1 Геометрические фигуры

Все числовые данные в этой работе должны быть представлены значениями с плавающей запятой.

1. Создать файл base-types.hpp, 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.hpp, 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, 2 варианта: в конкретную точку и в виде смещений по осям абсцисс и ординат

3. Реализовать классы Rectangle и Circle в файлах rectangle.hpp, rectangle.cpp, circle.hpp и circle.cpp соответственно.

4. Продемонстрировать правильную работу классов простой программой. Демонстрация должна включать полиморфное применение классов.

2 Масштабирование фигур

1. *В виде исключения! Все дальнейшие работы должны следовать правилам оформления работ и не содержать скопированного кода, за исключением набора тестов.* Скопируйте исходные тексты задания 1.

2. Перенесите классы фигур в отдельное пространство имен. Имя этого пространства должно быть выбрано совпадающим с фамилией студента в нижнем регистре (соответственно, оно совпадает с частью имени каталога с работами до точки), например, для Петрова Ивана каталог будет называться petrov.ivan, соответственно, имя пространства имен — petrov. Это пространство имен должно сохраняться для всех оставшихся работ в этом семестре.

3. Добавить в класс фигуры метод scale(), осуществляющий изотропное масштабирование фигуры относительно ее центра с указанным коэффициентом.

   Если в первой работе был реализован треугольник, масштабирование необходимо реализовать и для него.

4. Написать тесты, проверяющие:

   · неизменность ширины и высоты, а также площади фигуры при перемещениях;

   · квадратичное изменение площади фигуры при масштабировании;

   · наличие и обработку некорректных параметров в функциях;

   Для написания тестов необходимо создать файл test-main.cpp, в котором реализовать тесты.

Созданная ранее демонстрационная программа должна быть доработана для демонстрации новых возможностей.

3 Составные фигуры

Расширить реализацию работы 2 путем добавления класса CompositeShape, хранящего список из произвольных фигур внутри массива. В этой работе *не допускается* использование стандартных контейнеров, необходимо самостоятельно реализовать хранение множества фигур на базе динамического массива. Класс должен быть размещен в файлах composite-shape.hpp и composite-shape.cpp.

Написать набор тестов, проверяющий корректную работу созданного класса. Необходимо помнить, что может потребоваться реализация дополнительных специальных методов в классе для обеспечения корректной работы.

Для CompositeShape масштабирование и перемещение работают относительно центра этого объекта, за который принимается центр ограничивающего прямоугольника.

Созданная ранее демонстрационная программа должна быть доработана для демонстрации новых возможностей.

4 Обработка фигур

Расширить реализацию работы 3:

1. Добавить поддержку вращения фигур на заданный угол в градусах, положительное направление — против часовой стрелки, центр поворота совпадает с центром фигуры. Необходимо помнить, что ограничивающий прямоугольник сохраняет параллельность сторон осям координат, а поворот CompositeShape также содержит перемещение элементов. Метод должен называться rotate().

   Тем, кто реализовывал в первой работе поддержку треугольников, необходимо поддержать вращение треугольников тоже.

2. Написать разбиение списка фигур (Rectangle, Circle, CompositeShape и, если был реализован, Triangle), представленного в виде единой фигуры, по слоям в порядке добавления:

   · фигуры, которые не перекрываются ограничивающими прямоугольниками, находятся на одном слое;

   · фигура, перекрывающаяся с другой, находится на следующем слое если она указана в списке после той фигуры, которую она перекрывает;

   · составные фигуры, добавленные в фигуру, подвергнутую разбиению, необходимо рассматривать как единое целое и обрабатывать ее ограничивающий прямоугольник, а не ее составляющие.

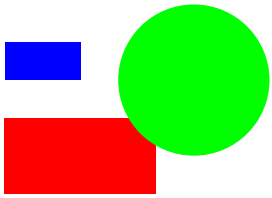
   Например, фигуры указаны в следующем порядке:

   · красный прямоугольник

   · зеленый круг

   · синий прямоугольник

   Соответственно, они рисуются друг за другом:



   Разбиение по слоям выглядит так:

   первый слой 2 прямоугольника, так как они не перекрываются, а круг закрывает красный прямоугольник

   второй слой зеленый круг, так как он указан после красного прямоугольника и перекрывается с ним.

   Результаты разбиения представить в виде матрицы, где строки представляют собой слои, в которых указаны фигуры. Матрицу реализовать самостоятельно на базе динамической памяти, данные должны храниться в 1 блоке (недопустимо создавать массив указателей на слои). Количество строк соответствует количеству слоев, количество столбцов — максимальному количеству фигур в одном слое.

3. Реализовать тесты, демонстрирующие корректную работу функции разбиения.

Созданная ранее демонстрационная программа должна быть доработана для демонстрации новых возможностей.