**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №2**

**по дисциплине «ООП»**

Тема: **Наследование**

|  |  |  |
| --- | --- | --- |
| Студент гр. 7303 |  | Алексо А.А. |
| Преподаватель |  | Размочаева Н.В. |

Санкт-Петербург

2019

**Цель работы**

Необходимо спроектировать систему классов для моделирования геометрических фигур (в соответствии с полученным индивидуальным заданием). Задание предполагает использование виртуальных функций в иерархии наследования, проектирование и использование абстрактного базового класса. Разработанные классы должны быть наследниками абстрактного класса Shape, содержащего методы для перемещения в указанные координаты, поворота на заданный угол, масштабирования на заданный коэффициент, установки и получения цвета, а также оператор вывода в поток. Необходимо так же обеспечить однозначную идентификацию каждого объекта.

Решение должно содержать:

* условие задания;
* UML диаграмму разработанных классов;
* текстовое обоснование проектных решений;
* реализацию классов на языке С++.

**Ход работы**

1. Были созданы вспомогательные структуры данных
   * 1. Struct Color – структура цвета, содержит три поля, которые хранят числа от 0 до 255 и характеризуют RGB.
     2. Class Point – класс точки, которое описывает координаты точки по оси X и Y.
2. Был создан абстрактный класс Shape, который содержит такие поля как: Цвет фигуры, точка центра фигуры, однозначный идентификатор,
   * вектор множества точке данной фигуры. И следующие методы:
     1. void setColor(Color color) –устанавливает цвет фигуры
     2. const Color getColor() const – выдает цвет фигуры
     3. const Point& getCenter() const –выдает центр фигуры
     4. void Moving(Point newCenter) – перемещает фигуру в заданную точку
     5. void Rotate(int angle,int direction) – поворачивает фигуру на определенный угол либо по часовой, либо против часовой стрелки
     6. virtual void Scale(double coefficient) – виртуальная функция масштабирования фигуры на заданный коэффициент, которая будет переопределятся в классах наследниках
     7. friend ostream& operator<<(ostream& out,const Shape& shape) –

переопределения метода вывода фигуры на экран

1. Был создан класс Square,который наследовался от класса Shape. Он имеет дополнительное поле – length ( длину стороны квадрата) и

переопределяетбазовыйметодScale(intangle,int direction),

масштабируя длину квадрата. А так же метод friend ostream& operator<<(ostream& out,const Square& square)

1. Был создан класс Ellipse, который наследовался от класса Shape. Он имеет два дополнительных поля: smallRadius и bigRadius и

переопределяетбазовыйметодScale(intangle,int direction),

масштабируя длину маленького радиуса и большого радиуса. А так же метод friend ostream& operator<<(ostream& out,const Ellipse& ellipse)

1. Был создан класс RegularPentagon, который наследовался от класса

Shape. Он имеет дополнительное поле – length ( длину стороны

правильного пятиугольника) и переопределяет базовый метод Scale(int angle,int direction), масштабируя длину пятиугольника. А так

же метод friend ostream& operator<<(ostream& out,const RegularPentagon & regulatPentagon).

