

CPSC 233: Assignment 2(*Due February 15 at 4 PM*)

New learning concepts: Creating an Object-Oriented program. Aside from ‘main’ you cannot implement other static methods. (Don’t worry if you don’t know the significance of the word ‘static’ yet, an explanation will come later in the term). Also you must implement the code in two Java classes (and not just a single class as you did with the previous assignment).

Problem Statement:

Write a program that draws simple shapes (rectangle and two different kinds of triangles) using ASCII graphics. As was the case with the previous assignment the information about the shapes and their characteristics will come from an input file. Each line in the input file will provide the information to draw one shape; and will come in the following format:

<Shape type><Element><dimensions>

Shape type: A two-character string that is a code for the type of shape to be drawn (“RE” = rectangle, “LT = right angle triangle with the corner on the top left, “RT” = right angle triangle with the corner on the bottom right)

Element: An ASCII character that acts as the drawing element repeated over and over to produce the shape.

Dimensions: for rectangles it will include two numbers (height and width), for the two triangles it will be a single number that specifies the size – as shown in Figure 1B triangles are symmetric so the height and width are the same. You can assume width and height dimensions will be single digit, *i.e.*, non-negative numbers from 1 – 9 for the rectangle, while the size of the triangle will range from 2 – 9.

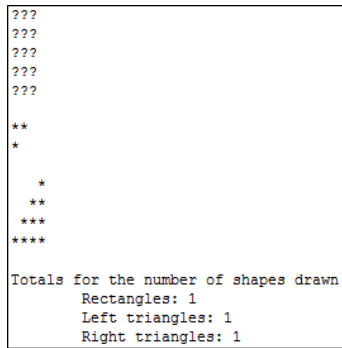
The example input file shown in Figure 1A will produce three shapes. The first shape is a 3 by 5 rectangle filled with the character ‘?’. The second shape is a left triangle of size 2, filled with the character ‘*’. The third shape is a right triangle of size 4, also filled with the character ‘*’. The corresponding sample output is provided in Figure 1B, in which each shape is separated by an empty line. Finally the program should display a tally of the number of each type of shape that was drawn.

```
RE?53
LT*2
RT*4
```

Figure 1A: Input file indicating drawing information for the shapes

```

???
```



```

???
```

```

**
*
  *
  **
 ***
****

Totals for the number of shapes drawn
  Rectangles: 1
  Left triangles: 1
  Right triangles: 1
```

Figure 1B: Corresponding output drawn by the program according to the information in the input file.

Programming Requirements:

- Like the previous assignment the user will enter the name of the input file in the form of a command line argument (e.g., “java Driver shapes.txt”).
- Unlike the previous assignment, your program should consist of two classes: ‘*Driver.java*’ which contains the **main ()** method, and ‘*Draw.java*’ which implements the methods that do the actual drawing.
- The **main ()** method reads from a file whose name is specified as the command-line argument. The **main ()** method then create an instance of the ‘**Draw**’ class. The **main ()** method should also be responsible for parsing the line read from file into the three parts (shape type, element, dimension(s)). Finally this method should invoke the appropriate methods from the ‘**Draw**’ class (passing the required parameters) to draw the shapes, as exemplified by Figure 1B. The Draw class will implement 3 drawing methods that accept parameters as specified below.

```
public void drawRectangle(char appearance, int width, int height)
```

```
public void drawLeftTriangle(char appearance, int size)
```

```
public void drawRightTriangle(char appearance, int size)
```

- Each method should draw the corresponding shape using a pair of nested loops.
- Finally the Draw class should have three numeric attributes, one for each supported shape that counts how many of each shape were drawn throughout the program execution. Although mutator/set methods aren’t needed because these counts shouldn’t be updated outside this class’ methods, some sort of accessor/get or display method will be needed so that the total number of each type shape can be displayed after drawing has been finished.

External libraries/methods that can be used (you don't write yourself)

You can use pre-created code for console input and output (e.g., 'print', class 'Scanner'), classes for file input and to methods convert from String to numeric types.

Important reminders

- **Assignment Guidelines:** When working on your assignments, please adhere strictly to the generic assignment submission guidelines (which apply to all assignments) as well as the ones listed in this assignment.
- **Handing in your assignment:** Use the UNIX 'submit' program. Make sure you submit the source code files (*.java) for both classes.
- **Collaboration:** Assignments must reflect individual work, group work is not allowed in this class nor can you copy the work of others.