CS100 (A and B)
Programming Assignment 1

Due: Tuesday, September 7

Reminders

- You may work with a partner, but remember to uphold academic integrity.
- Read and follow the instructions.

Goals

This assignment has two parts that will help you become familiar with Java, Matlab, and the CodeWarrior (CW) environment. You must complete both parts to receive full credit. Please refer to your course packet and the *Guide to CodeWarrior Java for CS100 and CS211* for help and further information.

Motivation

Take any positive integer and do the following: if the integer is even, divide it by 2; otherwise multiply it by 3 and add 1. Repeat the process as long as the value remains greater than 1. For example, suppose you start with 11. Then the sequence you get is: 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1.

It has long been conjectured that regardless of the starting point, you will always reach 1, but so far, no one has ever been able to prove this. Like Fermat's Last Theorem, which stumped researchers for many years, this deceptively simple statement is hard to prove or disprove.

In this assignment, you are given a Java program to enter and run that computes such sequences for arbitrary initial integers (up to the largest positive integer that can be represented in a Java **`int`** variable). Also, all computed values cannot exceed size constraints of a Java **`int`** variable (see Figure 3.1 on page 76 in your textbook *Java Software Solutions*). The program can be used to experiment with these mysterious sequences. We also show you how to plot the sequences using MATLAB.

If you find an integer for which the attached program runs forever (and you haven't made a typo in entering the program), you have found a counter-example to the conjecture and will become famous. Call the New York Times.

Part 1: Java

Refer to the attached sheet for a sample Java program. Although many of the statements might appear unfamiliar to you, this program should give you the “flavor” of programming. Refer to the *Guide to CodeWarrior Java for CS100 and CS211*. If you need further help, look for the help file (assuming it’s installed), **Help|CodeWarrior Help|IDE and RAD|Create new projects|stationery** after starting CodeWarrior. Do the following tasks for this part.

1. Start CodeWarrior.
2. Obtain your CW guide. Carefully read it. Follow the instructions to create a new project and open the **CUCS Java Application** stationery file.
3. Type the entire contents of the program on the attached sheet inside your **CUCS Java Application** stationery. Be sure to delete the unnecessary portions of the code already written inside **CUCS Java Application**. (It’s OK if you delete the entire contents and retype the entire program, though you will create more work for yourself.) Also, remember to save your work frequently, though you will need to ensure the file is saved on your disk if you are using a public computer. Reminder: Carefully maintain the EXACT punctuation and capitalization – Java is case-sensitive! However, the spacing and indentation matter only for style and do not need to perfectly match.
4. Remember to follow the guidelines of Section 4 (Programs) of the *CS100 General Information* packet. (The first comment in each program must contain your name(s), Cornell ID#s, and day&time of your section.)
5. Select **Project|Compile**. Does a window pop-up warning you of compiler errors, also known as *syntax errors*? If so, carefully review the statements inside your code since, most likely, you typed something wrong.
6. When finished correcting your *typos* (typing-errors), select **Run**. Enter the integer 10 to confirm that your program generates the output:
```
5 after dividing by 2
8 after dividing by 2
4 after dividing by 2
2 after dividing by 2
1 after dividing by 2
Done!
```
7. Modify the program so that it produces output for the odd cases as well as the even cases. For instance, for the input **`10`**, the modified program should produce the output:
```
5 after dividing by 2
16 after multiplying by 3 and adding 1
8 after dividing by 2
4 after dividing by 2
2 after dividing by 2
1 after dividing by 2
Done!
```
Hints: Refer to the code that generates output for the even case. In Java, the next *single* statement following the **`else`** keyword is executed when the **`if`**-condition is false. If you wish to execute several statements when the **`if`**-condition is false, you must enclose the list of

statements in curly braces **{}**, just as in the even case.

8. Test your code again by compiling and running it. Be sure to fix any bugs you may have introduced.
9. Enter each of the following inputs and try to understand what happens:

0	i.e., the number zero
-19	i.e., a negative number
four	i.e., text
10,11,12	i.e., a list of integers
4/2	i.e., an arithmetic expression
12345678900	i.e., an integer too big to store in an int variable
1234567890	i.e., a large integer that is not too big to store in an int variable
123456789	i.e., a large integer

   "Large" integers might run a "really long time." To stop your program on a PC, click on the close-box (the icon with the letter X inside) and select **End Task** in the dialog box that pops up. To stop the program on a Mac, press Option-Command-Escape, as explained in the CW guide.
10. Run your modified code for integer input of **11**. Print the modified program. Print the output for running your program with the input of **11**.

Part 2: MATLAB

This part will very briefly introduce some aspects of MATLAB by plotting data you computed in Part 1 of this assignment. MATLAB provides a very convenient and powerful software tool for data analysis, which typically involves plots of data. Plotting your data from Part 1 will demonstrate how the algorithm you programmed converges upon the integer 1. Note: *Pressing Enter* means pressing either the Enter or Return key.

1. Consult your tip sheets and run MATLAB.
2. You should see a command window pop-up. This is where you will enter commands. Note: Do NOT type the prompt >>. Enter commands after the >>.
3. Enter the following command statements to generate a plot of your test case from Part 1, Problem 9.

   >> **plot([11 34 17 52 26 13 40 20 10 5 16 8 4 2 1])** *This command creates the initial plot of your data. The data is plotted on the vertical (Y) axis. Press the Enter key after typing the input. A graphics window containing your plot should pop-up.*

   >> **xlabel('count')** *This command labels the horizontal (X) axis with text. Press Enter after typing.*

   >> **ylabel('k')** *This command labels the Y-axis. Press Enter after typing.*

   >> **title('HW1: *<type your name(s) here>*')** *This command titles your plot with text. You should enter something like title('HW1: Pat Smith'). Press Enter after typing.*

   Matlab will automatically "update" (modify) your plot each time you press Enter.
4. Print out your plot.

What to Hand In

Read Section 4 (Programs) of the *CS100 General Information* packet for explanations of requirements. For this assignment, staple the following sheets together:

- Cover sheet with your name(s), student IDs, and sections clearly written on the top, right corner.
- Printout of modified code. Don't forget to sign the first page of your program!
- Printout of output from running program with input of **`11`**.
- Printout of MATLAB plot.

Grading

Read Section 4 (Programs) of the *CS100 General Information* packet for explanations of due date and grading policies.