

```

                                                                    labo4
//////////////////////////////////////
//NOM:Delplanque                                                    //
//PRENOM:Yoann                                                       //
//2° INFO                                                            //
//labo 4                                                             //
//idem labo3 mais avec des surcharges supplémentaires             //
//( voir feuille d'énoncé )                                          //
//////////////////////////////////////

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

class matrice{
private:
    int lignes;
    int colonnes;
    int tab[10][10];
    bool tailleidentique(matrice m1,matrice m2);
    bool produitpossible(matrice m1,matrice m2);

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

    friend ostream& operator<<(ostream& o,matrice m1);
    friend istream& operator>>(istream& ii,matrice& m1);

    matrice& operator+=(matrice& m1);
    matrice& operator-=(matrice& m1);
    matrice& operator*=(matrice& m1);
    friend bool operator==(matrice m1,matrice m2);
    matrice operator++();
    matrice operator++(int);
    matrice operator--();
    matrice operator--(int);
};

//////////////////////////////////////
//      définition des méthodes de la classe //
//      //
//////////////////////////////////////

matrice::matrice(int l,int c)
{
    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)
{
    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];
    }
}

ostream& operator<<(ostream& o,matrice m1)
{
    for (int i=0;i<m1.lignes;i++)
    {
        for(int j=0;j<m1.colonnes;j++)
        {
            o<<"tab["<<i<<"]["<<j<<"]="<<m1.tab[i][j]<<endl;
        }
    }
    o<<endl;
    return o;
}

```

```

istream& operator>>(istream& ii,matrice& m1){
    for (int i=0;i<m1.lignes;i++)
    {
    for(int j=0;j<m1.colonnes;j++)
    {
    cout<<"entrez une valeur pour tab["<<i<<"]["<<j<<"]:";
    ii>>m1.tab[i][j];
    }
    }
    return ii;
}

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

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

matrice& matrice::operator+=(matrice& m1)
{
    //on vérifie la taille de l'objet courant avec la matrice m1
    if(tailleidentique(*this,m1)==true)
    {
    cout<<"on peut additionner\n";
    for(int i=0;i<m1.lignes;i++)
    {
        for(int j=0;j<m1.colonnes;j++)
        {
            tab[i][j]=tab[i][j]+m1.tab[i][j];
        }
    }
    return *this;
    }
    else cout<<"impossible\n";
}

matrice& matrice::operator--=(matrice& m1)
{
    //on vérifie la taille de l'objet courant avec la matrice m1
    if(tailleidentique(*this,m1)==true)
    {
    cout<<"on peut soustraire\n";
    for(int i=0;i<m1.lignes;i++)
    {
        for(int j=0;j<m1.colonnes;j++)
        {
            tab[i][j]=tab[i][j]-m1.tab[i][j];
        }
    }
    return *this; //on retourne l'objet courant
    }
    else cout<<"impossible\n";
}

matrice& matrice::operator*=(matrice& m1)
{
    //on teste si un produit peut se faire,
    //teste sur l'objet courant et m1
    if(produitpossible(*this,m1)==true)
    {
    cout<<"on peut faire le produit\n";

```

```

        for(int i=0;i<m1.lignes;i++)
        {
            for(int j=0;j<m1.colonnes;j++)
            {
                tab[i][j]=tab[i][j]*m1.tab[i][j];
            }
        }
        return *this;
    }
    else cout<<"impossible\n";
}

//on vérifie la taille des matrices:
//si égales, on compare chaque éléments
//si pas égales, on fait rien
bool operator==(matrice m1,matrice m2)
{
    if(m1.tailleidentique(m1,m2)==true)
    {
        cout<<"on peut comparer\n";
        for(int i=0;i<m1.lignes;i++)
        {
            for(int j=0;j<m1.colonnes;j++)
            {
                if(m1.tab[i][j]==m2.tab[i][j]) return true;
                else return false;
            }
        }
    }
    else cout<<"on peut pas comparer\n";
}

//pré-incrémentation
matrice matrice::operator++()
{
    for(int i=0;i<lignes;i++)
    {
        for(int j=0;j<colonnes;j++)
        {
            ++tab[i][j];
        }
    }
    return *this;
}

//post-incrémentation
matrice matrice::operator++(int)
{
    matrice tmp=*this;
    for(int i=0;i<lignes;i++)
    {
        for(int j=0;j<colonnes;j++)
        {
            tab[i][j]++;
        }
    }
    return tmp;
}

//pré-décrémententation
matrice matrice::operator--()
{
    for(int i=0;i<lignes;i++)
    {
        for(int j=0;j<colonnes;j++)
        {
            --tab[i][j];
        }
    }
    return *this;
}

//post-décrémententation
matrice matrice::operator--(int)
{

```

```

matrice tmp=*this;
for(int i=0;i<lignes;i++)
{
    for(int j=0;j<colonnes;j++)
    {
        tab[i][j]--;
    }
}
return tmp;
}

```

```

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

```

```

void main()
{
matrice res;
matrice m1(2,2); //j'ai défini une matrice 2x2
matrice m2(2,2);
cin>>m1;
cout<<m1;
cin>>m2;
cout<<m2;

```

```

int choix;

```

```

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 comparer 2 matrices"<<endl;
cout<<"5 pour incrementer les 2 matrices"<<endl;
cout<<"6 pour decrementer les 2 matrices"<<endl;
cout<<"7 pour quitter"<<endl;
cin>>choix;

```

```

switch(choix){

```

```

    case 1:

```

```

        m1+=m2;
        cout<<m1;
        break;

```

```

    case 2:

```

```

        m1-=m2;
        cout<<m1;
        break;

```

```

    case 3:

```

```

        m1*=m2;
        cout<<m1;
        break;

```

```

    case 4:

```

```

        if(m1==m2)
        {
            cout<<"les matrices sont egales\n";
            cout<<m1;
            cout<<m2;
        }
        else cout<<"les matrices sont differentes\n";
        break;

```

```

    case 5:

```

```

        cout<<"version pre-incrementation:\n";
        ++m1;
        ++m2;
        cout<<m1;
        cout<<m2;
        cout<<"version post incrementation:\n";
        m1++;
        m2++;
        cout<<m1;

```

```

labo4
cout<<m2;
break;

case 6:
    cout<<"version pre-decrementation:\n";
    --m1;
    --m2;
    cout<<m1;
    cout<<m2;
    cout<<"version post-decrementation:\n";
    m1--;
    m2--;
    cout<<m1;
    cout<<m2;

case 7:break;

}
}while(choix!=7);
}

```