

Travaux Pratiques

EXERCICES



✚ Ce TP réalisé par :

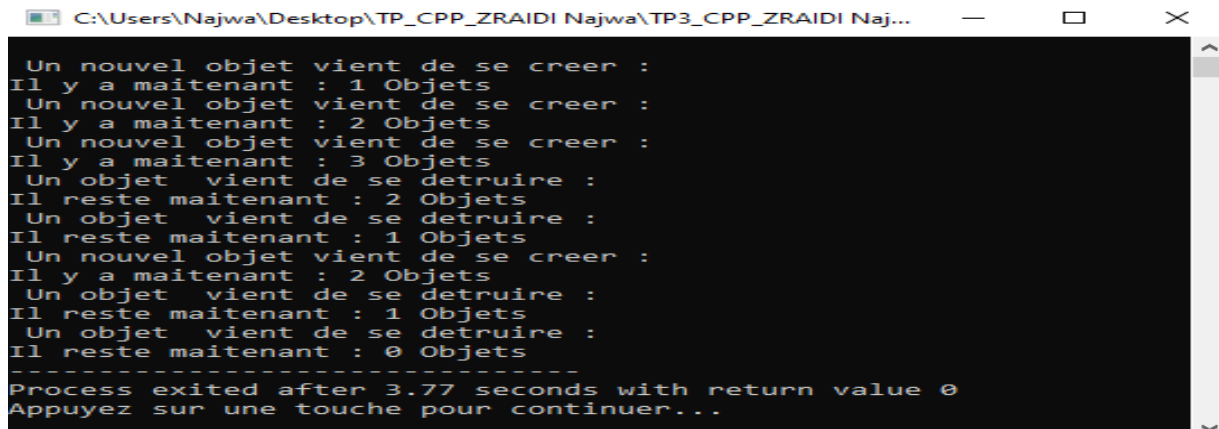
ZRAIDI NAJWA
(GLSID1)

EXERCICE 1:

- ❖ L'objectif de ce programme permettant à tout moment de connaître le nombre d'objets existants. Pour se faire ,j'ai déclaré statique un membre de la classe Compteur appelé ctr. Sa valeur incrémentée de 1 chaque appel du constructeur et décrémentée de 1 à chaque appel du destructeur .

```
#include<iostream>
#include<conio.h>
using namespace std;
class Compteur {
    static int ctr;
    public :
        Compteur();
        ~ Compteur();
};
int Compteur::ctr = 0;// initialisation de variable static ctr  dans la
classe Compteur
Compteur :: Compteur()
{
    cout<<"\n Un nouvel objet vient de se creer : "<<endl;
    cout<<"Il y a maintenant : "<<++ctr<<" Objets ";
    getch();}
Compteur :: ~ Compteur()
{
    cout<<"\n Un objet  vient de se detruire : "<<endl;
    cout<<"Il reste maintenant : "<<--ctr<<" Objets ";
    getch();}
void Essai(){    Compteur u,v; }
int main(){
    Compteur a;
    Essai();
    Compteur b;
    return 0;}
```

- ❖ L'exécution de programme donne :



```
Un nouvel objet vient de se creer :
Il y a maintenant : 1 Objets
Un nouvel objet vient de se creer :
Il y a maintenant : 2 Objets
Un nouvel objet vient de se creer :
Il y a maintenant : 3 Objets
Un objet  vient de se detruire :
Il reste maintenant : 2 Objets
Un objet  vient de se detruire :
Il reste maintenant : 1 Objets
Un nouvel objet vient de se creer :
Il y a maintenant : 2 Objets
Un objet  vient de se detruire :
Il reste maintenant : 1 Objets
Un objet  vient de se detruire :
Il reste maintenant : 0 Objets
-----
Process exited after 3.77 seconds with return value 0
Appuyez sur une touche pour continuer...
```

