МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»

(ННГУ)

Институт информационных технологий, математики и механики

Направление подготовки: «Прикладная математика и информатика»

ОТЧЕТ

по лабораторной работе

СОЗДАНИЕ КЛАССОВ ГЕОМЕТРИЧЕСКИХ ФИГУР, ИСПОЛЬЗУЯ НАСЛЕДОВАНИЕ

Выполнил: студент группы

3822Б1ПМ2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_М. М. Хамков

Подпись

Проверил: к.ф.-м.н., доц.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. А. Черных

Подпись

Нижний Новгород  
2023

Содержание.

[Введение](#_Toc532934514) 3

[Постановка задачи](#_Toc532934515) 4

[Описание алгоритмов.](#_Toc532934532) 5

[*1.* Изменение размера вектора](#_Toc532934533) 5

[*2.* Добавление координаты в конец вектора](#_Toc532934534) 6

[*3.* Добавление координаты в начало вектора](#_Toc532934534) 7

[Описание программы.](#_Toc532934537) 8

[Результаты работы.](#_Toc532934537) 11

[Заключение.](#_Toc532934539) 12

[Литература.](#_Toc532934540) 13

[Приложение.](#_Toc532934541) 14

**Введение**

В современном программировании наследование является одним из основных принципов ООП и позволяет создавать иерархию классов, где дочерние классы наследуют свойства и методы от родительского класса. В данной лабораторной работе мы будем создавать классы геометрических фигур, используя наследование на языке C++. Это позволит нам оптимизировать код и упростить процесс создания новых классов, а также понять принципы работы наследования и его преимущества в разработке программного обеспечения.

**Постановка задачи**

В этой работе я с помощью средств языка программирования C++ реализую классы следующих геометрических фигур: точка, отрезок, круг, треугольник, прямоугольник и квадрат. Каждая фигура будет хранить в себе информацию, необходимую для её построения в декартовой системе координат. Также в каждой фигуре будут храниться её периметр (для точки - это 0, а для отрезка - его длина) и площадь (для точки и отрезка – это 0). Кроме этого будет создан класс «Image» для работы с вектором фигур по принципу холста.

**Описание алгоритмов**

1. *Подсчёт периметра и площади фигуры*

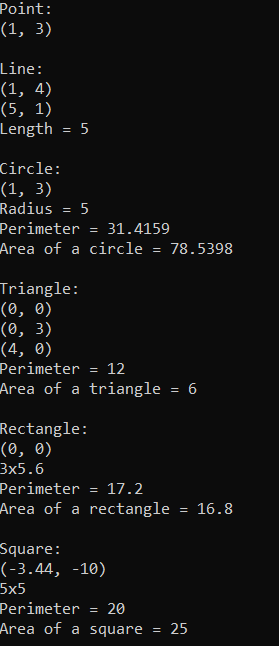
При создании объекта любого из реализованных классов геометрических фигур в него записывается информация об его периметре и площади. Эти значения вычисляются по стандартным математическим формулам вычисления периметра и площади круга, треугольника, прямоугольника и квадрата. Исключения составляют элементы классов точка и отрезок. У точки значения периметра и площади всегда равны 0. У отрезка площадь также равна 0, а его периметр равен длине этого отрезка.

**Описание программы**

Необходимо реализовать классы следующих геометрических фигур: точка, отрезок, круг, треугольник, прямоугольник и квадрат. Для них необходимо сделать конструкторы и метод вывода в консоль. Также необходимо сделать класс «Image» для работы с вектором фигур по принципу холста.

