

10/20

LHEUREUX.cpp

Page 1/4

```

/*****
Cercle.h
*****/
#pragma once
#include <math.h>
#include "Point2D.h"
#include "Figure.h"

#define PI 3.14159265358979323846

class Point2D;
class Figure;

class Cercle :Figure
{
private :
    double rayon;
    Point2D centre;

public:
    Cercle(double LeRayon, Point2D leCentre);
    double getPerimetre();
    double getSurface();
};

/*****
Commande.h
*****/
#pragma once
#define _Use_MATH_DEFINES
#include <vector>
#include <string>
#include "Figure.h"

using namespace std;

class Figure;

class Commande
{
private:
    bool commandeTerminee;
    double prixMetreDecoupe , prixMetreCarreMatiere ;
    string idCommande;
    vector<Figure*> lesFigures;

public:
    Commande(string identifiantCommande , double lePrixMetreDecoupe , double lePrixMetreCarreMatiere);
    string getIdCommande() { return idCommande; }
    void ajouterNouvelleFigure(Figure* laFigure);
    void cloturerCommande();

    double getPrix() ;
};

/*****
Figure.h
*****/
#pragma once

class Figure
{
public:
    double getPerimetre();
    double getSurface();
};

/*****
Point2D.h
*****/
// Cette classe n'est pas à modifier
#pragma once

class Point2D
{
private:
    double x , y ;

public:

```

1 point

Ce sont des méthodes
virtuelles pures

```

        Point2D(double x=0 , double y=0);
        double getX();
        double getY();
        void setX(double newX);
        void setY(double newY);
};

/*****
Polygone.h
*****/
#pragma once

#include <vector>
#include "Figure.h"
#include "Point2D.h"

using namespace std;

class Point2D;
class Figure;

#define abs(x) ( (x) >=0 ? (x) : -(x) )

class Polygone :Figure
{
protected:
    vector<Point2D *> lesSommets;
    bool estFerme;

public:
    Polygone(void);
    static double distance( const Point2D *p1, const Point2D *p2);
    void insereUnNouveauSommet( Point2D *leSommet, int position =-1);
    void fermeLePolygone();
    double getPerimetre();
    double getSurface();
};

/*****
Cercle.cpp
*****/
#include "Cercle.h"

Cercle::Cercle(double leRayon, Point2D leCentre)
{
    this->rayon = leRayon;
    this->centre = leCentre ;
}

double Cercle::getPerimetre()
{
    return 2 * PI * rayon;
}

double Cercle::getSurface()
{
    return PI * rayon * rayon;
}

/*****
Commande.cpp
*****/
#include "Commande.h"

Commande::Commande(string identifiantCommande , double lePrixMetreDecoupe , double lePrixMetreCarreMatiere)
{
    this->idCommande = identifiantCommande;
    this->prixMetreDecoupe = lePrixMetreDecoupe;
    this->prixMetreCarreMatiere = lePrixMetreCarreMatiere;
}

void Commande::ajouterNouvelleFigure(Figure* laFigure)
{
    lesFigures.push_back(laFigure);
}

```

0,75 point

1 point

1,5 points

```

void Commande::cloturerCommande()
{
    this->commandeTerminee = true;
}

double Commande::getPrix()
{
    return 0;
}

/*****
main.cpp
*****/
#include <iostream>
#include <conio.h>
#include "Polygone.h"
#include "Cercle.h"
#include "Commande.h"

using namespace std ;           // espace de nommage standard

int main()
{
    // Testez la classe Cercle
    Cercle cercle((3.5, 5.5),0.5);
    cout << "Le périmètre du cercle est: " << cercle.getPerimetre() << "La surface du cercle est: " << cercle.getSurface() << endl;

    // Testez la classe Polygone avec la figure de test du sujet
    double Coordonnees[6][2]={ { 1 , 1 } , { 3 , 5 } , { 5 , 7 } , { 5 , 1 } , { 3 , 3 } , { 3 , 1 } };

    ...

    // Sapin de Noel et boules
    double CoordonneesSapin[15][2]={ { 2 , 2 } , { 5 , 4 } , { 3 , 4 } , { 5 , 6 } , { 4 , 6 } , { 6 , 8 } ,
    { 8 , 6 } , { 7 , 6 } ,
    { 9 , 4 } , { 7 , 4 } , { 10 , 2 } , { 6
    .5 , 2 } , { 6.5 , 1 } , { 5.5 , 1 } , { 5.5 , 2 } };

    double CoordonneesCentreCercles[6][2]={ { 2.5 , 3.5 } , { 3.5 , 5.5 } , { 4.5 , 7.5 } , { 7.5 , 7.5 }
    , { 8.5 , 5.5 } , { 9.5 , 3.5 } };
    int i;

    // Création du polygone sapin
    ...

    cout <<"superficie du sapin = " << ... << " " ;
    cout <<"Perimetre du sapin = " << ... << endl;

    // Création des 6 cercles
    ...
    cout <<"superficie du cercle " << i <<" = " <<... << " " ;
    cout <<"Perimetre du cercle " << i <<" = " << ...<< endl;
    ...

    // Création de la commande du Père Noel
    ...

    // Ajout des figures (le sapin et les 6 cercles) à la commande
    ...

    // Affichage du prix de cette commande
    cout <<"\n\nCout de la commande: " << ... <<" = " << ... <<" euros" << endl;

    _getch();           // on attend l'appui sur une touche
    return 0 ;          // fin du programme
}

```

Test de la classe
Cercle : 0,25 point

```
/******  
Point2D.cpp  
*****/  
// Cette classe n'est pas à modifier  
#include "Point2D.h"  
  
Point2D::Point2D(double x , double y)  
{  
    this->x = x;  
    this->y = y;  
}  
  
double Point2D::getX()  
{ return x ;}  
  
double Point2D::getY()  
{  
    return y;  
}  
  
void Point2D::setX(double newX)  
{  
    x = newX;  
}  
  
void Point2D::setY(double newY)  
{  
    y = newY;  
}  
  
/******  
Polygone.cpp  
*****/  
  
#include <math.h>  
#include "Polygone.h"  
  
Polygone::Polygone(void)  
{  
    this->estFerme = false;  
}  
  
double Polygone::distance(const Point2D* p1, const Point2D* p2)  
{  
}  
  
void Polygone::insereUnNouveauSommet( Point2D* leSommet, int position )  
{  
    lesSommets.push_back(leSommet);  
}  
  
void Polygone::fermeLePolygone()  
{  
    this->estFerme = true;  
}  
  
double Polygone::getPerimetre()  
{  
    .....  
}  
  
double Polygone::getSurface()  
{  
  
}
```

0,5 point