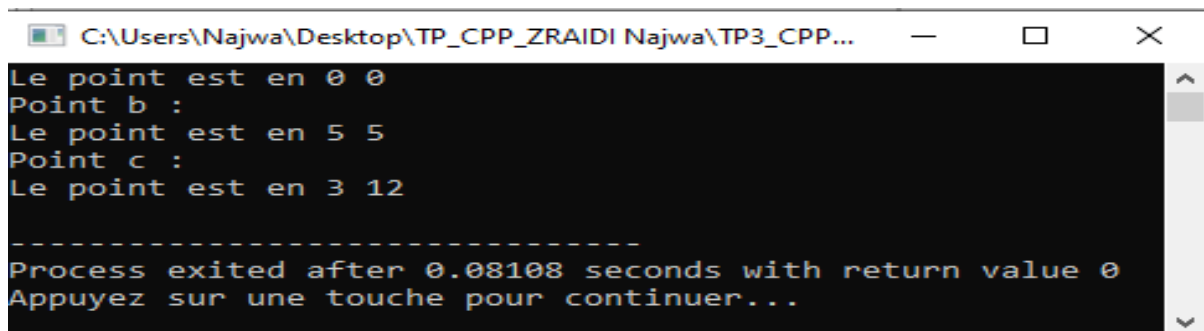
EXERCICE 2:

- ❖ L'objectif de ce programme est d'utiliser plusieurs constructeurs et des méthodes de même nom « point() » et « affiche() » mais avec des différents arguments .

```
#include<iostream>
using namespace std;
class point {
    int x,y;
    public:
        point();//constructeur 1
        point(int);//constructeur 2
        point(int,int);//constructeur 3
        void affiche();
        void affiche(char *);
};

point :: point() //constructeur 1
{
    x=0;
    y=0;
}
point :: point(int abs) //constructeur 2
{
    x=abs;
    y=abs;
}
point :: point(int abs,int ord) //constructeur 3
{
    x=abs;
    y=ord;
}
void point :: affiche(){
    cout<<"Le point est en "<< x<<" "<<y<<endl;
};
void point :: affiche(char *message){
    cout<<message;
    affiche();
}
int main(){
    point a;
    a.affiche();
    point b(5);
    b.affiche("Point b : \n");
    point c(3,12);
    c.affiche("Point c : \n");
}
```

- ❖ L'exécution de programme donne :



```
C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP...
Le point est en 0 0
Point b :
Le point est en 5 5
Point c :
Le point est en 3 12

-----
Process exited after 0.08108 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXERCICE 3:

- ❖ L'objectif de ce programme est d'utiliser les fonctions en ligne au sien de la classe .

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

class point
{
    int x,y;
    public :
        point(){ //constructeur 1
            x=0;
            y=0;
        }
        point(int abs) //constructeur 2
        {
            x=abs;
            y=abs;
        }
        point(int abs,int ord) //constructeur 3
        {
            x=abs;
            y=ord;
        }
        void affiche();
};

void point :: affiche(){
    cout<<"Le point est en "<< x<<" "<<y<<endl;
};

int main(){
    point a,b(5);
    cout<<"Le point A :\n ";
    a.affiche();
    cout<<"Le point B :\n ";
```

```

b.affiche();
point c(3,12);
cout<<"Le point C :\n ";
c.affiche();
}

```

❖ L'exécution de programme donne :

```

C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP...
Le point A :
Le point est en 0 0
Le point B :
Le point est en 5 5
Le point C :
Le point est en 3 12

-----
Process exited after 0.05801 seconds with return value 0
Appuyez sur une touche pour continuer...

```

❖ La comparaison entre le programme de l'exercice 2 et le programme de cet exercice :

Exercice2.cpp	08/04/2022 16:01	C++ Source File	1 Ko
Exercice2.o	10/04/2022 14:42	Fichier O	<u>4 Ko</u>
Exercice3.cpp	08/04/2022 17:16	C++ Source File	1 Ko
Exercice3.o	10/04/2022 14:42	Fichier O	<u>5 Ko</u>

➤ La taille de fichier qui contient les méthodes offlines > La taille de fichier qui contient les méthodes en ligne .

EXERCICE 4:

❖ L'objectif de ce programme est d'introduire une fonction membre nommée « coïncidence » .Elle permet de détecter la coïncidence éventuelle entre deux points et qui a comme paramètre un objet de classe **point** .

```

#include<iostream>
using namespace std;
class point
{
    int x,y;
    public :
    point(int abs=0,int ord=0) //constructeur
    {
        x=abs;

