CSCI 2110 Data Structures and Algorithms
Lab No. 5
Week of October 16th, 2017

This lab is on recursion.

Remember – in order to design a recursive program, you need to come up with a (a) base case – this is the case where the problem can be solved without recursion and (b) recursive case – this is the case where the problem calls a smaller version of itself. Once you are able to define the problem in this manner, programs in recursion are very simple (they involve minimal coding).

Marking Scheme
Each exercise carries 10 points. Your final score will be scaled down to a value out of 10.
Working code, Outputs included, Efficient, Good basic comments included: 10/10
No comments: subtract one point
Unnecessarily inefficient: subtract one point
No outputs: subtract two points
Code not working: subtract up to six points depending upon how many methods are incorrect.

For exercises 1(c), 2, 3, 4, 5 and 6, DO NOT HARDCODE THE INPUTS.

Error checking*: Unless otherwise specified, you may assume that the user enters the correct data types and the correct number of input entries, that is, you need not check for errors on input.*

Submission*: All submissions are through Brightspace. Log on dal.ca/brightspace using your Dal NetId. Submissions are pretty straightforward. Instructions will be also be given in the first lab.*
Deadline for submission: Sunday, October 22nd, 2017 at 11.55 p.m. (five minutes before midnight).

What to submit*:*
A zip file containing all source codes and a text document containing sample outputs.

Exercise 1: These are recursive programs discussed in the lectures.

a) Fibonacci series
0,1,1,2,3,5,8,13,21,34,…
if n==0, then fib(n)=0 //base case
if n==1, then fib(n)=1 //base case
if n>1, then fib(n) = fib(n-1) + fib(n-2)

Write a recursive method public static int fib(int n) that finds the nth number in the Fibonacci series.
Use this method in the main method to print the first 20 numbers in the Fibonacci series.

b) Factorial of an integer n
if n == 0, then factorial(n) = 1
if n > 0, then factorial(n) = n * factorial(n-1)

Write a recursive method public static int factorial(int n) that finds factorial of a non-negative integer n. Use this method in the main method to print the factorials of 1 to 10.

c) x to the power n
if n == 0, then power(x,n) = 1
if n>0, then power(x,n) = power(x,n-1)*x

Write a recursive method public static int power(int x, int n) that finds the x to the power n, where x and n are positive integers. Use this method in the main method that prompts the user to enter x and n and prints x to the power of n.

Exercise 2: Write a recursive method called countDown(int n) that takes a single int parameter, n>=0, and displays a countdown. For example, countDown (10) would display
10 9 8 7 6 5 4 3 2 1 BlastOff!

Write a main method that prompts the user to enter a positive integer n and counts down to 1 from n.

Exercise 3: Modify the above method so that if n is even, it displays only even numbers in the countdown, whereas if n is odd, it displays only odd numbers in the countdown.
For example, countDown(10) would print
10 8 6 4 2 BlastOff!
and countdown(9) would print
9 7 5 3 1 BlastOff!

Exercise 4: Write a recursive method to display the first m multiples of a positive integer n. For example, if n is 2 and m is 5, it should display 2, 4, 6, 8, 10. As a second example, if n is 3 and m is 6, then it should display 3, 6, 9, 12, 15, 18 (Order doesn't matter).

Write a main method that prompts the user to enter a positive integer n and displays as above.

Exercise 5: Write a recursive method public static void writeVertical(int n) that takes one non-negative integer and displays that integer with the digits going down the screen one per line. For example, the invocation writeVertical(1234) would produce the output
1
2
3
4
Write a main method that prompts the user to enter a positive integer n and displays as above.

Exercise 6: Write a recursive method public static int squares(int n) that takes one non-negative integer and displays the sum of squares of all integers up to n. For example, squares(4) should display the value of the sum $1^2 + 2^2 + 3^2 + 4^2$

Write a main method that prompts the user to enter a positive integer n and displays as above.

Exercise 7: Rewrite the Tower of Hanoi program discussed in the lectures. Modify it to display the number of moves for a given number of discs.
Determine the execution time for solving the Tower of Hanoi problem given n, the number of discs. You can use the following code template to obtain the execution time:

```
long startTime, endTime, executionTime;
startTime = System.currentTimeMillis();

//code segment
```

```
endTime = System.currentTimeMillis();
executionTime = endTime – startTime;
```

The above segment will give the time for executing the code segment in milliseconds.

Use the following values of n: 8, 12, 16, 20, 24, 28, and 32. Record your values of execution time vs. n in a table.