OOP small project for point of cyclinder

#include "stdafx.h"

#include <iostream>

#include "circle.h";

#include "Point.h";

int \_tmain()

{

Point p1,p2(2,4),p3(10);

return 0;

}

#include "stdafx.h"

#include "Point.h"

Point::Point(int a, int b)

{

x = a;

y = b;

}

#pragma once

class Point

{

public:

Point(int x = 0, int y = 0);

~Point(void);

private:

int x;

int y;

};

#include "stdafx.h"

#include "circle.h"

circle::~circle(void)

{

}

circle::circle(double r ,int a , int b): // calling constructor

centre(a,b)

{

radius = r;

}

#pragma once

#include "Point.h";

class circle

{

public:

circle(double r = 0,int x = 0,int y = 0);

circle(void);

~circle(void);

private:

Point centre;

double radius;

};

#include "stdafx.h"

#include "cyclinder.h"

cyclinder::cyclinder(double h,double r ,int a, int b)

:base(r,a,b)

{

height = h;

}

cyclinder::~cyclinder(void)

{

}

#pragma once

#include "circle.h";

class cyclinder

{

public:

cyclinder(double h = 1,double r = 1,int x = 0, int y = 0);

~cyclinder(void);

private:

circle base;

double height;

};

OOP further expansion of OOP testing of previous code

#include "stdafx.h"

#include <iostream>

#include "point.h"

#include "circle.h"

#include "Line.h"

int main()

{

Point p(30, 50);

p.print();

Circle c(12, 12, 12);

c.print();

return 0;

}

#include <iostream>

#include "Point.h"

using namespace std;

Point::Point(int a, int b) { setPoint(a, b); }

void Point::setPoint(int a, int b)

{

x = a;

y = b;

}

void Point::print()

{

cout << '[' << x << ", " << y << ']' << endl;

}

Point::~Point()

{

}

#pragma once

class Point

{

public:

   Point(int = 0, int = 0);

     ~Point();

void setPoint(int, int); // set coordinates

int getX() const { return x; } // get x coordinate

int getY() const { return y; } // get y coordinate

void print();

   protected:

      int  x, y;

};

#include <iostream>

using std::cout;

using std::endl;

using std::ios;

#include <iomanip>

using std::setiosflags;

using std::setprecision;

#include "circle.h"

Circle::Circle(double r, int a, int b)

: Point(a, b)

{

setRadius(r);

}

void Circle::setRadius(double r)

{

radius = (r >= 0 ? r : 0);

}

double Circle::getRadius() const { return radius; }

double Circle::area() const

{

return 3.14159 \* radius \* radius;

}

void Circle::print()

{

cout << "Center = " << x << " " << y

<< "; Radius = "

<< radius << endl;

cout << "The area is: "

<< setiosflags(ios::fixed)

<< setprecision(2) << area() << endl;

}

#include <iostream>

#include "point.h"

class Circle : public Point {

public:

Circle(double r = 0.0, int x = 0, int y = 0);

void setRadius(double); // set radius

double getRadius() const; // return radius

double area() const; // calculate area

void print();

protected:

double radius;

};

#include "stdafx.h"

#include "cyclinder.h"

cyclinder::cyclinder(double h, double r, int a, int b)

{

height = h;

radius = r;

x = a;

y = b;

}

cyclinder::~cyclinder()

{

}

#pragma once

class cyclinder

{

public:

cyclinder(double h = 1, double r = 1, int x = 0, int y = 0);

~cyclinder();

private:

double height;

double radius;

int x;

int y;

};

#include "stdafx.h"

#include "Line.h"

#include "Point.h"

Line::Line(int x, int y) :

Point(a, b)

{

}

void Point::setLine(int a, int b)

{

x = a;

y = b;

}

Line::~Line()

{

}

#pragma once

class Line

{

public:

Line(int a = 0, int b = 0);

~Line()

void setLine(int, int); // set coordinates

int getX() const { return x; } // get x coordinate

int getY() const { return y; } // get y coordinate

void printLine();

protected:

int  x, y;

};