```

```

    y=ord;
    }
    int coincidence(point);
};

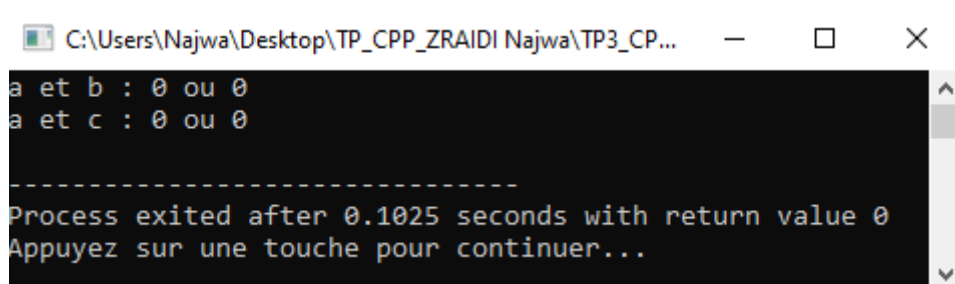
int point :: coincidence(point pt){ //passage de paramètres par valeur

    if((pt.x==x)&&(pt.y==y))
        /*pt.x acces d'absice de point
        pt et pt.x acces d'ordonnee de point pt*/
        return 1;
    else
        return 0;
}

int main(){
    int test1,test2;
    point a,b(1),c(0,2);
    test1=a.coincidence(b);
    test2=b.coincidence(a);
    cout<<"a et b : "<<test1 <<" ou "<<test2<<endl;
    test1=a.coincidence(c);
    test2=c.coincidence(a);
    cout<<"a et c : "<<test1 <<" ou "<<test2<<endl;
}

```

- ❖ L'exécution de programme donne :



```

C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CP...
a et b : 0 ou 0
a et c : 0 ou 0

-----
Process exited after 0.1025 seconds with return value 0
Appuyez sur une touche pour continuer...

```

EXERCICE 5:

- ❖ L'objectif de programme de modifier la fonction membre « coïncidence » de l'exercice précédent de sorte que son prototype devienne **int point :: coincidence(point *adpt)**

```

#include<iostream>
using namespace std;
class point
{
    int x,y;

```

```

    public :
    point(int abs=0,int ord=0) //constructeur
    {
        x=abs;
        y=ord;
    }
    int coincidence(point *); //passage paramètres par adresse
};

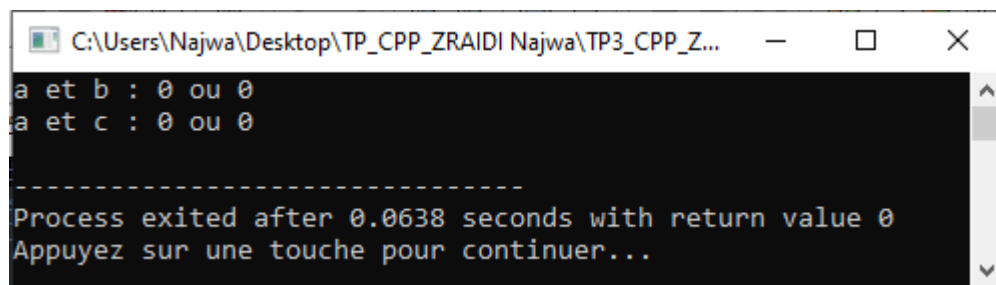
int point :: coincidence(point *pt){

    if((pt->x==x)&&(pt->y==y))
        /*pt.x acces d'absice de point
        pt et pt.x acces d'ordonnee de point pt*/
        return 1;
    else
        return 0;
}

int main(){
    int test1,test2;
    point a,b(1),c(0,2);
    test1=a.coincidence(&b);
    test2=b.coincidence(&a);
    cout<<"a et b : "<<test1 <<" ou "<<test2<<endl;
    test1=a.coincidence(&c);
    test2=c.coincidence(&a);
    cout<<"a et c : "<<test1 <<" ou "<<test2<<endl;
}

```

❖ L'exécution de programme donne :



```

C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP_Z...
a et b : 0 ou 0
a et c : 0 ou 0

-----
Process exited after 0.0638 seconds with return value 0
Appuyez sur une touche pour continuer...

