Chun-Wei Chen

Assignment #2 Written Report

2012/01/17

I started planning the program by thinking about what attributes is shared among all Shapes, and then I realized that I should specify the location of the upper left corner of the bounding box when drawing Shapes in Java. Therefore, I decided to use the location of upper left corner, upperLeftX and upperLeftY, as instance variables in AbstractShape class. I also made the height of bounding box as a instance variable in AbstractShape class. Afterwards, I found that I should be able to get the Color of the Shape and keep track on whether the Shape is selected. I added Color color and boolean selected as instance variables in AbstractShape class. I used the location of the center and radius as instance variables in Circle class, and I just used the location of the apex as instance variables because I could use the same instance variable height in AbstractShape class. Since a Tee is constructed by the location of the upper left corner and height, I didn’t add any new instance variable in Tee class.

I overrode all the methods implements from interface Shape except isOn and toString in AbstractShape because isOn and toString may vary in different Shapes. I added getUpperLeftX and getUpperLeftY in AbstractShape in order to let every shape to keep track on the bounding box. I overrode moveTo and toString in Circle and Delta since these two Shapes required different point location to construct, but I didn’t override moveTo in Tee because it just used the same instance variables in AbstractShape to construct.

The most not-straightfoward method was isOn. Every Shape has a specific way to determine whether the point is on the Shape. I used distance formula to find out if the distance between the point passed in and the center is less or equal to radius in Circle, and I used two rectangles boundary to determine in Tee. It took me some time to figure out how to write isOn in Delta. After a while, I figured out the equation of the lines of the boundary of the Delta; consequently, I made use of these equations in Delta’s isOn method.

I used main methods to test every concrete Shape class. I also let the program print out the location of the upper left corner of the Shapes. Nevertheless, I found that the location of the upper left corner didn’t move after I used shiftBy methods to the Shapes, then I found that I should also override shiftBy method in every concrete Shape whose construct point is not the upper left corner. Thus I solved this problem. After I finished the DrawingBoard class and tried to use the method to find the topmost Shape which contains the specified point, it didn’t work. I inserted some System.out.println statements in that method to figure out where the method failed. I tried a lot of times. Finally, I realized that I used hasPrevious to determine whether the ArrayList has previous item at the beginning of the ArrayList, so it didn’t work. I solved that problem by using the Iterator to go through the ArrayList from the first item to the last item, and then used hasPrevious and previous to go backwards. Those are the main bugs I encountered.

This assignment really helps me to understand how to make use of inheritance composition and the interface. Inheritance composition really reduces a lot of redundancy.

It also helps me comprehend how to make use of keyword “abstract” and “super.” I also learned how to go through the ArrayList in the reverse direction and how to return the copy of ArrayList. It seemed easy when just reading the textbook and listening the lecture, but it was actually not that simple when it became the assignment. It was not easy, but I learned a lot from it.