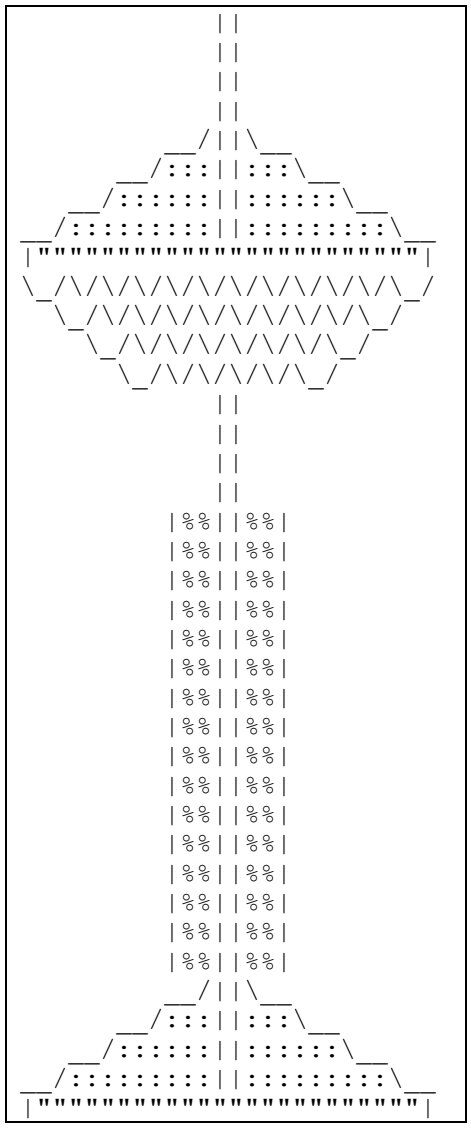
CS 210 Programming Assignment #2: ASCII Art (50 points)

Due Friday, October 12, 2012, 11:59 PM

This assignment focuses on `for` loops, expressions, variables, and constants.



Space Needle:

This assignment is to produce a specific text figure that is supposed to look like Seattle's Space Needle. Turn in a class named `SpaceNeedle` in a file named `SpaceNeedle.java`. You should **exactly** reproduce the format of the output at left. This includes having identical characters and spacing.

One way to write a Java program to draw this figure would be to write a single `System.out.println` statement that prints each line of the figure. However, this solution would only receive 25% of credit. A major part of this assignment is showing that you understand `for` loops.

In lines that have repeated patterns of characters that vary in number from line to line, represent the lines and character patterns using nested `for` loops. (See Chapter 2's case study.) It may help to write pseudocode and tables to understand the patterns, as described in the textbook.

Another significant component of this assignment is the task of generalizing the program using a single class constant that can be changed to adjust the size of the figure.

This assignment will be graded both on "external correctness" (whether the program compiles and produces exactly the expected output) and "internal correctness" (whether your source code follows the style guidelines in this document).

Style Guidelines:

Use of `for` loops (nested as appropriate)

This program is intended to test your knowledge through Chapter 2, especially nested `for` loops. If you like, you may also use the Java features from Chapter 3 such as parameters, although you are not required to do so and will receive no extra credit for doing so. You may not use any Java constructs beyond Chapter 3.

Use of static methods for structure and elimination of redundancy

Continue to use static methods to structure your solution in such a way that the methods match the structure of the output itself. Avoid significant redundancy; use methods so that no substantial groups of identical statements appear in your code. No `println` statements should appear in your `main` method. You do not need to use methods to capture redundancy in partial lines, such as the two groups of colons in the following line:

```
   __/::::::||::::::\__
```

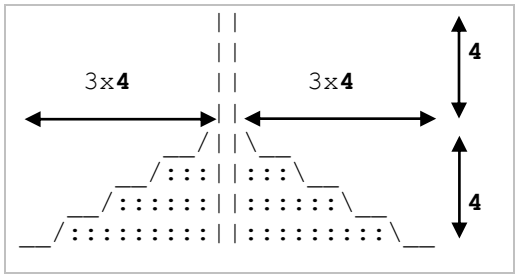
Source code aesthetics (commenting, indentation, spacing, identifier names)

You are required to properly indent your code and will lose points if you make significant indentation mistakes. See the textbook for examples of proper indentation. No line of your code should be over 100 characters long.

Give meaningful names to methods and variables in your code. Follow Java's naming standards about the format of `ClassNames`, `methodAndVariableNames`, and `CONSTANT_NAMES`.

Include a comment header at the beginning of your program with basic information and a description of the program. **Also include a comment at the start of each method**, describing its behavior. Your comments should be in your own words.

Class constant for figure's size



You should create one (and only one) class constant to represent the size of the pieces of the figure. Use **4** as the value of your constant. Your figure must be based on that exact value to receive full credit.

On any given execution your program will produce just one version of the figure. However, you should refer to the class constant throughout your code, so that by simply changing your constant's value and recompiling, your program would produce a figure of a different size. Your program should scale correctly for any constant value of 2 or greater.

Please note that the height of the needle's midsection grows as the square of the figure size. In the default figure size of 4, the midsection is 16 lines tall. If the size were 7, the midsection would be 49 lines tall.

Development Strategy (How to Get Started):

This program is best completed in stages. We strongly recommend the following development strategy:

1. **Tables**: Examine the output and write tables to discover the patterns of repeated characters on each line.
2. **Code w/o Constant**: Completely write the Java code to draw the Space Needle at its default size of 4.
3. **Code w/ Constant**: Add a constant to your code so that the needle can scale to different sizes.

To summarize, you should not worry about the constant at first. Write an initial program without a constant, using loop tables or pseudocode to help you deduce the patterns in the output. After your figure looks correct at the default size, begin a second version with the constant. See Chapter 2's case study for a good example program.

Submit your assignments using our Canvas Assignment tool:

- 1 source code file (*.java files)
- sample runs file: This should convince me that your program works even before I run it. Submit at least three images of different sizes. Annotate to describe all the different tests that you've tried.
- short summary of your work: What does or doesn't work and why, any additional functionality reasons or excuses for missing functionality. What you learned, what you found difficult or unclear about the assignment, etc.
- Place all of the above documents in a folder (name it you name), zip it and submit the zipped folder as an attachment.

Make sure you keep a copy of everything that you submit for your records.