```

EXERCICE 6:

❖ L'objectif de programme de modifier la fonction membre « coincidence » de l'exercice précédent de sorte que son prototype devienne **int point :: coincidence(point &adpt)**

```

#include<iostream>
#include<conio.h>
using namespace std;
class point
{
    int x,y;
    public :
    point(int abs=0,int ord=0) //constructeur
    {
        x=abs;
        y=ord;
    }
    int coincidence(point &);//passage de paramètres par référence
};

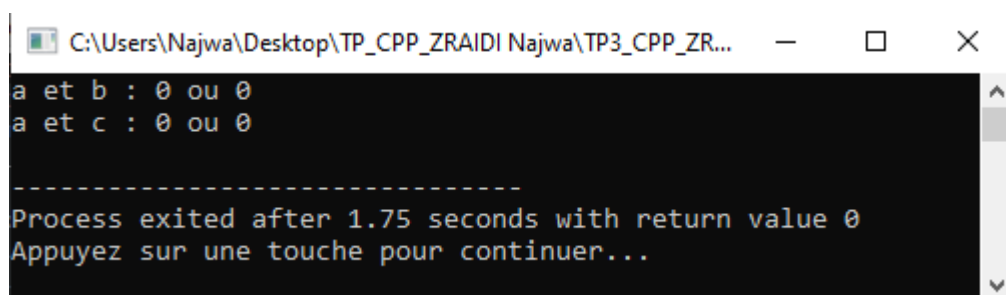
int point :: coincidence(point &pt){

    if((pt.x==x)&&(pt.y==y))
        /*pt.x accès d'abscisse de point
        pt et pt.x accès d'ordonnée de point pt*/
        return 1;
    else
        return 0;
}

int main(){
    int test1, test2;
    point a,b(1),c(0,2);
    test1=a.coincidence(b);
    test2=b.coincidence(a);
    cout<<"a et b : "<<test1 <<" ou "<<test2<<endl;
    test1=a.coincidence(c);
    test2=c.coincidence(a);
    cout<<"a et c : "<<test1 <<" ou "<<test2<<endl;
    getch();
}

```

❖ L'exécution de programme donne :



```

C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP_ZR...
a et b : 0 ou 0
a et c : 0 ou 0

-----
Process exited after 1.75 seconds with return value 0
Appuyez sur une touche pour continuer...

```


EXERCICE 7_1 :

- ❖ L'objectif de ce programme est d'introduire une fonction membre nommée « det » .Elle permet de calculer le déterminant de deux vecteurs qui a comme paramètre un objet de classe **vecteur**.

```
#include<iostream>
#include<conio.h>
using namespace std;

class vecteur
{
    float x,y;
    public :
        vecteur(float,float);
        void homotethie(float);
        float det(vecteur);
        void affiche();
};

vecteur :: vecteur(float abs=0.,float ord =0.){
    x=abs;
    y=ord;
}

void vecteur :: homotethie(float val){
    x=x*val;
    y=y*val;
}

void vecteur :: affiche(){ cout<<"x = "<<x<<" y= "<<y<<endl;}
float vecteur:: det(vecteur v1){ return v1.x * y - x *v1.y;}
int main(){
    vecteur v1(1,2),v2(0.5,4);
    cout<<"Le vecteur 1 : "<<endl;
    v1.affiche();
    cout<<"Le vecteur 2 : "<<endl;
    v2.affiche();
    cout<<"Les valeurs nouvelles de la vecteur 1 : "<<endl;
    v1.homotethie(2);
    v1.affiche();
    cout<< "Le determinant des deux vecteurs v1 et v2 : \n";
    cout <<"det(v1,v2) = "<<v1.det(v2);}
```