1. Point::Point(double x, double y)  
   Конструктор объекта класса «Point». Координаты точки равны x и y, а периметр и площадь равны 0.
2. void Point::show()  
   Метод вывода в консоль объекта класса «Point». Выводит координаты точки.
3. void Point::show\_type()  
   Метод вывода в консоль типа объекта класса «Point». Выводит строку «Point:» в консоль. Используется для удобства вывода в объекта класса «Image».
4. Line::Line(Point p1, Point p2)  
   Конструктор объекта класса «Line». Точки концов отрезка равны p1 и p2, периметр равен длине отрезка, а площадь равна 0.
5. void Line::show()  
   Метод вывода в консоль объекта класса «Line». Выводит координаты точек концов отрезка, а также длину отрезка.
6. void Line::show\_type()  
   Метод вывода в консоль типа объекта класса «Line». Выводит строку «Line:» в консоль. Используется для удобства вывода в объекта класса «Image».
7. Circle::Circle(Point p, double radius)  
   Конструктор объекта класса «Circle». Точка центра окружности равна p, а радиус равен radius. Периметр и площадь вычисляются по стандартным математическим формулам.
8. void Circle::show()  
   Метод вывода в консоль объекта класса «Circle». Выводит координаты точки центра, радиус, а также периметр и площадь окружности.
9. void Circle::show\_type()  
   Метод вывода в консоль типа объекта класса «Circle». Выводит строку «Circle:» в консоль. Используется для удобства вывода в объекта класса «Image».
10. Triangle::Triangle(Point p1, Point p2, Point p3)  
    Конструктор объекта класса «Triangle». Точки вершин треугольника равны p1, p2 и p3. Периметр и площадь вычисляются по стандартным математическим формулам.
11. void Triangle::show()  
    Метод вывода в консоль объекта класса «Triangle». Выводит координаты точек вершин, а также периметр и площадь треугольника.
12. void Triangle::show\_type()  
    Метод вывода в консоль типа объекта класса «Triangle». Выводит строку «Triangle:» в консоль. Используется для удобства вывода в объекта класса «Image».
13. Rectangle::Rectangle(Point p, double height, double width)  
    Конструктор объекта класса «Rectangle». Точка левой верхней вершины прямоугольника равна p, высота равна height, а ширина - width. Периметр и площадь вычисляются по стандартным математическим формулам.
14. void Rectangle::show()  
    Метод вывода в консоль объекта класса «Rectangle». Выводит координаты точки левой верхней вершины, высоту, ширину, а также периметр и площадь прямоугольника.
15. void Rectangle::show\_type()  
    Метод вывода в консоль типа объекта класса «Rectangle». Выводит строку «Rectangle:» в консоль. Используется для удобства вывода в объекта класса «Image».
16. Square::Square(Point p, double width)  
    Конструктор объекта класса «Square». Точка левой верхней вершины квадрата равна p, а ширина - width. Периметр и площадь вычисляются по стандартным математическим формулам.
17. void Square::show()  
    Метод вывода в консоль объекта класса «Square». Выводит координаты точки левой верхней вершины, ширину, а также периметр и площадь прямоугольника.
18. void Square::show\_type()  
    Метод вывода в консоль типа объекта класса «Square». Выводит строку «Square:» в консоль. Используется для удобства вывода в объекта класса «Image».
19. void Image::append(Figure\* const figure)  
    Добавляет figure в конец вектора фигур объекта класса «Image».
20. void Image::show()  
    Метод вывода в консоль объекта класса «Image». Выводит содержащиеся в его векторе фигур фигуры в формате:  
    «Тип объекта:»  
    «Данные объекта»

**Результаты работы**

1.  Результат запуска тестового файла:

**Заключение**

В ходе выполнения лабораторной работы были изучены принципы наследования в ООП. Эти принципы были применены на практике при создании классов геометрических фигур на языке C++. Также были изучены преимущества использования наследования в разработке программного обеспечения. В результате была получена более гибкая и удобная система классов, которая может быть легко расширена и модифицирована в будущем.

**Литература**

* Записи с лекций по предмету «Языки и методы программирования»;
* Записи с практических занятий по предмету «Языки и методы программирования».

**Приложения**

<https://github.com/Pocehome/Figures_Pocehome.git>