);
    printf("Différence : %f + %fi\n", differenceReelle, differenceImaginaire);
    printf("Produit : %f + %fi\n", produitReel, produitImaginaire);
    printf("Quotient : %f + %fi\n", quotientReel, quotientImaginaire);
    printf("Module du premier nombre complexe : %f\n", module1);
    printf("Module du deuxième nombre complexe : %f\n", module2);
    printf("Argument du premier nombre complexe : %f radians\n", argument1);
    printf("Argument du deuxième nombre complexe : %f radians\n", argument2);

    return 0;
}
```

b- À l'aide de la programmation orientée objet en C++, créez un programme permettant d'effectuer les mêmes opérations

```
#include <iostream>
#include <cmath>
using namespace std;

class Complexe {
private:
    float reel1, imaginaire1, reel2, imaginaire2;

public:
    void saisie() {
        cout << "Entrez la partie réelle du premier nombre complexe : ";
        cin >> reel1;
        cout << "Entrez la partie imaginaire du premier nombre complexe : ";
        cin >> imaginaire1;
        cout << "Entrez la partie réelle du deuxième nombre complexe : ";
        cin >> reel2;
        cout << "Entrez la partie imaginaire du deuxième nombre complexe : ";
        cin >> imaginaire2;
    }

    void addition() {
        float sommeReelle = reel1 + reel2;
        float sommeImaginaire = imaginaire1 + imaginaire2;
        cout << "Somme : " << sommeReelle << " + " << sommeImaginaire << "i" << endl;
    }

    void soustraction() {
        float differenceReelle = reel1 - reel2;
        float differenceImaginaire = imaginaire1 - imaginaire2;
        cout << "Différence : " << differenceReelle << " + " << differenceImaginaire << "i" << endl;
    }

    void multiplication() {
```

```
        float produitReel = reel1 * reel2 - imaginaire1 * imaginaire2;
        float produitImaginaire = reel1 * imaginaire2 + imaginaire1 * reel2;
        cout << "Produit : " << produitReel << " + " << produitImaginaire << "i" << endl;
    }

    void division() {
        float denominateur = reel2 * reel2 + imaginaire2 * imaginaire2;
         if (reel2 * reel2 + imaginaire2 * imaginaire2 != 0) {
        float quotientReel = (reel1 * reel2 + imaginaire1 * imaginaire2) / denominateur;
        float quotientImaginaire = (imaginaire1 * reel2 - reel1 * imaginaire2) / denominateur;
        cout << "Quotient : " << quotientReel << " + " << quotientImaginaire << "i" << endl;
    }
    else
    cout<<"Division par zéro impossible"<< endl;
    }

    void calculerModule() {
        float module = sqrt(reel1 * reel1 + imaginaire1 * imaginaire1);
        cout << "Module du premier nombre complexe : " << module << endl;
        module = sqrt(reel2 * reel2 + imaginaire2 * imaginaire2);
        cout << "Module du deuxième nombre complexe : " << module << endl;
    }

    void calculerArgument() {
        float argument1 = atan2(imaginaire1, reel1);
        float argument2 = atan2(imaginaire2, reel2);
        cout << "Argument du premier nombre complexe : " << argument1 << " radians" << endl;
        cout << "Argument du deuxième nombre complexe : " << argument2 << " radians" << endl;
    }
};

int main() {
    Complexe nombre;
    nombre.saisie();
    nombre.addition();
    nombre.soustraction();
    nombre.multiplication();
    nombre.division();
    nombre.calculerModule();
    nombre.calculerArgument();
    return 0;
}
```

TP3:

Enoncé

Écrivez un programme en C++ utilisant la programmation orientée objet qui permet à l'utilisateur de saisir une matrice carrée, puis affiche cette matrice ainsi que sa partie triangulaire inférieure, sa partie triangulaire supérieure, sa diagonale principale et sa diagonale secondaire.

```
#include <iostream>
using namespace std;
class Matrice {
private:
   int matrice[10][10];
   int taille;

public:
           void tailleMatrice()
           {
   cout << "Entrez la taille de la matrice : ";
   cin >> taille;
}
   void lireMatrice() {
      cout << "Saisie de la matrice : " << std::endl;
      for (int i = 0; i < taille; ++i) {
         for (int j = 0; j < taille; ++j) {
            cout << "Matrice[" << i+1 << "][" << j+1 << "] : ";
            std::cin >> matrice[i][j];
         }
      }
   }
   void afficherMatrice() {
      std::cout << "Saisie de la matrice : " << std::endl;
      for (int i = 0; i < taille; ++i) {
         for (int j = 0; j < taille; ++j) {
            std::cout<<matrice[i][j]<<" ";
         }
        std::cout << std::endl;
      }
   }

   void afficherPartieTriangulaire() {
      std::cout << "Partie triangulaire inférieure : " << std::endl;
      for (int i = 0; i < taille; ++i) {
         for (int j = 0; j < taille; ++j) {
            if (j<=i)
            std::cout<<matrice[i][j]<<" ";
            else
            std::cout<<0<<" ";
         }
         std::cout << std::endl;
      }

      std::cout << "Partie triangulaire supérieure : " << std::endl;
      for (int i = 0; i < taille; ++i) {
         for (int j = 0; j < taille; ++j) {
             if (j>=i)
            std::cout<<matrice[i][j]<<" ";
            else
```