- ❖ L'exécution de programme donne :

```
C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP_ZRAI...
Le vecteur 1 :
x = 1 y= 2
Le vecteur 2 :
x = 0.5 y= 4
Les valeurs nouvelles de la vecteur 1 :
x = 2 y= 4
Le determinant des deux vecteurs v1 et v2 :
det(v1,v2) = -6
-----
Process exited after 0.06584 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXERCICE 7_2 :

- ❖ L'objectif de programme de modifier la fonction membre « det » de l'exercice précédent de sorte que son prototype devienne **float vecteur :: det(vecteur &v1)**.

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

class vecteur
{
    float x,y;
    public :
        vecteur(float,float);
        void homotethie(float);
        float det(vecteur &);//passage par reference
        void affiche();
};

vecteur :: vecteur(float abs=0.,float ord =0.){
    x=abs;
    y=ord;
}

void vecteur :: homotethie(float val){
    x=x*val;
    y=y*val;
}

void vecteur :: affiche(){
    cout<<"x = "<<x<<" y= "<<y<<endl;
}

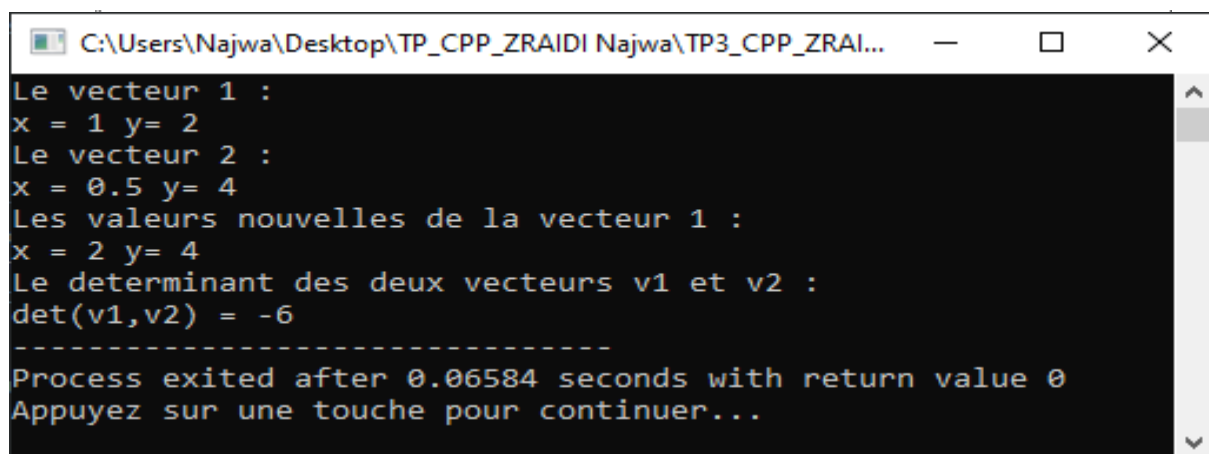
float vecteur:: det(vecteur &v1)
{
    return v1.x * y - x *v1.y;
}
```

```

}
int main(){
    vecteur v1(7,2),v2(8.2,3);
    cout<<"Le vecteur 1 : "<<endl;
    v1.affiche();
    cout<<"Le vecteur 2 : "<<endl;
    v2.affiche();
    cout<<"Les valeurs nouvelles de la vecteur 1 : "<<endl;
    v1.homotethie(2);
    v1.affiche();
    cout<< "Le determinant des deux vecteurs v1 et v2 : \n";
    cout <<"det(v1,v2) = "<<v1.det(v2);
}

```

❖ L'exécution de programme donne :



```

C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP_ZRAI...
Le vecteur 1 :
x = 1 y= 2
Le vecteur 2 :
x = 0.5 y= 4
Les valeurs nouvelles de la vecteur 1 :
x = 2 y= 4
Le determinant des deux vecteurs v1 et v2 :
det(v1,v2) = -6
-----
Process exited after 0.06584 seconds with return value 0
Appuyez sur une touche pour continuer...

```

EXERCICE 7_3 :

❖ L'objectif de programme de modifier la fonction membre « det » de l'exercice précédent de sorte que son prototype devienne **float vecteur :: det(vecteur *v1)**.

```

#include<iostream>
using namespace std;
class vecteur
{
    float x,y;
    public :
        vecteur(float,float);
        void homotethie(float);
        float det(vecteur *); //passage par reference
        void affiche();
};
vecteur :: vecteur(float abs=0.,float ord =0.){
    x=abs;
    y=ord;
}

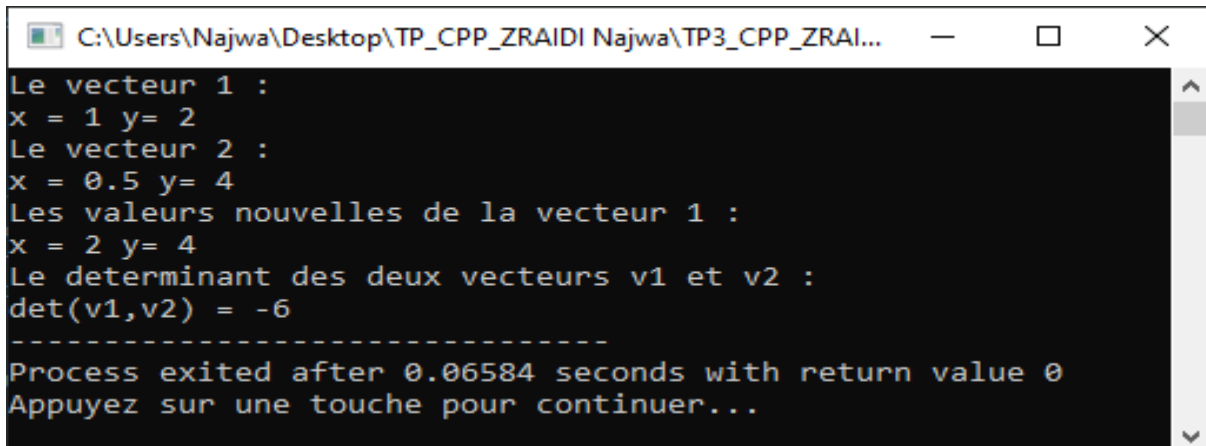
```

```

void vecteur :: homotethie(float val){
    x=x*val;
    y=y*val;
}
void vecteur :: affiche(){
    cout<<"x = "<<x<<" y= "<<y<<endl;
}
float vecteur:: det(vecteur *v1)
{
    return v1->x * y - x *v1->y;
}
int main(){
    vecteur v1(1,2),v2(0.5,4);
    cout<<"Le vecteur 1 : "<<endl;
    v1.affiche();
    cout<<"Le vecteur 2 : "<<endl;
    v2.affiche();
    cout<<"Les valeurs nouvelles de la vecteur 1 : "<<endl;
    v1.homotethie(2);
    v1.affiche();
    cout<< "Le determinant des deux vecteurs v1 et v2 : \n";
    cout <<"det(v1,v2) = "<<v1.det(&v2);
}

