CSC171 — Lab 2

Iteration and Control

This assignment will give you practice with the basics of iteration. In order to keep things simple, you can put all your answers into one main method for this one. Soon, we will see how to define methods and design more nuanced solutions. You should name your class `HW2`, and all work should go into a file named `HW2.java`. Your file must begin with a block comment listing your name, netid (email), and the assignment type/number. You should also add blank lines to the output between the three parts.

1. Add a print statement to say "Multiplication Table" and then prompt the user to enter an integer. Your program should then prints "times table" for the numbers from 1 to the given integer (inclusive). That is, for input n, you would print a table of n rows (lines) each with n columns, where the cell at row i and column j contains the value $i \times j$. You do need separate rows, but don't worry about the columns lining up nicely yet.

2. Add a print statement to say "Prime Testing".

   Using the integer entered by the user above, print out all the prime numbers between 2 and the number squared. E.g., if the user types 12 for the multiplication table part, then you should print all the prime numbers which are not larger than 144.

   All integers are either *prime* or *composite*. A prime number is one which has no integer factors other than itself or one. Examples are 2, 3, 5, 7, 11, 13, ... A composite number, by contrast, has multiple factors. Examples are 4, 6, 12, 15, 18, 21, ...

   Testing whether or not a number is prime is an important application of computing power. One very simple method is to simply check every possible factor from 2 up to the number in question.

   For example to test whether or not the value 35 is prime, you can test if 35 % 2 == 0, then if 35 % 3 == 0, then is 35 % 4 == 0, and so on, up until you find a number that evenly divides 35 or confirm that no such number exists. You must write this yourself, and cannot use any library methods for primality testing.

   A helpful list of prime numbers is available on Wikipedia. You may wish to use this to test the accuracy of your code: https://en.wikipedia.org/wiki/List_of_prime_numbers

There is one more problem on the next page.

3. Add another print statement that says “Infinite Series Test”.

   Consider the infinite series

$$\sum_{i=1}^{\infty} \frac{1}{2^i} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \cdots$$

   This time you should prompt the user to enter a second integer k, and then calculates and prints the sum of the first k terms of the sequence. Do not use `Math.pow`. Instead, compute the value of 2^k incrementally in your loop. Try your program for different values of k. Do you notice a trend?

Submit your work as a single file named `HW2.java` to blackboard by 1159pm on Thursday September 29. All problems are equally weighted. The instructor may deduct points for code with inconsistent style or unnecessary package or module declarations. The instructor may also deduct points for submitting your answer in the wrong format. If your program does not compile and execute as submitted, you will receive a zero. Your program must be able to be successfully compiled without modification in order for it to be graded.