**Обоснование**

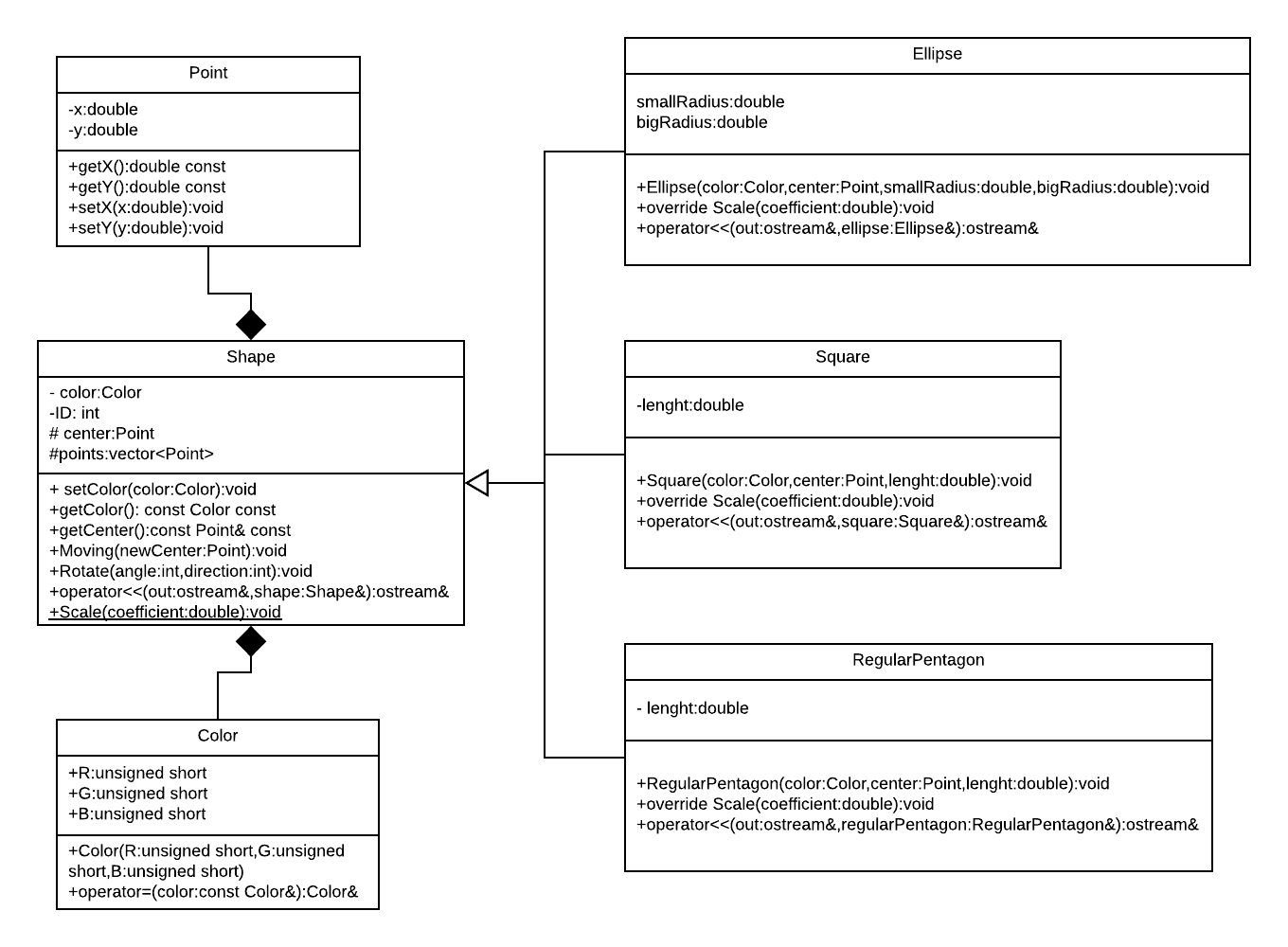
* + данной работе в абстрактном классе Shape только один метод Scale является виртуальным. Это происходит из-за того, что любую фигуру можно представить в виде множества точек. Виртуальными методами не являются методы Rotate и Moving, потому что:
* Rotate можно представить как умножение каждой точки на матрицу поворота и тем самым получение новых координат. В итоге изменяется только поле points(vector<Point> points),которое находится в абстрактном классе Shape. Вообщем изменяются только координаты самой фигуры.
  + - Moving это перемещение каждой точки фигуры на определенное расстояние. Для любой фигуры можно найти вектор, который будет показывать направление точки относительно центра. Следовательно мы можем просто перенести центр в заданную

точку и переместить каждую точку в соответствие с новым центром фигуры.

Переопределение в классах наследниках метода Scale требуется потому что, каждая фигура обладает своими индивидуальными свойствами. В данном случае, квадрат и правильный пятиугольник имеет длину стороны, а Эллипс имеет два радиуса(больший и меньший). Поэтому масштабирование должно изменять атрибуты, которые не являются полями абстрактного класса Shape.

Поля цвет, координаты центра и вектор точек фигуры являются общими для любой фигуры, поэтому они содержатся в абстрактном классе Shape. Конструкторы классов наследников всегда вызывают конструктор класса Shape и плюс для каждой отдельной фигуры, определенным образом вычисляются точки это фигуры. Так же, квадрат можно описать 4 точками и центром, правильный пятиугольник – 5 точками и центром, эллипс – 360 точек и центр. Для вычисления каждой точки эллипса используется соответствующая формула, которая по радиусам и углу определяет положение точки относительно центра.