```
            std::cout<<0<<" ";
        }
        std::cout << std::endl;
    }
  }

  void afficherDiagonales() {
    std::cout << "Diagonale principale : "<<endl;
     for (int i = 0; i < taille; ++i) {
        for (int j = 0; j < taille; ++j) {
            if (j==i)
            std::cout<<matrice[i][j]<<" ";
            else
            std::cout<<0<<" ";
        }
        std::cout << std::endl;
    }
    std::cout << "Diagonale secondaire : "<<endl;
     for (int i = 0; i < taille; ++i) {
        for (int j = 0; j < taille; ++j) {
            if (j == taille - 1-i)
            std::cout<<matrice[i][j]<<" ";
            else
            std::cout<<0<<" ";
        }
        std::cout << std::endl;
    }
  }
};

int main() {
  Matrice mat;
  mat.tailleMatrice();
  mat.lireMatrice();
  mat.afficherMatrice();
        mat.afficherPartieTriangulaire();
  mat.afficherDiagonales();

  return 0;
}
```

TP4:

Enoncé

Ecrire un programme en C++ (classe d'objet), qui résoudre l'EDO suivant avec la méthode de RK4 :

$$\begin{cases} y'(x) = xsin(y(x)), & 0 \le x \le 1 \\ y(0) = 2 \end{cases}$$

Rappel : La méthode RK4 est donnée par l'équation :

$$y_{n+1} = y_n + \frac{h}{6}[k_1 + 2k_2 + 2k_3 + k_4]$$

Avec :

$k_1 = f(x_n, y_n)$; k_2=f(x_n+h/2, $y_n + hk_1/2$); k_3=f(x_n+h/2, $y_n + hk_2/2$); k_4=f(x_n+h, $y_n + hk_3$) et h= $(x_f - x_0)/N$;

Solution :

```
#include <iostream>
#include<math.h>
/* run this program using the console pauser or add your own getch, system("pause") or input loop */
using namespace std;
class rk4
{
   private:
      float  x,y,x0,y0,h,xfinal,k1,k2,k3,k4;
      int N;
   public:
      void lecture()
      {
         cout<<"entrez les valeurs x0, xfinal, y0 et N"<<endl;
         cin>>x0>>xfinal>>y0>>N;//donner les valeurs initiale et final de x, la valeur initiale de y et la dimension de l'espace N
         h=(xfinal-x0)/N;//calculer le pas de deplacement
      }
      float  fcalc (float x, float y)//fonction pour calculer f(x,y)
      {
            y=x*sin(y);
         return y;
      }

      void calcul()
      {
            y= y0 ; x=x0 ;
         while (x<xfinal)
         {
           k1= fcalc(x,y);k2=fcalc(x+h/2,y+h*k1/2);k3=fcalc(x+h/2,y+h*k2/2);k4=fcalc(x+h,y+h*k3);// calculer k1, k2, k3 et k4
           cout<<"k1="<<k1<<", k2="<<k2<<", k3="<<k3<<"  et k4="<<k4<<endl; //afficher k1, k2, k3 et k4
            y=y+(k1+2*k2+2*k3+k4)/6;//calculer la solution
            x=x+h;// calculer noeud suivant
               cout<<"x="<<x<<"  et y="<<y<<endl; //afficher x et y
             }
```

```
        cout<<"fin de résolution";
    }
};

int main(int argc, char** argv) {
rk4 R;
    R.lecture();
    R.calcul();
return 0;
}
```

TP5:

Enoncé

Considérons le problème de l'équation de poisson en 2D avec conditions aux limites homogène de Dirichlet

$$\begin{cases} -u_{xx}(x,y) - u_{yy}(x,y) = 2x + y, & (x,y) \in \Omega \times \Omega \\ u(x,y) = 0, & (x,y) \in \partial\Omega \times \partial\Omega \end{cases}$$

Ecrire un programme en C++, qui résoudre cette EDP par la méthode de gauss avec Ω=[0,1].

Solution :

Une façon simple d'approcher ce problème est de diviser l'intervalle [0, 1] en (N+1) mailles $[x_j , x_{j+1}]$ pour j ∈ {0, . . . , N}, avec x_0 = -1, x_{N+1}= 1, $x_{j+1} - x_j$ = h, et de résoudre le système discret

$$\begin{cases} \dfrac{-u_{i+1j} - u_{i-1j} + 2u_{ij} - u_{ij+1} - u_{ij-1}}{h^2} = f_{ij}; \quad i,j \in \{1,\dots,N\} \\ u_{0j=}u_{N+1j} = u_{i0=}u_{iN+1} = 0, \quad i,j \in \{0,\dots,N+1\} \end{cases}$$

Pour N=3, ce système peut se mettre sous forme matricielle :

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} u_{11} \\ u_{12} \\ u_{13} \\ u_{21} \\ u_{22} \\ u_{23} \\ u_{31} \\ u_{32} \\ u_{33} \end{bmatrix} = h^2 \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix}$$

Avec $f_{ij} = 2x_i + y_j = 2x_i + x_j$.

Notre but est d'écrire un programme en C++ qui résout ce système linéaire.

