МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙФЕДЕРАЦИИ МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСТИТЕТ)

ЛАБОРАТОРНАЯ РАБОТА №3по курсу объектно-ориентированное программирование I семестр, 2021/22уч. год

Студент: Попов Матвей Романович, группа М8О-208Б-20Преподаватель: Дорохов Евгений Павлович

**Задание**

Необходимо спроектировать и запрограммировать на языке C++ классы трех фигур, согласно варианту задания.

**Вариант 18**

Треугольник, квадрат, прямоугольник

**Описание программы**

Программа состоит из 10 файлов: main.cpp, figure.h, point.cpp, point.h, rectangle.cpp, rectangle.h, square.cpp, square.h, triangle.cpp, triangle.h, содержит реализации классов всех фигур, перегрузки операторов ввода/вывода и методы для вычисления площади каждой фигуры.

**Дневник отладки**

При отладке ошибок в выполнении программы не выявлено.

**Выводы**

Проделав лабораторную работу, познакомился с наследованием в ООП.

**Листинг**

main.cpp

#include <iostream>

#include "triangle.h"

#include "square.h"

#include "rectangle.h"

using namespace std;

int main()

{

bool s = 1;

while (s == 1)

{

cout << "Select the figure:\n";

cout << "1) Triangle\n";

cout << "2) Square\n";

cout << "3) Rectangle\n";

int f;

cin >> f;

if (f == 1)

{

cout << "Enter 3 points:\n";

Triangle t(cin);

t.Print(cout);

cout << "Triangle contains " << t.VertexesNumber() << " vertices.\n";

cout << "Area: " << t.Area() << endl;

}

if (f == 2)

{

cout << "Enter 4 points:\n";

Square s(cin);

s.Print(cout);

cout << "Square contains " << s.VertexesNumber() << " vertices.\n";

cout << "Area: " << s.Area() << endl;

}

if (f == 3)

{

cout << "Enter 4 points:\n";

Rectangle r(cin);

r.Print(cout);

cout << "Rectangle contains " << r.VertexesNumber() << " vertices.\n";

cout << "Area: " << r.Area() << endl;

}

cout << "Want to continue? (1 or 0)\n";

cin >> s;

}

cout << "Finished.\n";

return 0;

}

figure.h

#ifndef FIGURE\_H

#define FIGURE\_H

#include <cstddef>

#include "point.h"

using namespace std;

class Figure

{

public:

virtual ~Figure()

{};

virtual double Area() = 0;

virtual void Print(ostream& os) = 0;

virtual size\_t VertexesNumber() = 0;

};

#endif

point.cpp

#include "point.h"

#include <cmath>

Point::Point() : x\_(0.0), y\_(0.0) {}

Point::Point(double x, double y) : x\_(x), y\_(y) {}

Point::Point(std::istream &is) {

is >> x\_ >> y\_;

}

double Point::dist(Point& other) {

double dx = (other.x\_ - x\_);

double dy = (other.y\_ - y\_);

return std::sqrt(dx\*dx + dy\*dy);

}

std::istream& operator>>(std::istream& is, Point& p) {

is >> p.x\_ >> p.y\_;

return is;

}

std::ostream& operator<<(std::ostream& os, Point& p) {

os << "(" << p.x\_ << ", " << p.y\_ << ")";

return os;

}

point.h

#ifndef POINT\_H

#define POINT\_H

#include <iostream>

class Point {

public:

Point();

Point(std::istream &is);

Point(double x, double y);

double dist(Point& other);

friend std::istream& operator>>(std::istream& is, Point& p);

friend std::ostream& operator<<(std::ostream& os, Point& p);

private:

double x\_;

double y\_;

};

#endif // POINT\_H

rectangle.cpp

#include <cmath>

#include "rectangle.h"

using namespace std;

Rectangle::Rectangle(istream& is)

{

is >> p1 >> p2 >> p3 >> p4;

}

void Rectangle::Print(ostream& os)

{

os << "Rectangle: " << p1 << " " << p2 << " " << p3 << " " << p4 << endl;

}

double Rectangle::Area()

{

double a = p1.dist(p2);

double b = p1.dist(p3);

double c = p1.dist(p4);

double d1 = max(a, b);

double d2 = max(d1, c);

if (d2 == a)

return b \* c;

if (d2 == b)

return a \* c;

if (d2 == c)

return a \* b;

return 0.0; //How?

}

size\_t Rectangle::VertexesNumber()

{

return 4;

}

Rectangle::~Rectangle()

{

cout << "Done\n";

}

rectangle.h

#ifndef RECTANGLE\_H

#define RECTANGLE\_H

#include <iostream>

#include "figure.h"

using namespace std;

class Rectangle : public Figure

{

private:

Point p1, p2, p3, p4;

public:

Rectangle();

Rectangle(istream& is);

double Area();

void Print(ostream& os);

size\_t VertexesNumber();

virtual ~Rectangle();

};

#endif

square.h

#ifndef SQUARE\_H

#define SQUARE\_H

#include <iostream>

#include "figure.h"

using namespace std;

class Square : public Figure

{

private:

Point p1, p2, p3, p4;

public:

Square();

Square(istream& is);

double Area();

void Print(ostream& os);

size\_t VertexesNumber();

virtual ~Square();

};

#endif

square.cpp

#include <cmath>

#include "square.h"

using namespace std;

Square::Square(istream& is)

{

is >> p1 >> p2 >> p3 >> p4;

}

void Square::Print(ostream& os)

{

os << "Square: " << p1 << " " << p2 << " " << p3 << " " << p4 << endl;

}

double Square::Area()

{

double a = p1.dist(p2);

double b = p1.dist(p3);

double c = p1.dist(p4);

double d = a;

if (d > b)

d = b;

if (d > c)

d = c;

return d \* d;

}

size\_t Square::VertexesNumber()

{

return 4;

}

Square::~Square()

{

cout << "Done\n";

}

triangle.h

#ifndef TRIANGLE\_H

#define TRIANGLE\_H

#include <iostream>

#include "figure.h"

using namespace std;

class Triangle : public Figure

{

private:

Point p1, p2, p3;

public:

Triangle();

Triangle(istream& is);

double Area();

void Print(ostream& os);

size\_t VertexesNumber();

virtual ~Triangle();

};

#endif

triangle.cpp

#include <cmath>

#include "triangle.h"

using namespace std;

Triangle::Triangle(istream& is)

{

is >> p1 >> p2 >> p3;

}

void Triangle::Print(ostream& os)

{

os << "Triangle: " << p1 << " " << p2 << " " << p3 << endl;

}

double Triangle::Area()

{

double a = p1.dist(p2);

double b = p2.dist(p3);

double c = p3.dist(p1);

double p = (a + b + c)/2;

double s = sqrt(p \* (p - a) \* (p - b) \* (p - c));

return s;

}

size\_t Triangle::VertexesNumber()

{

return 3;

}

Triangle::~Triangle()

{

cout << "Done\n";

}