NAME: **LOGIN ID:**

PRECEPT:

COS 126 Midterm 1 Programming Exam Fall 2014

This part of your midterm exam is like a mini-programming assignment. You will create three programs, compile them, and run them on your laptop. Debug your code as needed. This exam is open book, open browser—but only our course website and booksite! Of course, no internal or external communication is permitted (e.g., talking, email, IM, texting, cell phones) during the exam. You may use code from your assignments or code found on the COS126 website. When you are done, submit your program via the course website using the submit link for Programming Exam 1 on the Assignments page.

Grading. Your program will be graded on correctness, clarity (including comments), design, and efficiency. You will lose a substantial number of points if your program does not compile or does not have the proper API, or if it crashes on typical inputs.

Even though you will electronically submit your code, *you must turn in this paper* so that we have a record that you took the exam and signed the Honor Code. *Print your name, login ID, and precept number on this page* (now), and write out and sign the Honor Code pledge before turning in this paper. *Note*: It is a violation of the Honor Code to discuss this midterm exam question with anyone until after everyone in the class has taken the exam. You have 90 minutes to complete the test.

"I pledge my honor that I have not violated the Honor Code during this examination."

____

[*copy the pledge onto this line*]

____

[*signature*]

Part 1	/13
Part 2	/11
Part 3	/ 6
TOTAL	/30

The *Collatz Conjecture*, proposed by Lothar Collatz in 1937, states that for any natural number *N*, the *Collatz sequence* starting at *N* has finite length. (If *N* is even, divide by 2; if *N* is odd, multiply by 3 and add 1, and continue until reaching 1.) In lecture, we looked at the following recursive code to print Collatz sequences, (see `Collatz.java` in Section 2.3 of the booksite).

```
public class Collatz
{
   public static void collatz(int N)
   {
      StdOut.print(N + " ");
      if (N == 1) return;
      if (N % 2 == 0) collatz(N/2);
      else collatz(3*N + 1);
   }

   public static void main(String[] args)
   {
      int N = Integer.parseInt(args[0]);
      collatz(N);
      StdOut.println();
   }
}
```

This implementation is useful for studying Collatz sequences:

```
% java-introcs Collatz 7
7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
% java-introcs Collatz 8
8 4 2 1
```

For this exam, your task is to write programs to study the *length* of these sequences. Start by creating a class `F14Part1` in a file `F14Part1.java` consisting of the `collatz()` function above and an empty `main()`.

Part 1 (13 points). Add to `F14Part1` a public *nonrecursive* static method `seqLength()` that has one integer argument *N* and returns the length of the Collatz sequence starting at *N*. You may assume that *N* is positive. Use a `while` loop and a counter to compute the length. To test your program, fill in `main()` with code that takes *N* from the command line, computes the length of its Collatz sequence, calls `collatz()` to print the sequence if the length is less than 20, and then prints the length within square brackets. For example, your program must produce this output:

```
% java-introcs F14Part1 7
7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1 [17]
% java-introcs F14Part1 1000
[112]
```

Submission. Submit your `seqLength()` and `main()` in `F14Part1.java` via Dropbox at

https://dropbox.cs.princeton.edu/COS126_F2014/Exam1

(This is the submit link for Programming Exam 1 on the Assignments page.) Be sure to click the *Check All Submitted Files* button to verify your submission.

Part 2 (11 points). Compose a program `F14Part2.java` that defines a public static method `maxLength()` that has one integer argument *N* and returns the *maximum* length of the Collatz sequences for all positive integers less than or equal to *N*. Compute each sequence length by calling `F14Part1.seqLength()` (which runs the code in your `F14Part1.java` file). For example, the sequence lengths for 1 through 8 are 1, 2, 8, 3, 6, 9, 17, and 4 (use your program from Part 1 to learn this), so the maximum when *N* is 8 is 17. Write a `main()` that just takes *N* from the command line and prints the maximum. Your program must produce this output:

```
% java-introcs F14Part2 8
17

% java-introcs F14Part2 1000
179

% java-introcs F14Part2 10000
262
```

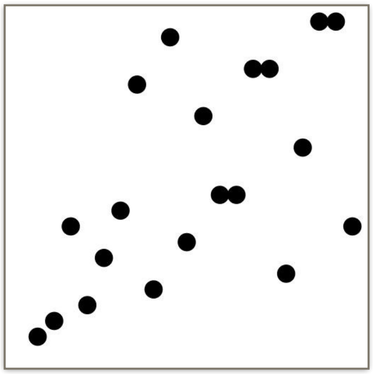
Submission. Submit your `maxLength()` and `main()` in `F14Part2.java` via Dropbox at

https://dropbox.cs.princeton.edu/COS126_F2014/Exam1

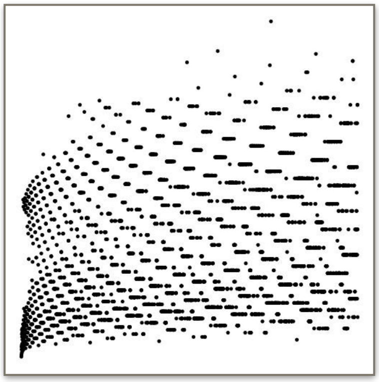
Again, be sure to click the *Check All Submitted Files* button to verify your submission.

Part 3 (6 points). Your task in this part is to visualize the sequence lengths. Write a program `F14Part3` that takes from the command line an `int` value *N* and plots a point at `(i, F14Part1.seqLength(i))` for each integer *i* from 1 to *N* in a `StdDraw` window. Use `StdDraw.point()` with a pen radius of .05 when *N* is less than 100, .01 otherwise. All the points you plot must have coordinates in the range 0 to 1, and fill the unit square `StdDraw` window, as shown below. Use *N* or `F14Part2.maxLength()` where appropriate to scale your points. Do *not* use our scaling methods `StdDraw.setXscale()` and `StdDraw.setYscale()`. Your program must produce output like this:

```
% java-introcs F14Part3 20
```



```
% java-introcs F14Part3 5000
```



Submission. Submit `F14Part3.java` via Dropbox at

https://dropbox.cs.princeton.edu/COS126_F2014/Exam1

Again, be sure to click the *Check All Submitted Files* button to verify your submission.