**CSC142, Computer Science II, Project 2 assignment**

Develop the following programs and submit each java file to D2L. Later submission is not accepted.

Part 1: Basic concepts (Extend the lecture samples for the solution)

1. Sum.java: Write a program that will repeatedly prompt the user to type a number until the user types a POSITIVE number, then print the sum of the digits of that number. For instance, 12345 will lead to the print of 15.
2. FiveOnly.java: Write a program that asks the user for the name of a file. The program should display only the first five lines of the file’s contents. If the file contains fewer than five lines, it should display the file’s entire contents. See Challenge 13 on p265.
3. Perfect.java: Write a program that reads in an integer *N* from the keyboard, and displays whether *N* is a "perfect number" or not (Yes or No). A number is "perfect" if it is equal to the sum of all of its factors (not including itself as a factor, but including 1 as a factor). 6 is the first perfect number, because its factors are 1, 2, and 3, and 1+2+3 = 6.

Part 2: Fencepost problems (Focus on specified initialization).

1. MinAvg.java: Write a program that reads 10 integers and displays its minimum and average. Please consider the extreme case when all integers are positive. For instance, by given 1, 2, …, 10, the minimum will be 1.
2. PrintGrid.java: Write a program accepts two integers *rows* and *cols*. The output is a comma-separated grid of numbers where *rows* represents the number of rows of the grid and *cols* represents the number of columns. The numbers count up from 1 to (*rows* x *cols*). The output will be displayed in column-major order, meaning that the numbers shown increase sequentially down each column and wrap to the top of the next column to the right once the bottom of the current column is reached. You may assume that both inputs are greater than 0.

Here are some example calls to your method and their expected results:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | 3 6 | 5 3 | 4 1 | 1 3 |
| **Output** | 1, 4, 7, 10, 13, 16  2, 5, 8, 11, 14, 17  3, 6, 9, 12, 15, 18 | 1, 6, 11  2, 7, 12  3, 8, 13  4, 9, 14  5, 10, 15 | 1  2  3  4 | 1, 2, 3 |

Part 3: Loop design (Need a comprehensive plan)

1. Duration.java:

Emma and her colleagues plan to have lunch outside today. Right now, it is 11:34. Emma’s boss wants her back before 13:15 for a very important meeting. In order to help Emma to know if she has enough time, develop a program and tell the time duration from now to the meeting in the format of HH:MM. Your program should support the input of any given pair of start/end time via the keyboard.

Use loop to avoid invalid inputs. Ensure the hour input and the minute input is in the right range [0..23] and [0..59] respectively. When start time is later than end time, ask the user to choose reenter both times; otherwise, quit the execution without giving any wrong information in the display. Please be aware of the correct use of next, nextLine, and nextInt.

1. Max3.java: Write a program that reads 10 integers and displays top 3 records (i.e., the largest three elements). Please consider the extreme case when all integers are negative. For instance, by given -1, -2, …, -10, the display will be -1, -2, and -3.
2. ESP.java: challenges 19, p266. Note that all colors should be handled by using their numeric identity. This is a number problem, not a String problem!

