

```

                                                                    Labo3
//////////////////////////////////////
//NOM:Delplanque                                                    //
//PRENOM:Yoann                                                       //
//2° INFO                                                            //
//labo 3                                                             //
//( voir feuille d'énoncé )                                         //
//////////////////////////////////////

#include<iostream.h>
#include<stdiostr.h>
#include<string.h>

class matrice{
private:
    int lignes;
    int colonnes;
    int tab[10][10];          //taille MAX du tableau

public:
    matrice(int l=1,int c=1);
    matrice(matrice &m);

    void saisir(int l,int c);
    void afficher(int l,int c);
    bool tailleidentique(matrice m1,matrice m2);
    bool produitpossible(matrice m1,matrice m2);

    friend matrice operator+(matrice m1,matrice m2);
    friend matrice operator-(matrice m1,matrice m2);
    friend matrice operator*(matrice m1,matrice m2);
    friend matrice operator*(int n,matrice m1);
};

//////////////////////////////////////

matrice::matrice(int l,int c) //initialisation
{
    lignes=l;
    colonnes=c;
    for (int i=0;i<l;i++)
    {
        for(int j=0;j<c;j++)
            tab[i][j]=0;
    }
}

matrice::matrice(matrice &m) //constructeur par copie
{
    lignes=m.lignes;
    colonnes=m.colonnes;
    for (int i=0;i<lignes;i++)
    {
        for(int j=0;j<colonnes;j++)
            tab[i][j]=m.tab[i][j];
    }
}

void matrice::saisir(int l,int c)
{
    lignes=l;
    colonnes=c;
    for (int i=0;i<lignes;i++)
    {
        for(int j=0;j<colonnes;j++)
        {
            cout<<"entrez une valeur pour tab["<<i<<"]["<<j<<":";
            cin>>tab[i][j];
        }
    }
}

void matrice::afficher(int l,int c)
{
    lignes=l;
    colonnes=c;
    for (int i=0;i<lignes;i++)
    {

```

labo3

```

    for(int j=0;j<colonnes;j++)
    {
        cout<<"tab["<<i<<"]["<<j<<"]="<<tab[i][j]<<endl;
    }
}

```

```

bool matrice::tailleidentique(matrice m1,matrice m2)
{
    if((m1.lignes==m2.lignes)|| (m1.colonnes==m2.colonnes))
    {
        return true;
    }
    else
    {
        return false;
    }
}

```

```

//pour qu'on puisse faire un produit,
//la condition est que le nbre de colonnes
//de la première matrice soit égale au nbre
//de ligne de la deuxième

```

```

bool matrice::produitpossible(matrice m1,matrice m2)
{
    if(m1.colonnes==m2.lignes) return true;
    else return false;
}

```

```

matrice operator+(matrice m1,matrice m2)
{
    //matrice resultat qui contient le nbre de lignes
    //et de colonnes de la première matrice
    matrice res(m1.lignes,m1.colonnes);
    for(int i=0;i<m1.lignes;i++)
    {
        for(int j=0;j<m1.colonnes;j++)
        {
            res.tab[i][j]=m1.tab[i][j]+m2.tab[i][j];
        }
    }
    return res;
}

```

```

matrice operator-(matrice m1,matrice m2)
{
    matrice res(m1.lignes,m1.colonnes);
    for(int i=0;i<m1.lignes;i++)
    {
        for(int j=0;j<m1.colonnes;j++)
        {
            res.tab[i][j]=m1.tab[i][j]-m2.tab[i][j];
        }
    }
    return res;
}

```

```

matrice operator*(int n,matrice m1)
{
    matrice res(m1.lignes,m1.colonnes);
    for (int i=0;i<m1.lignes;i++)
    {
        for(int j=0;j<m1.colonnes;j++)
        {
            res.tab[i][j]=n*m1.tab[i][j];
        }
    }
    return res;
}

```

```

//voir le labo sur les matrices fait
//en 1°année pour comprendre la formule
//de la multiplication de matrices

```

```

matrice operator*(matrice m1,matrice m2)
{
    //lignes de la première matrice et colonnes de la deuxième
    matrice res(m1.lignes,m2.colonnes);
    for(int i=0;i<m1.lignes;i++)
    {
        for(int j=0;j<m2.colonnes;j++)
        {
            for(int k=0;k<m1.colonnes;k++)
            {
                res.tab[i][j]=res.tab[i][j]+m1.tab[i][k]*m2.tab[k][j];
            }
        }
    }
    return res;
}

////////////////////////////////PROGRAMME PRINCIPAL////////////////////////////////

void main()
{
    matrice res;    //matrice resultat
    int choix;

    int cols1,cols2,lines1,lines2;

    cout<<"Nombre de lignes pour m1?";    // déclaration de la
    cin>>lines1;                            // taille de la
    cout<<"Nombres de colonnes pour m1?";  // première matrice
    cin>>cols1;                               //

    matrice m1(lines1,cols1);
    m1.saisir(lines1,cols1);
    cout<<endl;
    m1.afficher(lines1,cols1);
    cout<<endl;

    cout<<"Nombre de lignes pour m2?";    //
    cin>>lines2;                            // déclaration de la
    cout<<"Nombres de colonnes pour m2 ?"; // taille de la
    cin>>cols2;                               // deuxième matrice

    matrice m2(lines2,cols2);
    m2.saisir(lines2,cols2);
    cout<<endl;
    m2.afficher(lines2,cols2);
    cout<<endl;

    do{
        cout<<endl;
        cout<<"****MENU****"<<endl;
        cout<<"1 pour additionner 2 matrices"<<endl;
        cout<<"2 pour soustraire 2 matrices"<<endl;
        cout<<"3 pour multiplier 2 matrices"<<endl;
        cout<<"4 pour multiplier 1 matrice par un entier"<<endl;
        cout<<"5 pour quitter"<<endl;
        cin>>choix;
    }while(choix!=5);

    switch(choix){
        case 1:
            if(res.tailleidentique(m1,m2)==true)
            {
                cout<<"taille identique,on peut additionner"<<endl;
                res=m1+m2;
                res.afficher(lines1,cols1);
            }
            else{
                cout<<"on ne peut pas additionner car pas meme type!"<<endl;
            }
            break;

        case 2:
            if(res.tailleidentique(m1,m2)==true)

```

```

                                labo3
                                {
                                cout<<"taille identique,on peut soustraire"<<endl;
                                res=m1-m2;
                                res.afficher(lines1,cols1);
                                }
                                else{
                                cout<<"on ne peut pas soustraire car pas meme type!"<<endl;
                                }
                                break;

                                case 3:
                                if(res.produitpossible(m1,m2)==true)
                                {
                                cout<<"il y a possibilite de faire le produit des 2 matrices!"<<endl;
                                cout<<"la matrice sera de style "<<lines1<<" x "<<cols2<<endl;
                                res=m1*m2;
                                res.afficher(lines1,cols2);
                                }
                                else
                                {
                                cout<<"impossible de faire le produit\n";
                                }
                                break;

                                case 4: int line,col,entier;
                                cout<<"Nombre de colonnes pour m?";
                                cin>>col;
                                cout<<"Nombres de lignes pour m?";
                                cin>>line;

                                matrice m(line,col);
                                m.saisir(line,col);
                                cout<<endl;
                                m.afficher(line,col);
                                cout<<endl;

                                cout<<"entre l'entier qui va multiplier votre matrice:";
                                cin>>entier;
                                res=entier*m;
                                res.afficher(line,col);
                                break;

                                case 5: break;

                                }
                                }while(choix!=5);
                                }

```