

Московский Авиационный Институт  
(Национальный исследовательский Университет)

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

**Лабораторная работа  
по курсу «ООП»**

**Тема:  
Простые классы.**

Студент:	Николаев В.А.
Группа:	М80-206Б-18
Преподаватель:	Журавлев А.А.
Вариант:	14
Оценка:	
Дата:	

Москва  
2019

## 1. Код программы на языке C++:

### **point.h:**

```
#include <iostream>
```

```
struct point {  
    double x, y;  
    point (double a, double b) { x = a, y = b;};  
    point() = default;  
};
```

```
std::istream& operator >> (std::istream& npt, point& p );  
std::ostream& operator << (std::ostream& out, const point& p);
```

### **point.cpp:**

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

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

```
std::ostream& operator << (std::ostream& out, const point& p) {  
    return out << p.x << ' ' << p.y << "\n";  
}
```

### **pentagon.h:**

```
#pragma once
```

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

```
struct pentagon : figure{  
    point a1, a2, a3, a4, a5;  
    point center() const override;  
    void print(std::ostream& out) override;  
    double area() const override;  
    pentagon() = default;  
    pentagon(std::istream& is);  
};
```

### **pentagon.cpp:**

```
#include "pentagon.h"
```

```
point pentagon::center() const {  
    double x, y;  
    x = (a1.x + a2.x + a3.x + a4.x + a5.x) / 5;  
    y = (a1.y + a2.y + a3.y + a4.y + a5.y) / 5;  
    point p(x, y);  
    return p;  
}
```

```

void pentagon::print(std::ostream& out) {
    out << "Coordinates are:\n" << "{\n" << a1 << a2 << a3 << a4 << a5 << "}\n";
}
double pentagon::area() const {
    return (0.5) * std::abs((a1.x*a2.y + a2.x*a3.y + a3.x*a4.y + a4.x*a5.y + a5.x*a1.y)
- ( a1.y*a2.x + a2.y*a3.x + a3.y*a4.x + a4.y*a5.x + a5.y*a1.x ));
}
pentagon::pentagon(std::istream& is) {
    is >> a1 >> a2 >> a3 >> a4 >> a5;
}

```

### **hexagon.h:**

```
#pragma once
```

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

```

struct hexagon : figure
{
    point a1, a2, a3, a4, a5, a6;
    point center() const override;
    void print(std::ostream& out) override;
    double area() const override;
    hexagon() = default;
    hexagon(std::istream& is);
};

```

### **hexagon.cpp:**

```
#include "hexagon.h"
```

```

point hexagon::center() const {
    double x,y;
    x = (a1.x + a2.x + a3.x + a4.x + a5.x + a6.x) / 6;
    y = (a1.y + a2.y + a3.y + a4.y + a5.y + a6.y) / 6;
    point p(x,y);
    return p;
}
void hexagon::print(std::ostream& out) {
    out << "Coordinates are:\n{\n" << a1 << a2 << a3 << a4 << a5 << a6 << "}\n";
}

```

```

double hexagon::area() const {
    return 0.5 * std::abs((a1.x*a2.y + a2.x*a3.y + a3.x*a4.y + a4.x*a5.y + a5.x*a6.y +
a6.x*a1.y) - ( a1.y*a2.x + a2.y*a3.x + a3.y*a4.x + a4.y*a5.x + a5.y*a6.x + a6.y*a1.x
));
}

```

```
hexagon::hexagon(std::istream& is) {
```

```
    is >> a1 >> a2 >> a3 >> a4 >> a5 >> a6;
}
```

### **octagon.h:**

```
#pragma once
```

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

```
struct octagon : figure
{
    point a1, a2, a3, a4, a5, a6, a7, a8;
    point center() const override;
    void print(std::ostream& out) override;
    double area() const override;
    octagon() = default;
    octagon(std::istream& is);
};
```

### **octagon.cpp:**

```
#include "octagon.h"
```

```
point octagon::center() const {
    double x,y;
    x = (a1.x + a2.x + a3.x + a4.x + a5.x + a6.x + a7.x + a8.x) / 8;
    y = (a1.y + a2.y + a3.y + a4.y + a5.y + a6.y + a7.y + a8.y) / 8;
    point p(x,y);
    return p;
}

void octagon::print(std::ostream& out) {
    out << "Coordinates are:\n{"<< a1 << a2 << a3 << a4 << a5 << a6 << a7 << a8
    << "}";
}
```

```
double octagon::area() const {
    return 0.5 * std::abs((a1.x*a2.y + a2.x*a3.y + a3.x*a4.y + a4.x*a5.y + a5.x*a6.y +
a6.x*a7.y + a7.x*a8.y + a8.x*a1.y) - ( a1.y*a2.x + a2.y*a3.x + a3.y*a4.x + a4.y*a5.x
+ a5.y*a6.x + a6.y*a7.x + a7.y*a8.x + a8.y*a1.x ));
}
```

```
octagon::octagon(std::istream& is) {
    is >> a1 >> a2 >> a3 >> a4 >> a5 >> a6 >> a7 >> a8;
}
```