```
#include <iostream>
/* run this program using the console pauser or add your own getch, system("pause") or input loop */
using namespace std;
class   eq_poisson_2d
   {private:
              int M,N,i,j,k;
                        float a,b,dy,dx;
                        float A[100][100],C[100][100];
                        float B[100], v[100], x[100],y[100];
            public:
             void  taille (  )
              {cout<<"donner la dimension de l'espace:  ";
                        cin>>N;
              }
              void eq_poisson1 ()
                        {
                           /*donner les condition initiales de x et y */
                           cout<<"donner les valeurs initiales de x et y";
                           cin>>a>>b;
```

```
                    //verifier que a<b;
                    while(b<a)
                    {
                        cout<<"donner nouvelle valeur de b";
                         cin>>b;
                             }
                    cout<<"les valeurs sont:  a="<<a<<"  b="<<b<<"\n";
                    //calculer le pas dx
                    dx=(b-a)/(N+1);
                    cout<<"le pas dx= "<<dx<<"\n";
                    /*calculer les valeurs x0,x1,x2...de vecteur x */
                    x[0]=a;
cout<<"x[0]="<<x[0]<<" ";
                    for(j=1;j<=N+1;j++)
                    {
                              x[j]=x[j-1]+dx;
                              cout<<"x["<<j<<"]="<<x[j]<<" ";
                    }
                    cout<<"\n";
                    cout<<"donner les valeur de matrice A et vecteur B\n";
                    M=N*N;
                    for(i=0;i<M;i++)
                   {
                              for(j=0;j<M;j++)
                              {
                                        cout<<"donner le "<<i+1<<" ligne de A avec "<<j+1<<" colonne\n";
                                         cin>>A[i][j];
                     }
                   }
                   k=0;
                   for(i=0;i<N;i++)
                   {
                              for(j=0;j<N;j++)
                              {
                                //on initialise le vecteur:
                                        C[i][j]=dx*dx*(2*x[i+1]+x[j+1]);
                                        B[k]=C[i][j];
                                        k=k+1;
                              }
                    }
                   }

                 void  afficher()
                 {
                   /*afficher la matrice A et le vecteur B*/
                              cout<<"la matrice A et vecteur B sont:\n";
                              for(i=0;i<N*N;i++)
                    {
                              for(j=0;j<N*N;j++)
                              {
                                        cout<< A[i][j]<<"   ";
                                }
                              cout<<"          ";
                                cout<<B[i]<<" "<<endl;
                 }
        }
                 void gauss()
                 {
                 float som,c,tem;
                              //la methode de  gauss
                              for(k=0;k<M-1;k++)
                              {if(A[k][k]==0)
                              {cout<<"il ya un pivot nul\n";
                              exit(EXIT_FAILURE);
                              }
                              else
                              {for(i=k+1;i<M;i++)
```

```
                                {c=A[i][k]/A[k][k];
                                for(j=k;j<M;j++)
                                A[i][j]=A[i][j]-c*A[k][j];
                                B[i]=B[i]-c*B[k];
                                }
                                }
                                }
                                for(i=M-1;i>=0;i--)
                                {som=0;
                                for(j=i+1;j<M;j++)
                                som+=A[i][j]*v[j];
                                v[i]=(B[i]-som)/A[i][i];
        }
                        cout<<"la matrice A et le vecteur B aprés l'élimination de gauss':\n";
                        afficher();
                        cout<<"\n";
                        cout<< "la solution du système est:\n";
                                        for(j=0;j<M;j++)
                                {
                                cout<<"v["<<j+1<<"]="<<v[j]<<"     ";
                                }
                                  cout<<"\n";
                        }
                };

int main(int argc, char** argv)
 {
            /*creation de l'objet*/
   eq_poisson_2d   l;
            //donner la dimension
            l.taille();
   //remplir Aet B
   l.eq_poisson1();
   //afficher A et B
   l.afficher();
   //fait la factorisation
            l.gauss();
            return 0;
}
```