Московский Авиационный Институт
(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Лабораторная работа
по курсу «ООП»

Тема:
Наследование,полиморфизм.

Студент:	Суворова С. А.
Группа:	М80-206Б-18
Преподаватель:	Журавлев А.А.
Вариант:	22
Оценка:	
Дата:	

Москва
2019

1.**Код на C++:**
point.h:

```
#ifndef D_POINT_H_
#define D_POINT_H_

#include <iostream>

struct point{
    double x,y;
};

std::istream& operator>>(std::istream& is, point& p);
std::ostream& operator<<(std::ostream& os,const point& p);
point operator+(point x1,point x2);
point& operator/= (point& x1, int number);

#endif
```

point.cpp:

```
#include <iostream>

#include "point.h"

std::istream& operator>> (std::istream& is, point& p){
    is >> p.x >>p.y;
    return is;
}

std::ostream& operator<< (std::ostream& os, const point& p){
    os << p.x << " " << p.y;
    return os;
}

point operator+(point x1,point x2){
    point x3;
    x3.x=x1.x+x2.x;
    x3.y=x1.y+x2.y;
    return x3;
}
point& operator/= (point& x1, int number){
    x1.x=x1.x/number;
    x1.y=x1.y/number;
    return x1;
}
```

figure.h:

```
#ifndef D_FIGURE_H_
#define D_FIGURE_H_

#include <iostream>

#include "point.h"

struct figure{
    virtual point center() const = 0;
    virtual void print(std::ostream &os) const = 0;
```

```
    virtual double square() const = 0;

    virtual ~figure() {};
};
#endif
```

five_angles.h:

```
#ifndef D_FIVE_ANGLES_H_
#define D_FIVE_ANGLES_H_

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

struct five_angles : figure{

    five_angles(std::istream &is);

    point center() const override;
    void print(std::ostream &os) const override;
    double square() const override;

private:
point one,two,three,four,five;

};

#endif
```

five_angles.cpp:

```
#include <iostream>

#include "five_angles.h"

five_angles::five_angles(std::istream &is){
    is >> one >> two >> three >> four >> five;
}

point five_angles::center() const {
    point p;
    p=one+two+three+four+five;
    p/=5;
    return p;
}

void five_angles::print(std::ostream &os) const {
    os << one << " " << two << " " << three << " " << four << " " << five
<<"\n";
}

double five_angles::square() const {
    double s=0;
    s=(one.x*two.y+two.x*three.y+three.x*four.y+four.x*five.y+five.x*one.y-
two.x*one.y-
            three.x*two.y-four.x*three.y-five.x*four.y-one.x*five.y)/2;
    if(s<0){
        return -s;
    }else {
        return s;
```

```
    }
}
```

six_angles.h:

```
#ifndef D_SIX_ANGLES_H_
#define D_SIX_ANGLES_H_

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


struct six_angles : figure{

    six_angles(std::istream &is);

    point center() const override;
    void print(std::ostream &os) const override;
    double square() const override;
private:
    point one,two,three,four,five,six;
};

#endif
```

six_angles.cpp:

```
#include <iostream>

#include "six_angles.h"

six_angles::six_angles(std::istream &is){
    is >> one >> two >> three >> four >> five >>six;
}

point six_angles::center() const {
    point p;
    p=one+two+three+four+five+six;
    p/=6;
    return p;
}

void six_angles::print(std::ostream &os) const {
    os << one << " " << two << " " << three << " " << four << " " << five <<
" " << six <<"\n";
}

double six_angles::square() const {
    double s=0;

s=(one.x*two.y+two.x*three.y+three.x*four.y+four.x*five.y+five.x*six.y+six.x*
one.y-two.x*one.y-
       three.x*two.y-four.x*three.y-five.x*four.y-six.x*five.y-
one.x*six.y)/2;
    if(s<0){
        return -s;
    }else {
        return s;
    }
}
```

eight_angles.h:

```
#ifndef D_EIGHT_ANGLES_H_
#define D_EIGHT_ANGLES_H_

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

struct eight_angles : figure{

    eight_angles(std::istream &is);

    point center() const override;
    void print(std::ostream &os) const override;
    double square() const override;
private:
    point one,two,three,four,five,six,seven,eight;
};

#endif
```

