Hailstone Sequence (Due 22 February 2013)

Take any natural number *n*. If *n* is even, divide it by 2 to get *n / 2*. If *n* is odd, multiply by 3 and add 1 to obtain *3 n + 1*. Repeat the process indefinitely. The conjecture is that no matter what number you start with, you will eventually reach 1.

This conjecture has never been proved. But it has been verified on the computer for all starting values up to 20 * 2^{58}. To disprove this conjecture one has to find a single starting number that goes into a cycle that does not contain 1.

```
The function can be written as follows:

     f ( n ) = ( n / 2 ) if n is even

     f ( n ) = ( 3 * n + 1 ) if n is odd
```

For a given starting number, this function generates a series of numbers ending at 1 that is called the hailstone sequence. The hailstone sequences for starting numbers 7, 8, and 9 are:

```
7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

8 4 2 1

9 28 14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

All sequences tested so far end in an endless cycle of 4, 2, and 1. Hence the sequences are halted when they reach 1. We will define the cycle length of a sequence to be the number of steps it takes from the starting number to reach 1. Here are the cycle lengths of a few starting numbers:

Number	Cycle Length
1	0
2	1
3	7
4	2
5	5

In your program you will verify this conjecture in a user defined range. You will prompt the user to enter the first number of the range and the last number. You will check if both numbers are positive (> 0) individually and then you will check that the first number in the user defined range is less than or equal to the last number in that range. Use nested loops to accomplish this. Keep prompting the user to input positive numbers in the right order.

Once the beginning and ending of the range have been verified, your program will compute the cycle length for each of the numbers in that range inclusive of the end points. Your program will print out the number that has the largest cycle length and what that cycle length is.

```
Your sample session will look like this:
```

```
Enter starting number of the range: 1

Enter ending number of the range: 5

The number 3 has the longest cycle length of 7.
```

In your program all your computations will be in the method main(). You must use nested loops to obtain your results.

The class that you will be writing will be called **Hailstone**. We will be looking at good documentation, design, and adherence to the coding convention mentioned below. You *must* use Scanner for your input. Your file Hailstone.java will have the following header:

```
/*
  File: Hailstone.java

  Description:

  Student Name:

  Student UT EID:

  Course Name: CS 312

  Unique Number:

  Date Created:

  Date Last Modified:

*/
```

You will follow the standard Java Coding Conventions. You can either view the HTML page or download the PDF or Postscript and print it out. There is a modification that I would like to make to the standard coding conventions. Please align the opening and closing braces vertically so that you can easily make out the blocks of code. For example:

```
Do this:
if ( x > 5 )
{
  a = b + c;
}

Not this:
if ( x > 5 ) {
  a = b + c;
}
```

Use the turnin program to submit your **.java** file. We should receive your work by 11 PM on Friday 22 February 2013. There will be substantial penalties if you do not adhere to the guidelines.

- You must submit the .java file and not the .class file.
- Your .java file should have the header with the proper documentation.

- You should be submitting your .java file through the web based *turnin* program. We will not accept files e-mailed to us.
- Compile and run your code on the command line.
- Your code must compile before submission.
- Here is the Grading Criteria.

References

1. Collatz Conjecture
2. Collatz Problem in Wolfram Math World