NAME: **LOGIN ID:**

PRECEPT:

COS 126 Midterm 1 Programming Exam Fall 2013

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*]

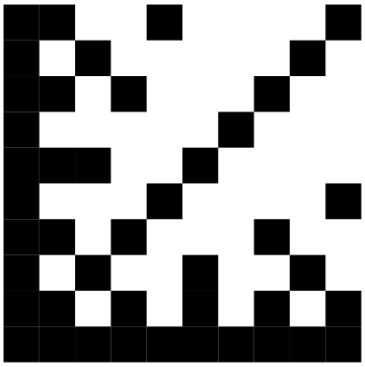
_____

[*signature*]

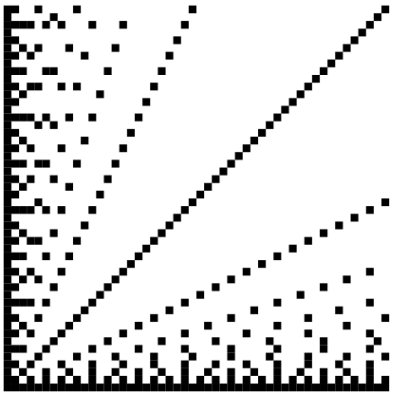
Part 1	/13
Part 2	/13
Part 3	/ 4
TOTAL	/30

Part 1 (13 points). Recall that we can use the Java remainder operator `%` to determine whether one integer evenly divides another: the value of the expression `98%7` is `0` because 7 evenly divides 98. Your task for this part is to visualize this concept. Write a `StdDraw` client `Part1` that takes from the command line an `int` value N and fills the `StdDraw` window with an N-by-N grid of squares such that the square in row i and column j colored `BLACK` if i evenly divides j or j evenly divides i, `WHITE` otherwise. Consider the rows to be numbered from 1 to N from bottom to top and the columns to be numbered from 1 to N from left to right. (The squares in the left column and the bottom row should all be `BLACK`, because 1 evenly divides every positive integer.) For example, your program must produce precisely the following patterns for $N = 10$ and $N = 50$.

```
% java-introcs Part1 10
```



```
% java-introcs Part1 50
```



It is not essential that your image be precisely centered or that it fill 100% of the drawing area. Your public API should consist of one item, the usual static `main()` method.

Hint: This program is easier if you use `StdDraw.setXscale()` and `StdDraw.setYscale()`.

Submission. Submit `Part1.java` via Dropbox at

https://dropbox.cs.princeton.edu/COS126_F2013/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 (13 points). Your task for this part is to write a program that prints the *greatest common factor* (GCF) of two positive integers (the largest integer that divides them both). For example, the GCF of 98 and 70 is 14 because $98 = 2 \times 7 \times 7$ and $70 = 2 \times 5 \times 7$, and the greatest common factor of 98 and 85 is 1 because they have no common factors greater than 1. Write a program `Part2` that reads from standard input two positive `int` values p and q and prints their GCF. To compute the GCF, define a public recursive static method `gcf()` that implements the following approach: the GCF of two numbers is their value if they are equal, and the GCF of the smaller one and the

absolute value of their difference otherwise. (This method, attributed to Euclid, was often used in the early days of computing because it does not use division, which was not available on early machines.) Your public API should consist of two items, the `gcf()` method (which must take two `int` arguments and return an `int` value) and a static `main()` method that takes two integers from standard input and prints their GCF:

```
% java-introcs Part2
98 70
14
```

In this example, the `98 70` was typed by the user as standard input and the line `14` was printed to standard output by the program.

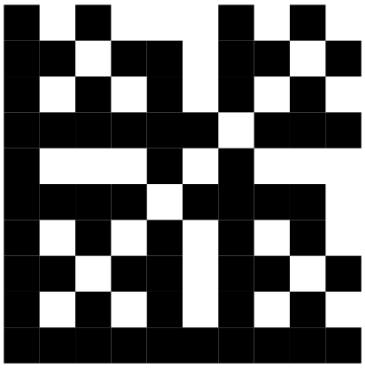
Submission. Submit `Part2.java` via Dropbox at

https://dropbox.cs.princeton.edu/COS126_F2013/Exam1

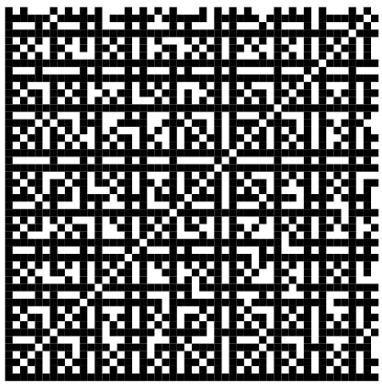
Again, be sure to click the *Check All Submitted Files* button to verify your submission.

Part 3 (4 points). Your task in this part is to write a `StdDraw` and `Part2` client that visualizes the concept of *relatively prime* pairs of integers. Two integers are relatively prime if they share no common factors. For example, 98 and 85 are relatively prime because 98=2×7×7 and 85=5×17, so they share no common factors, but 98 and 70 are not relatively prime because 7 is a common factor. In this definition, we do not count 1 as a factor of any number. Write a program `Part3` that takes from the command line an `int` value N and fills the `StdDraw` window with an N-by-N grid of squares with the square in row i and column j colored `BLACK` if i and j are relatively prime, `WHITE` otherwise, using the same numbering conventions as specified in Part 1. For example, your program must produce precisely the following patterns for $N = 10$ and $N = 50$.

```
% java-introcs Part3 10
```



```
% java-introcs Part3 50
```



Submission. Submit `Part3.java` via Dropbox at

https://dropbox.cs.princeton.edu/COS126_F2013/Exam1

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