eight_angles.cpp:

```
#include <iostream>

#include "eight_angles.h"

eight_angles::eight_angles(std::istream &is){
    is >> one >> two >> three >> four >> five >>six >>seven >>eight;
}

point eight_angles::center() const {
    point p;
    p=one+two+three+four+five+six+seven+eight;
    p/=8;
    return p;
}

void eight_angles::print(std::ostream &os) const {
    os << one << " " << two << " " << three << " " << four << " " << five <<
" " << six << " " << seven
    << " " << eight<<"\n";
}

double eight_angles::square() const {
    double s=0;

s=(one.x*two.y+two.x*three.y+three.x*four.y+four.x*five.y+five.x*six.y+six.x*
seven.y+seven.x*eight.y+
            eight.x*one.y-two.x*one.y-three.x*two.y-four.x*three.y-
five.x*four.y-six.x*five.y-seven.x*six.y
            -eight.x*seven.y-one.x*eight.y)/2;
    if(s<0){
        return -s;
    }else {
        return s;
    }
}
```

main.cpp:

```
#include <iostream>
#include <stdio.h>
#include <vector>
#include <string.h>
#include "figure.h"
#include "five_angles.h"
#include "six_angles.h"
#include "eight_angles.h"

int main() {
    std::vector<figure*> figures;
    char a[14];
    char a1[10];
    while(scanf("%s",a)>0){
        figure *new_figure;
        if(strcmp(a,"five_angles")==0){
            new_figure=new five_angles(std::cin);
        } else if(strcmp(a,"six_angles")==0){
            new_figure=new six_angles(std::cin);
        }else if(strcmp(a,"eight_angles")==0){
            new_figure=new eight_angles(std::cin);
        }
        figures.push_back(new_figure);
        int g;
        g=scanf("%s",a1);
        if(g==0){
            break;
        }
        if(strcmp(a1,"print")==0){
            for (size_t i = 0; i < figures.size(); ++i) {
                figures[i]->print(std::cout);
            }
        }else if(strcmp(a1,"center")==0){
            for (size_t i = 0; i < figures.size(); ++i) {
                point p=figures[i]->center();
                std::cout << p << "\n";
            }
        }else if(strcmp(a1,"square")==0){
            for (size_t i = 0; i < figures.size(); ++i) {
                std::cout << figures[i]->square() << "\n";
            }
        }else if(strcmp(a1,"destroy")==0){
            int id;
            std::cin >> id;
            delete figures[id];
            figures.erase(figures.begin() + id);
        }
    }
    for (size_t i = 0; i < figures.size(); ++i) {
        delete figures[i];
    }
    return 0;
}
```

2. **Ссылка на репозиторий в GitHub:**
https://github.com/Suvorova-Sofya/oop_exercise_03

3.**Набор testcases:**
test1:

five_angles 1 2 2 -1 -3 -3 -4 0 -3 2 print
1 2 2 -1 -3 -3 -4 0 -3 2
six_angles 1 2 2 -1 1 -3 -3 -3 -4 0 -3 2 print
2 2 -1 1 -3 -3 -3 -4 0 -3 2
eight_angles 1 2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2 print
2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2

test2:
five_angles 1 2 2 -1 -3 -3 -4 0 -3 2 center
-1.4 0
six_angles 1 2 2 -1 1 -3 -3 -3 -4 0 -3 2 center
-1 -0.5
eight_angles 1 2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2 center
-1 -1.625
test3:
five_angles 1 2 2 -1 -3 -3 -4 0 -3 2 square
21
six_angles 1 2 2 -1 1 -3 -3 -3 -4 0 -3 2 square
25
eight_angles 1 2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2 square
31

4.**Результаты выполнения программы:**
test1:
five_angles 1 2 2 -1 -3 -3 -4 0 -3 2 print
1 2 2 -1 -3 -3 -4 0 -3 2
six_angles 1 2 2 -1 1 -3 -3 -3 -4 0 -3 2 print
2 2 -1 1 -3 -3 -3 -4 0 -3 2
eight_angles 1 2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2 print
2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2

test2:
five_angles 1 2 2 -1 -3 -3 -4 0 -3 2 center
-1.4 0
six_angles 1 2 2 -1 1 -3 -3 -3 -4 0 -3 2 center
-1 -0.5
eight_angles 1 2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2 center
-1 -1.625
test3:
five_angles 1 2 2 -1 -3 -3 -4 0 -3 2 square
21
six_angles 1 2 2 -1 1 -3 -3 -3 -4 0 -3 2 square
25
eight_angles 1 2 2 -1 1 -3 0 -5 -2 -5 -3 -3 -4 0 -3 2 square
31

5**. Объяснение результатов работы программы:**
Пользователь вводит название фигуры, координаты фигуры и действие, которое программа должна выполнить, относящееся к данной фигуре. В итоге программа

выполняет одно из 3-х действий: выводит все координаты фигуры, выводит центр масс фигуры или выводит площадь фигуры.

6.**Вывод:**
В данной программе показывается каким образом можно использовать такие возможности языка C++, как наследование и полиморфизм, которые упрощают создание новых классов, используя уже существующие.