Drawing ASCII Art

Introduction:

In this assignment, you are to write Java code that produce an ASCII art picture that varies in size based upon input given by the user.

Learning Outcomes:

After completing this assignment, you will have experience with:

1. Writing loops that repeat some computation a specified number of times.
2. Writing conditional statements that alter the execution of your program based upon various criteria.
3. Writing methods that accept parameters to eliminate or reduce redundant code in your program.

Resources:

Please download the supplied zip file available with this specification. Extract the files onto your computer. The extracted files contain an Android Studio project. Start Android Studio and import the project as demonstrated in an earlier module. All your work for this assignment will be in the file `Logic.java`. Open this file in the IDE and look for the comment:

```
// TODO -- add your code here
```

As you do your work, be sure to place all your code in the `process()` method. If/when you add helper methods, be sure to place them in the `Logic` class.

In order for evaluators to do their job they will need to download and compile your code. Your code should therefore be importable into Android Studio, should compile without error, and should then run correctly on an emulated Android device. You should just upload the file `Logic.java` for evaluation by the assessors.

Tasks:

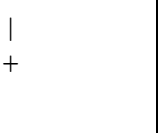
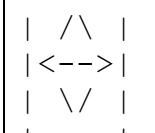
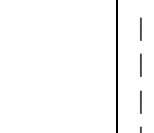
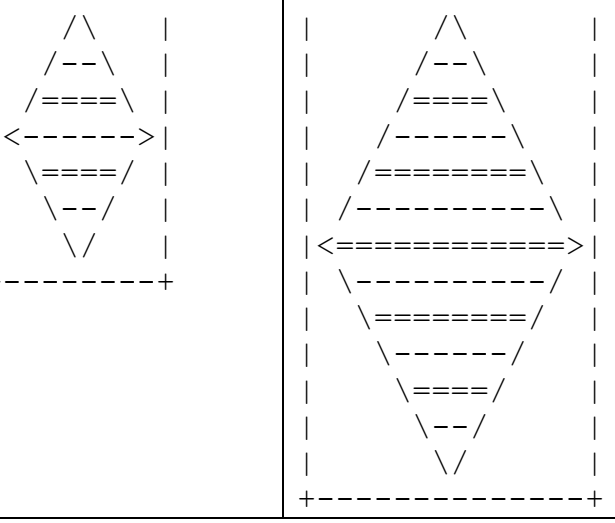
In this assignment, you are to write a program that will print an ASCII art drawing of a diamond in a picture frame that will be displayed on the screen of your Android device in a text box. You should **exactly** reproduce the format of the output samples below. This includes having identical characters and spacing. One way to write a Java program to draw this figure would be to write a

`println` statement that prints each line of the figure. However, this solution would not receive full credit. A major part of this assignment is showing that you understand `for` loops and `if` statements.

Another significant component of this assignment is the task of generalizing the program using a variable to adjust the size of the figure. When the program is run, the user will select the size of the figure they want produced. You are provided with code that accesses that value and passes it to the method named `process()` in the integer parameter named `size`. Your code should be written in terms of the parameter `size` so that it produces the correctly sized figure.

Here are four sample outputs of the program, with diamonds of size 1, 2, 4, and 7 respectively.

Sample outputs:

Size 1:	Size 2:	Size 4:	Size 7:
			

Besides the top and bottom of the picture frame, each line of the diagram consists of the following:

1. The left side of the picture frame, followed by
2. Some number of spaces (including zero spaces on the line in the center), followed by
3. A forward slash '`/`' if on the top half of the diamond or a back slash '`\`' if on the bottom half of the diamond, or a left angle bracket '`<`' if at the center of the diamond, followed by
4. Some number of hyphens '`-`' or equal signs '`=`' depending upon whether it was an even or odd line, followed by
5. A back slash '`\`' if on the top half of the diamond or a forward slash '`/`' if on the bottom half of the diamond, or a right angle bracket '`>`' if at the center of the diamond, followed by

6. Some number of spaces, followed by
7. The right side of the picture frame and a newline.

On any given execution, your program will produce just one version of the figure. However, you should refer to the variable `size` throughout your code, so that when a user enters a different value, your program would produce a figure of a different size. Your program should scale correctly for any `size` value of 1 or greater.

Another significant component of this assignment is the task of recognizing or identifying sections of code that are identical (or redundant) in different parts of your program, and eliminating that redundancy by defining a method that contain that code and then calling the method from the appropriate points.

Use of for-loops: This program is intended to test your knowledge of for-loops and conditionals. Some code will need to test whether you are printing an even numbered line or an odd numbered line. You can use the modulo division operator '%' and test the remainder after division by two. For full credit, you should also create helper methods that can handle common or redundant tasks for you.

Source code aesthetics (commenting, indentation, spacing, identifier names): You are required to properly indent your code and will lose points if you make significant indentation mistakes. No line of your code should be over 100 characters long (even better is limiting lines to 80 characters). You should use a consistent programming style. This should include the following.

- Create helper method as needed
- Meaningful variable & method names
- Consistent indenting
- Use of "white-space" and blank lines to make the code more readable
- Use of comments to explain pieces of tricky code

Submission:

You should just upload the file `Logic.java` for evaluation by the auto-grader.

Peer Assessment:

You will be asked to evaluate five other student's projects as well as do a self-assessment of your own project. If you do not assess the work of others, then you will receive a 10% penalty for your solution.