CENG 242

Hw #5

Spring 2006/2007

(Due: May 14th, 2007 Monday 23:59)

In this homework, you will write a C++ code for a simple drawing tool. You will have a base class called **Shape** and four derived classes **Rectangle**, **Ellipse**, **Triangle** and **ShapeGroup**. ShapeGroup is group of Shape's.

Shape class has these methods:

- virtual void move(int x, int y) = 0;

  This method moves the shape in x and y directions without deforming the shape.

- virtual void scale(int ratio) = 0;

  Assume that all the shapes have rectangular bounding boxes (the minimum sized rectangle whose sides are assumed to be parallel to x and y axis and covering the whole shape). Assume that size of bounding box is *w* x *h*, the new size should be *ratio*w* x *ratio*h* after scale operation, and the center of bounding box should be same as before.

- virtual void print() = 0;

  Print the information about shape. The format will be given below.

All the derived classes should implement these virtual methods of Shape.

Rectangle has this constructor:

- Rectangle(Point p1, Point p2);

  where Point is a structure:

  ```
  struct Point {
      int x;
      int y;
  };
  ```

  These points will represent the lower left and upper right corners of the rectangle. Sides of rectangles are assumed to parallel to x and y axis.

Triangle’s constructor is:

- Triangle(Point p1, Point p2, Point p3);

Ellipse class has the constructor:

- Ellipse(Point p1, Point p2);

  These points will represent the lower left and upper right corners of the bounding box of the ellipse. Major axis of the ellipse is assumed be parallel to x-axis.

ShapeGroup have these methods:

- bool addShape(Rectangle r1);
  bool addShape(Triangle t1);
  bool addShape(Ellipse e1);

  These methods add the given shape to shape group if shape group is empty or the given shape has a non-empty intersection with at least one of the shapes in the shape group. If it does not intersect with any of the shapes in the shape group, do not add it. The return value is true if is added, false otherwise.

- bool merge(ShapeGroup s1);

  This method adds all the shapes of s1 to the shape group, if there exists two intersecting shapes, one from each shape group. If there is no intersection, do not merge. The return value is true if you merged, false otherwise.

- int removeShapes(point p1);

  Detect which shapes of the shape group include the point and remove these shapes from the group. A shape includes a point if the point is on the sides of the shape or inside the shape. The methods should return the number of removed shapes.

print method has the format as follows:

If it is rectangle:

Rectangle *x1 y1 x2 y2*

where (*x1*, *y1*) is lower left corner and (*x2*, *y2*) is upper right corner. There should be newline character at the end.

If it is triangle:

Triangle *x1 y1 x2 y2 x3 y3*

where the points ((*x1*, *y1*) (*x2*, *y2*) (*x3*, *y3*)) are in the order they are given while constructing the shape (values may be different from initial values because of move and scale operations). There should be newline character at the end.

If it is ellipse:

Ellipse *x1 y1 x2 y2*

where (*x1*, *y1*) is left focus and (*x2*, *y2*) is the right focus. There should be newline character at the end.

If it is shape group, print the shapes in the order they are added. Merge operation adds the shapes of second group after shapes of first group.

Defining virtual methods like isInside(Point) and doesIntersect(…) in Shape class will be helpful for you. But you are free to define or not.

Specifications:

- For detailed information, you can take a look at wikipedia.
- You can define more public or private methods, or other classes. But the names and types are strict for the methods and classes given.
- You can assume that constructors will be called with reasonable parameters.
- You will have two files. One is **hw5.h** for your class declarations, structures etc.; the other is **hw5.cpp** for your definitions. You should **not** write **main** function in these files.
- You will submit a single tar file **hw5.tar** including *hw5.h* and *hw5.cpp*. You can tar your files with the command "*tar cvf hw5.tar hw5.cpp hw5.h*". Do not send me in other formats like ".tar.gz", ".rar", ".zip" etc.
- You will submit your codes through cow system. Specifications (file name, method names, class name, types etc.) are strict. Breaking any of them will cost you getting a 0 since black box method is used.