### **main.cpp:**

```
#include <iostream>
```

```
#include <vector>
```

```
#include "pentagon.h"
```

```
#include "hexagon.h"
```

```

#include "octagon.h"

int main()
{
    int i;
    std::vector<figure*> v;
    figure* f;
    i = 0;
    while (i != 5) {
        " std::cout << "\n 1) Add figure \n 2) Delete figure \n 3) Call all functions for whole
vector \n 4) Total area \n 5) Exit \n\n ";
        std::cin >> i;
        switch (i) {
            case 1:
                int j;
                "n std::cout << "\n 5) pentagon \n 6) hexagon \n 8) octagon \n\n";
                std::cin >> j;
                if (j == 5) {
                    f = new pentagon(std::cin);
                    v.push_back(f);
                }
                else if (j == 6) {
                    f = new hexagon(std::cin);
                    v.push_back(f);
                }
                else if (j == 8) {
                    f = new octagon(std::cin);
                    v.push_back(f);
                }
                break;
            case 2:
                int k;
                std::cout << "Enter index:\n";
                std::cin >> k;
                if (k > v.size()) {
                    break;
                }
                else {
                    v[k]->~figure();
                    v.erase(v.begin() + k);
                }
                break;
            case 3:
                for (auto elem : v) {
                    elem -> print(std::cout);
                }
                break;
        }
    }
}

```

```

        std::cout << elem -> area() << "\n";
        std::cout << elem -> center();
    }
    break;
case 4:
    double s = 0;
    for (auto elem : v) {
        s += elem -> area();
    }
    std::cout << " Total area:\n" << s;
    break;
}
}
}

```

**CmakeLists.txt:**

```

project(lab3)

add_executable(lab3
main.cpp
point.cpp
point.h
pentagon.cpp
pentagon.h
hexagon.cpp
hexagon.h
octagon.cpp
octagon.h
figure.h
)

```

## 2. Ссылка на репозиторий на GitHub.

[https://github.com/a1dv/oop\\_exercise\\_03.git](https://github.com/a1dv/oop_exercise_03.git)

## 3. Набор тестов.

test\_01.txt:

```

1
5
0 0 2 0 2 2 1 3 0 2
1
6
0 0 1 -1 2 0 2 2 1 3 0 2
1
8
0 0 1 -1 2 0 3 1 2 2 1 3 0 2 -1 1
3

```

5

test\_02.txt:

1

5

0 0.5 0.5 0 1 0 1 0.5 0.5 0.5

1

6

0 0.5 0.5 0 1 0 1.5 0.25 1 0.5 0.5 0.5

1

8

0 0.5 0.5 0 0.75 -0.5 1 0 1.5 0.25 1 0.5 0.75 0.75 0.5 0.5

3

5

test\_03.txt:

1

5

0 100 0 0 100 0 150 50 100 100

1

6

0 100 -50 50 0 0 100 0 150 50 100 100

1

8

0 100 -50 50 0 0 50 -50 100 0 150 50 50 100 100 50 150

3

5

#### **4. Результаты выполнения тестов.**

test\_01.result:

Coordinates are:

{

0 0

2 0

2 2

1 3

0 2

}

5

1 1.4

Coordinates are:

{

0 0

```
1 -1
2 0
2 2
1 3
0 2
}
6
1 1
Coordinates are:
{
0 0
1 -1
2 0
3 1
2 2
1 3
0 2
-1 1
}
8
1 1
```

```
test_02.result:
Coordinates are:
{
0 0.5
0.5 0
1 0
1 0.5
0.5 0.5
}
0.375
0.6 0.3
Coordinates are:
{
0 0.5
0.5 0
1 0
1.5 0.25
1 0.5
0.5 0.5
}
0.5
0.75 0.291667
Coordinates are:
```



```
{
0 0.5
0.5 0
0.75 -0.5
1 0
1.5 0.25
1 0.5
0.75 0.75
0.5 0.5
}
0.6875
0.75 0.25
```

test\_03.result:  
Coordinates are:

```
{
0 100
0 0
100 0
150 50
100 100
}
```

12500

70 50

Coordinates are:

```
{
0 100
-50 50
0 0
100 0
150 50
100 100
}
```

15000

50 50

Coordinates are:

```
{
0 100
-50 50
0 0
50 -50
100 0
150 50
50 100
100 50
```

```
}  
15000  
50 37.5
```

### **5. Объяснение результатов работы программы.**

- 1) Ввод осуществляется через поток стандартного ввода
- 2) Вывод осуществляется через поток стандартного вывода.
- 3) С помощью класса `point` реализуется запись в память координат в двумерном пространстве.
- 4) В классе `figure` реализованы виртуальные функции для работы со всеми фигурами.
- 5) В классе `pentagon` реализованы функции для работы с пятиугольниками
- 6) В классе `hexagon` реализованы функции для работы с шестиугольниками
- 7) В классе `octagon` реализованы функции для работы с восьмиугольниками

### **6. Вывод.**

Изучил виртуальные функции.