CSE 190 D, Winter 2013
Programming Assignment #2: Rocket (14 points)
Due Monday, January 21, 2013, 11:30 PM

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

```
       *
      ***
     *****
    *******
    +++++++
    +++++++
    +++++++
    +++++++
   /+++++++\
  /-+++++++-\
 /--+++++++--\
/---+++++++---\
```

Your assignment is to produce a specific text figure that is supposed to look like a rocket. Turn in a class named `Rocket` in a file named `Rocket.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 not receive full 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 and lecture.

Another significant component of this assignment is the task of generalizing the program using a single variable that can be changed to adjust the scale of the figure. See the next page for a description of this variable and how it should be used in your program.

The course web site will contain files that show you the expected output if your scale variable is changed to various other values. You can use our Output Comparison Tool on the course web site to measure numbers of characters and to verify the correctness of your output for various values of the scale variable.

Your program 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 methods, 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.

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 variables in your code. Follow Java's naming standards about the format of `ClassNames` and `variableNames`.

Include a comment header at the beginning of your program with basic information and a description of the program. Also include a comment above any complex piece of code, describing its behavior. Your comments should be in your own words.

Variable for figure's size

You should create one (and only one) variable to represent the size of the pieces of the figure. Use **4** as the default value of your scale variable to produce the figure shown above. 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 scale variable throughout your code, so that by simply changing your variable's value and recompiling, your program would produce a figure of a different size. Your program should scale correctly for any constant value greater than 0.

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 scale variable**: Completely write the Java code to draw the rocket at its default size of 4.
3. **Code w/ scale variable**: Add a scale variable to your code so that the rocket can scale to different sizes.

To summarize, you should not worry about the scale variable at first. Write an initial program without a scale variable, 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 scale variable. See Chapter 2's case study for a good example program.