```

❖ L'exécution de programme donne :



```

C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP_ZRAI...
Le vecteur 1 :
x = 1 y= 2
Le vecteur 2 :
x = 0.5 y= 4
Les valeurs nouvelles de la vecteur 1 :
x = 2 y= 4
Le determinant des deux vecteurs v1 et v2 :
det(v1,v2) = -6
-----
Process exited after 0.06584 seconds with return value 0
Appuyez sur une touche pour continuer...

```

EXERCICE 8_1:

❖ L'objectif de ce programme est d'introduire une fonction membre nommée « homothétie ». Elle permet de modifier la valeur de vecteur qui a comme paramètre une valeur donnée par utilisateur .

```

#include<iostream>
using namespace std;

```

```

class vecteur
{
    float x,y;
    public :
        vecteur(float,float);
        vecteur homotethie(float);
        void affiche();
};

vecteur :: vecteur(float abs=0.,float ord =0.){
    x=abs;
    y=ord;
}

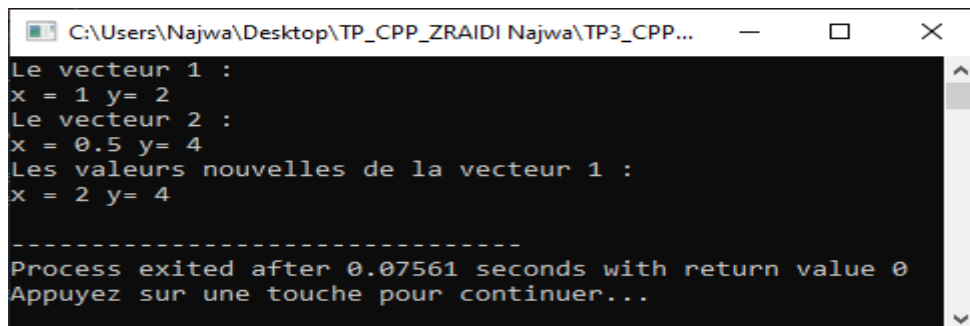
vecteur vecteur :: homotethie(float val){
    x=x*val;
    y=y*val;
}

void vecteur :: affiche(){
    cout<<"x = "<<x<<" y= "<<y<<endl;
}

int main(){
    vecteur v1(1,2),v2(0.5,4);
    cout<<"Le vecteur 1 : "<<endl;
    v1.affiche();
    cout<<"Le vecteur 2 : "<<endl;
    v2.affiche();
    cout<<"Les valeurs nouvelles de la vecteur 1 : "<<endl;
    v1.homotethie(2);
    v1.affiche();
}

```

❖ L'exécution de programme donne :



```

C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP...
Le vecteur 1 :
x = 1 y= 2
Le vecteur 2 :
x = 0.5 y= 4
Les valeurs nouvelles de la vecteur 1 :
x = 2 y= 4
-----
Process exited after 0.07561 seconds with return value 0
Appuyez sur une touche pour continuer...