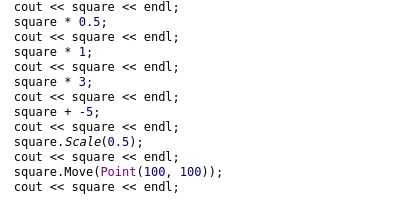
**UML диаграмма классов**



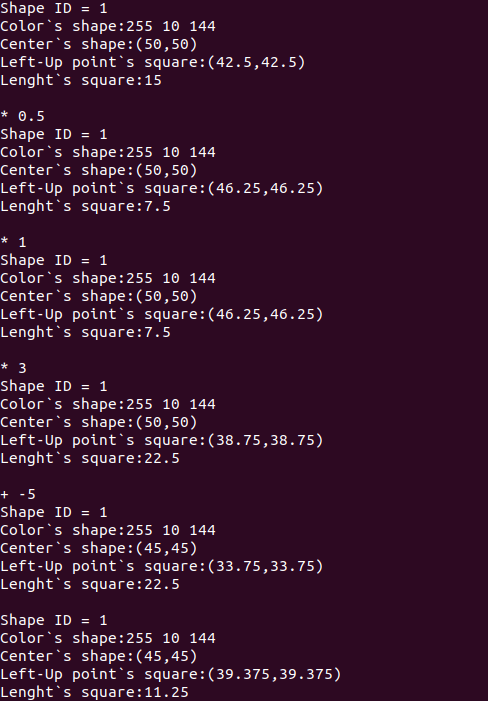
**Примеры работы программы**

* Пример (Квадрат):

1. Входные данные

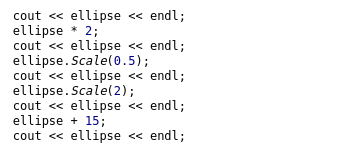
****

1. Выходные данные

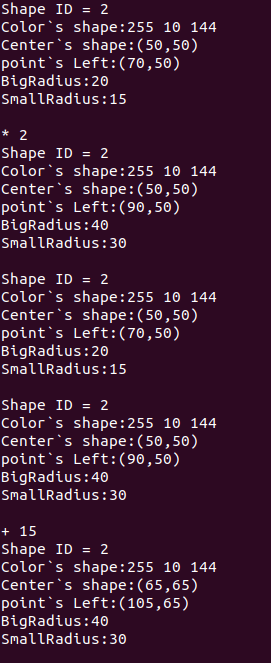
****

* Пример 2(Эллипс)

1. Входные данные

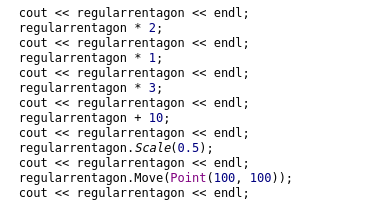


1. Выходные данные

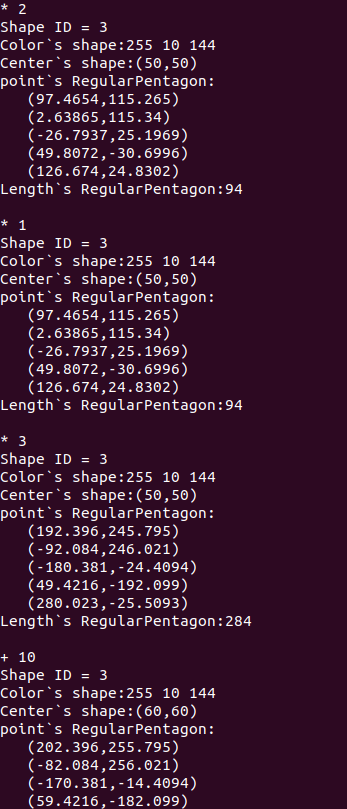


* Пример 3(Правильный пятиугольник)

o Входные данные

****

1. Выходные данные

****

**Вывод**

В ходе данной работы было изучено наследование классов в с++.

Была спроектирована система классов для моделирования геометрических фигур. Были использованы виртуальных функций в иерархии наследования. Разработанные классы являются наследниками абстрактного класса Shape, содержащего методы для перемещения в указанные координаты, поворота на заданный угол, масштабирования на заданный коэффициент, установки и получения цвета, а также оператор вывода в поток.