SYSC 2100 Algorithms and Data Structures
Winter 2019
Assignment 1: Recursion
Due: Feb 1st, 2019

All the methods asked below are to be included in a class named ***Assignment1***. **Assignment1.java** and **Assignment1.class** are the only 2 files to submit. Feel free to include a ***main()*** function in your ***Assignment1*** class so that you can run/test the methods that you are asked to write.

Name your methods strictly as specified (case sensitive). When marking, the TAs will use a class (***MarkingAssignment1***) that inherits yours (***Assignment1***). The TAs' class calls the methods by the names that are specified here. Therefore it is of utmost importance that you abide by these names (case sensitive). Any discrepancy in the name will cause issues.

1. Write a Java method for the Spock problem ***c(n, k)*** seen in class. Include output code that shows the actual sequence of calls that are made and the value that they will return when the method is executed. For example, ***c(3, 2)*** outputs the following:

   c(3,2) = c(2,1) + c(2,2)

   c(2,1) = c(1,0) + c(1,1)

   c(1,0) = 1

   c(1,1) = 1

   c(2,2) = 1

   Name your method **c**.

   Example method specification: `public static int c(int n, int k)`

2. Write a Java method for the problem of *Organizing a Parade* as presented in class. Name your method ***P***.
   Example method specification: `public static int P(int n)`

3. Write a recursive Java method ***writeLine*** that writes a character repeatedly to form a line of n characters. For example, ***writeLine('*',5)*** produces the line *****.
   Then write a recursive method ***writeBlock*** that uses ***writeLine*** to write m lines of n characters each. For example, ***writeBlock('*', 5, 3)*** produces the output

   *****
   *****
   *****

Example method specification: `public static void writeLine(char ch, int n)`
`public static void writeBlock(char ch, int n, int m)`

4. Write a recursive Java method that writes the digits of a positive decimal integer in reverse order. Name your method ***reverseDigits***.
   Example method specification: `public static void reverseDigits(int number)`

5. Implement the recursive binary search algorithm presented in class for an array of strings. Name your method ***myBinarySearch*.**

   Example method specification: `public static int myBinarySearch(String [] anArray, int first,int last, String value)`

Submission Requirements: Submit your assignment (**the source file (Assignment1.java) and bytecode (Assignment1.class**)) using **cuLearn**. Your program should compile and run as is in the default lab environment, and the code should be well documented. Submit all the files individually **without using any archive or compression**.

Marks will be based on:

- Completeness of your submission
- Correct solution to the problem
- Following good coding style
- Sufficient and high-quality in-line comments
- Adhering to the submission requirements (in particular the naming convention and the submission of uncompressed source files only)

The due date is based on the time of the **cuLearn** server and will be strictly enforced. If you are concerned about missing the deadline, here is a tip: multiple submissions are allowed. So you can always submit a (partial) solution early, and resubmit an improved solution later. This way, you will reduce the risk of running late, for whatever reason (slow computers/networks, unsynchronized clocks, failure of the Internet connection at home, etc.).

In **cuLearn**, you can manage the submission until the deadline, taking it back, deleting/adding files, etc, and resubmitting it. The system also provides online feedback whether you submitted something for an assignment. It may take a while to learn the submission process, so I would encourage you to experiment with it early and contact the TA(s) in case you have problems, as only assignments properly and timely submitted using **cuLearn** will be marked and will earn you assignment credits.