```

EXERCICE 8_2:

- ❖ L'objectif de programme de modifier la fonction membre « homothétie » de l'exercice précédent de sorte que son prototype devienne **vecteur *vecteur :: homothétie (float val)**.

```
#include<iostream>
#include<conio.h>
using namespace std;

class vecteur
{
    float x,y;
    public :
        vecteur(float,float);
        vecteur *homotethie(float);
        void affiche();
};

vecteur :: vecteur(float abs=0.,float ord =0.){
    x=abs;
    y=ord;
}

vecteur *vecteur :: homotethie(float val){
    vecteur *res=new vecteur;
    res->x=x*val;
    res->y=y*val;
    return res;
}

void vecteur :: affiche(){
    cout<<"x = "<<x<<" y= "<<y<<endl;
}

int main(){
    vecteur v1(1,2),v2(0.5,4);
    cout<<"Le vecteur 1 : "<<endl;
    v1.affiche();
    cout<<"Le vecteur 2 : "<<endl;
    v2.affiche();
    cout<<"Les valeurs nouvelles de la vecteur 1 : "<<endl;
    v1=*v1.homotethie(2);
    v1.affiche();}
```

- ❖ L'exécution de programme donne :

```
C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP...
Le vecteur 1 :
x = 1 y= 2
Le vecteur 2 :
x = 0.5 y= 4
Les valeurs nouvelles de la vecteur 1 :
x = 2 y= 4

-----
Process exited after 0.07561 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXERCICE 8_2:

- ❖ L'objectif de programme de modifier la fonction membre « homothétie » de l'exercice précédent de sorte que son prototype devienne **vecteur &vecteur :: homothétie (float val)**.

```
#include<iostream>
using namespace std;
class vecteur
{
    float x,y;
public :
    vecteur(float,float);
    vecteur &homotethie(float);
    void affiche();
};

vecteur :: vecteur(float abs=0.,float ord =0.){
    x=abs;
    y=ord;
}

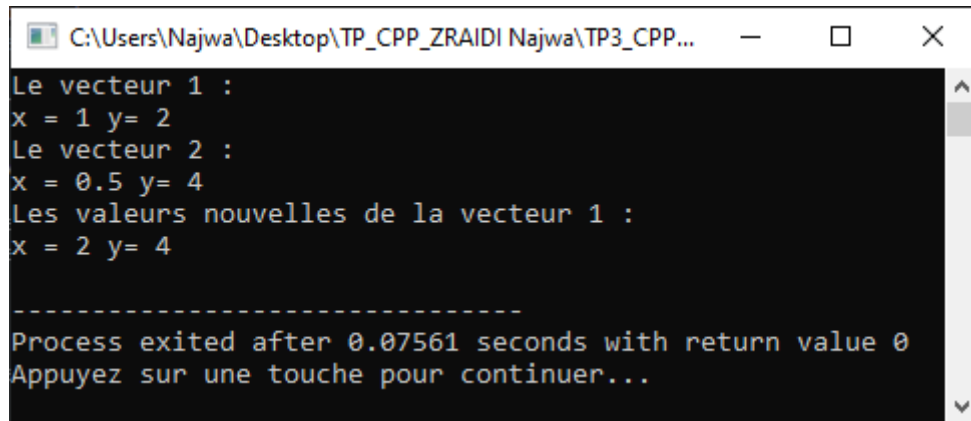
vecteur &vecteur :: homotethie(float val){
    static vecteur res;
    res.x=x*val;
    res.y=y*val;
    return res;
}

void vecteur :: affiche(){
    cout<<"x = "<<x<<" y= "<<y<<endl;
}

int main(){
    vecteur v1(1,2),v2(0.5,4);
    cout<<"Le vecteur 1 : "<<endl;
    v1.affiche();
    cout<<"Le vecteur 2 : "<<endl;
    v2.affiche();
}
```

```
cout<<"Les valeurs nouvelles de la vecteur 1 : "<<endl;  
v1=v1.homotethie(2);  
v1.affiche();}
```

❖ L'exécution de programme donne :



```
C:\Users\Najwa\Desktop\TP_CPP_ZRAIDI Najwa\TP3_CPP...  
Le vecteur 1 :  
x = 1 y= 2  
Le vecteur 2 :  
x = 0.5 y= 4  
Les valeurs nouvelles de la vecteur 1 :  
x = 2 y= 4  
-----  
Process exited after 0.07561 seconds with return value 0  
Appuyez sur une touche pour continuer...
```