**Набережнов Денис Андреевич, 3530904/20001, ООП Фигуры, 6 вариант.**

1. **Общая постановка задачи**

1. Создать файл base-types.h, содержащий определения следующих структур:

• point\_t, представляющую собой точку на плоскости. Координаты должны храниться в полях x и y.

• rectangle\_t, описывающую прямоугольник шириной width и высотой height с центром в точке pos.

2. Создать файл shape.h, содержащий определение абстрактного класса Shape. Этот класс должен предоставлять следующие методы:

• getArea вычисление площади

• getFrameRect получение ограничивающего прямоугольника для фигуры (см. типы из предыдущего пункта), стороны ограничивающего прямоугольника всегда параллельны осям

• move перемещение центра фигуры, реализовать в двух вариантах:

* перемещение в конкретную точку
* смещение по осям абсцисс и ординат

• scale изотропное масштабирование фигуры относительно её центра с указанным коэффициентом

• getName получение названия фигуры (RECTANGLE, CIRCLE, COMPLEX и т.д.), метод использовать при выводе

• clone создание копии фигуры в динамической памяти и получение указателя на эту копию

3. Реализовать класс Rectangle, производный от Shape, в файлах rectangle.h и rectangle.cpp, соответственно (см. раздел «Фигуры»).

4. Реализовать фигуру, указанную преподавателем (см. раздел «Фигуры»).

5. Реализовать класс CompositeShape, производный от Shape (см. раздел «Фигуры»), представляющий собой составную фигуру и хранящий массив в динамической памяти из произвольных фигур, реализованных в вашей программе. Использование умных указателей и контейнеров стандартной библиотеки недопустимо. Объекты класса должны корректно копироваться и перемещаться. При добавлении базовой (прямоугольник, круг и т.п.) фигуры в составную, добавляться должна копия базовой фигуры.

6. Перегрузить оператор вывода << для класса Shape. Вывод должен содержать в одну строчку: название фигуры, её площадь, координаты левого нижнего угла и правого верхнего угла ограничивающего прямоугольника. Элементы при выводе должны округляться до одного знака после запятой в соответствии с правилами математики.

7. Перегрузить оператор сравнения < для класса Shape. Сравнение проводить по площади фигуры.

8. Реализовать функцию, сортирующую массив указателей на фигуры в порядке неубывания их площадей.

9. Написать программу, которая считывает описания фигур и команд из файла (см. раздел «Требования») и после окончания ввода фигур, а также после выполнения каждой команды выводит в стандартный вывод на отдельных строках информацию о каждой фигуре в порядке неубывания их площадей. Вывод для составной фигуры должен быть таким же, как и для всех остальных: название, площадь, координаты углов ограничивающего прямоугольника.

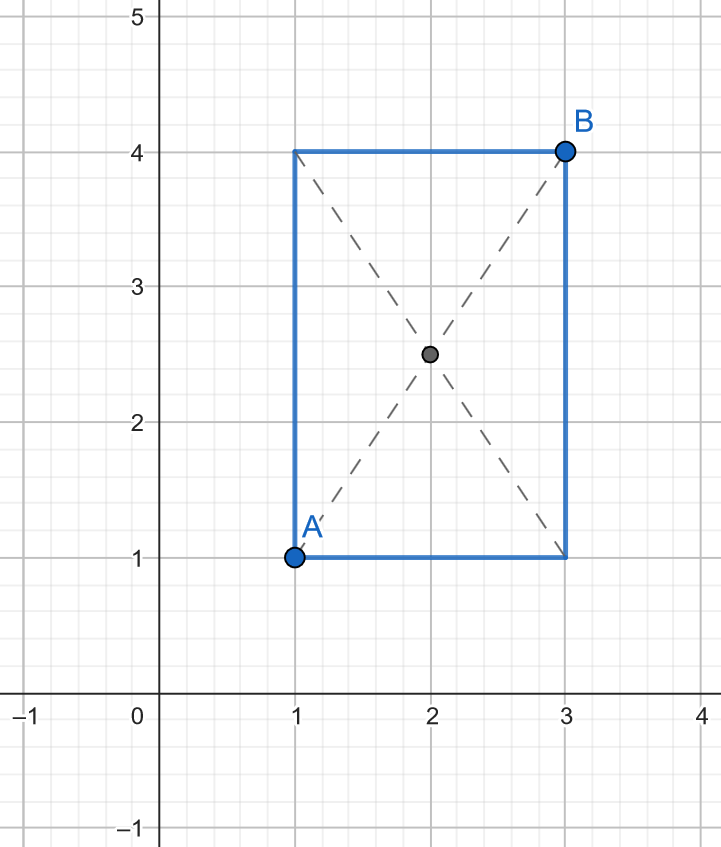
10. Отдельно продемонстрировать работу всех методов копирования и перемещения составной фигуры.

