МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСТИТЕТ)

ЛАБОРАТОРНАЯ РАБОТА №1

по курсу объектно-ориентированное программирование I семестр, 2021/22 уч. год

Студент *Шандрюк Пётр Николаевич, группа М8О-208Б-20*

Преподаватель *Дорохов Евгений Павлович*

Условие

Задание: Вариант 25: Треугольник, квадрат, прямоугольник. Необходимо спроектировать и запрограммировать на языке C++ классы трех фигур, согласно варианту задания. Классы должны удовлетворять следующим правилам:

1. Должны быть названы также, как в вариантах задания и расположенны в раздельных файлах: отдельно заголовки (имя_класса_с_маленькой_буквы.h), отдельно описание методов (имя_класса_с_маленькой_буквы.cpp).
2. Иметь общий родительский класс Figure;
3. Содержать конструктор, принимающий координаты вершин фигуры из стандартного потока std::cin, расположенных через пробел. Пример: "0.0 0.0 1.0 0.0 1.0 1.0 0.0 1.0"
4. Содержать набор общих методов:
   - size_t VertexesNumber() - метод, возвращающий количество вершин фигуры;
   - double Area() - метод расчета площади фигуры;
   - void Print(std::ostream os) - метод печати типа фигуры и ее координат вершин в поток вывода os в формате: "Triangle: (0.0, 0.0) (1.0, 0.0) (1.0, 1.0)"с переводом строки в конце.

Описание программы

Исходный код лежит в 11 файлах:

1. main.cpp: основная программа, взаимодействие с пользователем посредством команд из меню
2. figure.h: описание абстрактного класса фигур
3. point.h: описание класса точки
4. triangle.h: описание класса треугольник, наследующегося от figures
5. square.h: описание класса квадрата, наследующегося от figures
6. rectangle.h: описание класса прямоугольника, наследующегося от figures
7. point.cpp: реализация класса точки
8. trinagle.cpp: реализация класса треугольника, наследующегося от figures
9. square.cpp: реализация класса квадрата, наследующегося от figures
10. rectangle.cpp: реализация класса прямоугольника, наследующегося от figures

Дневник отладки

Недочёты

Выводы

Я научился создавать и реализовывать классы в C++, а также познакомился с дружественными функциями, перегрузкой операторов и изучил основные понятия ООП.

Исходный код

figure.h

```
#ifndef FIGURE_H
#define FIGURE_H

#include <iostream>

class Figure {
public:

    virtual unsigned VertexesNumber() = 0;
    virtual void Print(std::ostream &os) = 0;
    virtual double Area() = 0;

    ~Figure() {};

};

#endif //FIGURE_H
```

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
```

point.cpp

```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;
}
```

triangle.h

```
#include <iostream>
#include "figure.h"
#include "point.h"

class Triangle : public Figure {
public:
    Triangle();

    virtual ~Triangle();

    void Print(std::ostream &os);
    double Area();
    unsigned VertexesNumber();

private:
    Point a; Point b; Point c;
};
```

triangle.cpp

```cpp
#include "triangle.h"
#include <cmath>


Triangle::Triangle() {
    std::cout << "Enter numbers for triangle: " << std::endl;
    std::cin >> a;
    std::cin >> b;
    std::cin >> c;
}



void Triangle::Print(std::ostream &os){
    os << "Triangle: ";
    os << a << " " << b << " " << c << std::endl;
}

double Triangle::Area(){
    double first = a.dist(b);
    double second = b.dist(c);
    double third = a.dist(c);
    double p = (first + second + third) / 2.0;
    return sqrt(p * (p - first) * (p - second) * (p - third));
}

unsigned Triangle::VertexesNumber(){
    return 3;
}


Triangle::~Triangle(){
    std::cout << "Triangle deleted!" << std::endl;
}
```

rectangle.h

```
#ifndef RECTANGLE_H
#define RECTANGLE_H

#include "figure.h"
#include "point.h"

class Rectangle : public Figure{
public:
    Rectangle();

    virtual ~Rectangle();

    void Print(std::ostream &os);
    double Area();
    unsigned VertexesNumber();

private:
    Point a; Point b; Point c; Point d;
};


#endif //RECTANGLE_H
```

rectangle.cpp

```cpp
#include "rectangle.h"

Rectangle::Rectangle() {
    std::cout << "Enter numbers for rectangle: " << std::endl;
    std::cin >> a;
    std::cin >> b;
    std::cin >> c;
    std::cin >> d;
}

void Rectangle::Print(std::ostream &os) {
    os << "Square: ";
    os << a << " " << b << " " << c << " " << d << std::endl;
}

double Rectangle::Area() {
    double first = a.dist(b);
    double second = b.dist(c);
    return first * second;
}

unsigned Rectangle::VertexesNumber() {
    return 4;
}

Rectangle::~Rectangle() {
    std::cout << "Rectangle deleted!" << std::endl;
}
```

square.h

```
#include "figure.h"
#include "point.h"

class Square : public Figure{
public:
    Square();

    virtual ~Square();

    void Print(std::ostream &os);
    double Area();
    unsigned VertexesNumber();

private:
    Point a; Point b; Point c; Point d;
};
```

square.cpp

```cpp
#include "square.h"
#include <cmath>

Square::Square() {
    std::cout << "Enter numbers for square: " << std::endl;
    std::cin >> a;
    std::cin >> b;
    std::cin >> c;
    std::cin >> d;
}

void Square::Print(std::ostream &os){
    os << "Square: ";
    os << a << " " << b << " " << c << " " << d << std::endl;
}

double Square::Area(){
    double first = a.dist(b);
    return pow(first, 2);
}

unsigned Square::VertexesNumber() {
    return 4;
}

Square::~Square() {
    std::cout << "Square deleted!" << std::endl;
}
```

main.cpp

```cpp
#include "triangle.h"
#include "square.h"
#include "rectangle.h"

int main(){

    Triangle f;
    f.Print(std::cout);
    std::cout << f.Area() << std::endl;

    Square f2;
    f2.Print(std::cout);
    std::cout << f2.Area() << std::endl;

    Rectangle f3;
    f3.Print(std::cout);
    std::cout << f3.Area() << std::endl;


    return 0;

}
```