Homework #4

More practice with `for` loops and using class constants

1. Aim

The purpose of this lab is to:

a. Give you more practice with `for` loop structures by producing ASCII art.

b. Introduce you to class constants.

It is recommended that you read and understand all the instructions below before starting this exercise.

2. Files Needed

You will not be provided with any starter files for this homework. For this homework, you will be creating your own Java programs from scratch. Create a project in Eclipse for this assignment. Then for each program below, you will create a new Java class in the project. When you create the classes in Eclipse, be sure to have Eclipse create the `public static void main` method for you. Make sure you name the class exactly as specified. Be sure to include the standard header comments at the top of each file. The standard header comments are:

```
// Name:
// Email:
// Date:
// Description: <insert short description of the program here>
```

3. To be Handed In

The files **Window.java, DrawFigure1.java** and **DrawRocket.java** should be sent to me when you have them completed. Be sure to name the files/classes exactly as specified.

4. Exercises

Part I: Drawing a resizable figure

Create a Java class call **Window.java.** In that file, write a program that prints a window that has 4 window panes. The window uses equal signs for the horizontal pieces, vertical bars for the vertical pieces, and plus signs where the horizontal and vertical pieces meet. You should use a class constant to control the number of equal signs and vertical bars printed. For each pane, the number of equal signs printed is the same as the number of vertical bars printed.

Start the program by declaring a class constant that is the size of the desired window. There are many different ways to print the desired pattern, though currently we are limited to using only `for`-loops & method calls since those are the only construct we know so far that alter the flow of a program. This can be solved using only the material from chapters 1 & 2 from the class text, though you can also use the material in section 3.1 since we have also covered it.

You must use a class constant for the size of the desired window. Your printed window must be based on that constant value to receive full credit. On any given execution, your program will produce just one version of the window. 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 window of a different size. Your program should scale correctly for any positive value.

Sample outputs (one size = 6, the other size = 3):

```
+======+======+
|      |      |
|      |      |
|      |      |
|      |      |
|      |      |
|      |      |
+======+======+
|      |      |
|      |      |
|      |      |
|      |      |
|      |      |
|      |      |
+======+======+
```

```
+===+===+
|   |   |
|   |   |
|   |   |
+===+===+
|   |   |
|   |   |
|   |   |
+===+===+
```

Part II: Drawing a resizable figure

(Refer to project #1 on page 127 of the class text.) Create a Java class call **DrawFigure1.java** (that is a one digit, and not the letter L in the file name)**.** In that file write appropriate Java code to produce the desired figure.

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. It may help to write pseudocode and tables to understand the patterns, as described in the textbook and lecture.

Rather than writing a program that can only draw the given figure, we are going to enhance the program so that we can adjust the size of the figure. Add a single **class constant** to the program that specifies the number of lines that are to be drawn. Your program should produce a figure whose size is determined by the class constant. When you submit your program for grading, please set this class constant to the value 7.

Sample outputs:

```
Size 7:

****** //////////// ******
*****  //////////\\  *****
****   ////////\\\\   ****
***    //////\\\\\\    ***
**     ////\\\\\\\\     **
*      //\\\\\\\\\\      *
       \\\\\\\\\\\\
```

```
Size 4:

*** ////// ***
**  ////\\  **
*   //\\\\   *
    \\\\\\
```

Part III: Drawing ASCII art

Create a Java class call **DrawRocket.java.** In that file, complete the following exercise by writing appropriate Java code.

```
     /**\
    //**\\
   ///**\\\
  ////**\\\\
 /////**\\\\\
+=*=*=*=*=*=*+
|../\..../\..|
|./\/\../\/\.|
|/\/\/\/\/\/\|
|\/\/\/\/\/\/|
|.\/\/..\/\/.|
|..\/....\/..|
+=*=*=*=*=*=*+
|\/\/\/\/\/\/|
|.\/\/..\/\/.|
|..\/....\/..|
|../\..../\..|
|./\/\../\/\.|
|/\/\/\/\/\/\|
+=*=*=*=*=*=*+
     /**\
    //**\\
   ///**\\\
  ////**\\\\
 /////**\\\\\
```

You are to write a program that will print an ASCII art drawing of a rocket. 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 and related lecture notes.) It may help to write pseudo code 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 **class constant** that can be changed to adjust the size of the figure. See below for a description of this constant and how it should be used in your program.

The assignment page in Oak will contain files that show you the expected output if your size constant is changed to various other values. This 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 for the DrawRocket program:

As a reference, our solution for DrawRocket has 4 methods besides `main` and occupies around 90-100 lines of Java code including comments and blank lines, though you do not have to match this exactly.

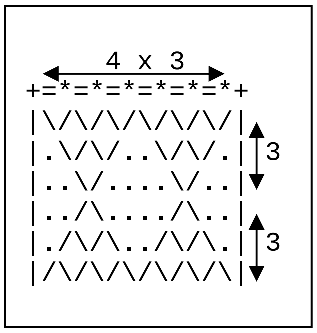
Use of `for` loops (nested as appropriate)
This program is intended to test your knowledge through Chapter 2, especially nested `for` loops. If you like, you may also use the Java features from Chapter 3 such as parameters, although you are not required to do so and will receive no extra credit for doing 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 `“\/”s` 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 go beyond column 80 of the editor window.

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

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 **3** as the value of your constant. Notice that the subfigures in the middle of the output have a height of 3, and the subfigures’ height determines their width, so there is only one size variable. 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.

5. Additional requirements

A. You must start your program with header comments that provide your name, VUnetID, email address, the date the program was last modified, an honor statement (“I have neither given nor received unauthorized help on this assignment”), and a short description of the program (see the examples distributed with homework #2).
B. You should use a consistent programming style. This should include the following.
    a. Meaningful variable & method names
    b. Consistent indenting
    c. Use of "white-space" and blank lines to make the code more readable
    d. Use of comments to explain pieces of tricky code
    e. A descriptive comment before each method (other than main, which already has a short description of the program)

See the code examples in the class text for a good formatting style.

6. Submission for grading

Once you have completed the exercise, submit the files **Window.java, DrawFigure1.java** and **DrawRocket.java** for grading by emailing all three files at once to me at roth.jerry@gmail.com.

7. Grading

This lab is worth 30 points, each part being worth 10 points. Your grade will be based on whether your solution is correct or not, and on how closely you followed the directions above. Remember that programming style will now be a part of your grade (on this and all subsequent assignments).