**Фигуры**

Каждая фигура описывается своим набором параметров. Отсутствие самопересечений, выпуклость фигур и корректное количество (и только количество) параметров гарантируется, если не сказано иного:

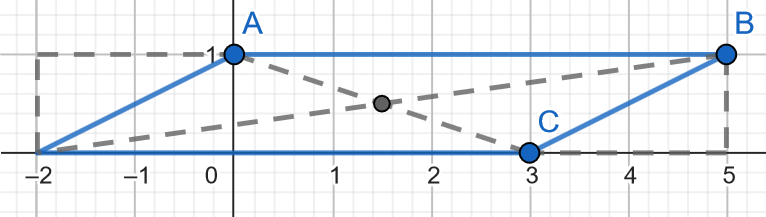
**Прямоугольник**. Описывается парой координат своих углов: левым нижним и правым верхним. Считается, что стороны прямоугольника параллельны осям координат. Центром фигуры считается точка пересечения диагоналей

RECTANGLE 1.0 1.0 3.0 4.0



**Параллелограмм**. Описывается тремя вершинами, составляющими треугольник, одна из сторон которого является диагональю параллелограмма, а две другие сторонами параллелограмма. Стороны параллелограмма формируются первой и последней парой вершин. При этом одна из сторон должна быть параллельна оси абсцисс. Центром фигуры считается точка пересечения диагоналей

PARALLELOGRAM 0.0 1.0 5.0 1.0 3.0 0.0



**Составная фигура**. Состоит из нескольких простых (не составных) фигур. Описывается на нескольких строках, начинаясь и завершаясь специальными командами (всегда находятся на отдельных строках). В первой строке указывается параметр – максимально возможное количество простых фигур. Центром фигуры считается центр ограничивающего прямоугольника

COMPLEX 5

CIRCLE 2.0 3.0 15.0

RECTANGLE -5.0 -3.0 3.0 4.0

CIRCLE 1.0 1.0 10.0

SQUARE -1.5 -1.5 10.0

SQUARE 0.0 0.0 5.0

COMPLEXEND

1. **Детальные требования, тест план**

**Таблица с детальными требованиями и тест планом.**

|  |  |  |  |
| --- | --- | --- | --- |
| Требование | Детальные требования | Данные | Ожидаемый результат |
| 1. Данные фигур и команд должны быть заданы корректно | | | |
| 1.1 Данные фигур – числа с плавающей точкой. | 1.1 Если условие не выполняется сообщение “Wrong parameters” | RECTANGLE 0 0 E 1 | Сообщение: “Wrong parameters”. |
| 1.2 Данные фигур образуют существующую фигуру. | 1.2 Если условие не выполняется сообщение “Wrong figure” | RECTANGLE 3 3 1 1 | Сообщение: “Wrong figure”. |
| 1.3 Данные команд – числа с плавающей точкой. | 1.3 Если условие не выполняется сообщение “Wrong parameters” | MOVE 1 B | Сообщение: “Wrong parameters”. |
| 2. Количество фигур в составной фигуре меньше или равно заданному | | | |
| 2.1 Количество фигур в составной фигуре меньше или равно заданному. | 2.1 Если условие не выполняется, сообщение “No place” | COMPLEX 1  RECTANGLE 0 0 4 4  PARALLELOGRAM 0 1 5 1 3 0  COMPLEXEND | Сообщение: “No place” |
| 3. Строки не соответствующие реализованным фигурам или командам пропускаются | | | |
| 3.1 Пустые строки пропускаются. | 3.1 Если условие выполняется строка игнорируется | “” | Игнорирование строки |
| 3.2 Строки с не реализованными в программе фигурами пропускаются. | 3.2 Если условие выполняется строка игнорируется | RHOMBUS 2 2 4 4 | Игнорирование строки |
| 4. Файл должен открываться | | | |
| 4.1 Файл открывается корректно | 4.1 Если условие не выполняется, сообщение “File doesn’t exist” | in.open(“out.txt”) | Сообщение “File doesn’t exist” |