DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING
CONCORDIA UNIVERSITY
COMP 352: Data Structure and Algorithms
Winter 2018
ASSIGNMENT 1
Due date and time: Wednesday January 31st, by midnight

***Written Questions (50 marks)**:*

Q1. What is the big-Oh (O) time complexity for the following algorithm (shown in pseudocode) in terms of input size *n*? Show all necessary steps:

(a)
```
Algorithm MyAlgorithm (A,B)
   Input: Arrays A and B each storing n >= 1 integers.
   Output: What is the output? (Refer to part b below)
   Start:
   count = 0
   for i = 0 to n-1 do {
      sum = 0
      for j = 0 to n-1 do {
         sum = sum + A[0]
         for k = 1 to j do
           sum = sum + A[k]
      }
      if B[i] == sum then count = count + 1
   }
   return count
```

(b) Document a hand-run on `MyAlgorithm` for input arrays A = [1 2 5 9] and B = [2 29 40 57] and show the final output.

Q2. Consider the following code fragments (a), (b) and (c) where *n* is the variable specifying data size and *C* is a constant. What is the big-Oh time complexity in terms of *n* in each case? Show all necessary steps.

(a)
```
for (int i = 0; i < n; i = i + C)
     for (int j = 0; j < 10; j++)
            Sum[i] += j * Sum[i];
```

(b)
```
for (int i = 1; i < n; i = i * C)
     for (int j = 0; j < i; j++)
          Sum[i] += j * Sum[i];
```

(c)
```
for (int i = 1; i < n; i = i * 2)
    for (int j = 0; j < n; j = j + 2)
        Sum[i] += j * Sum[i];
```

Q3. The number of operations executed by algorithms *A* and *B* are $12n^3 + 40n \log n$ and $5n^4 - 100n^2$ respectively. Determine an n_0 such that *B* is greater than *A* for $n \geq n_0$.

Q4. Answer the following questions:

a) Show that if *d(n)* is *O(f(n))* and *e(n)* is *O(g(n))*, then *d(n) + e(n)* is *O(f(n) + g(n))*.
b) Show that if *d(n)* is *O(f(n))* and *e(n)* is *O(g(n))*, then *d(n) – e(n)* is not necessarily *O(f(n) – g(n))*.

c) Show that $2^{n+1} + n^2$ is $O(2^n)$

d) Show that $f(n) = \sum_{i=1}^{n} i^2$ is $O(n^3)$

***Programming Questions (50 marks)**:*

Referring to the slides from text book, Chapter 5 (slides are available from the textbook link provided in Moodle) there are two versions of Fibonacci number calculators: *BinaryFib(n)* and *LinearFibonacci(n)*. The first algorithm has exponential time complexity, while the second one is linear.

a) In this programming assignment, you will implement in Java both the versions of Fibonacci calculators and experimentally compare their runtime performances. For that, with each implemented version you will calculate Fibonnaci (5), Fibonacci (10), etc. in increments of 5 up to Fibonacci (100) (or higher value if required for your timing measurement) and measure the corresponding run times. You need to use Java's built-in time function for this purpose. You should redirect the output of each program to an *out.txt* file. You should write about your observations on timing measurements in a separate text or pdf file. You are required to submit the two fully commented Java source files, the compiled executables, and the text/pdf files.

b) Briefly explain why the first algorithm is of exponential complexity and the second one is linear (more specifically, how the second algorithm resolves some specific bottleneck(s) of the first algorithm). You can write your answer in a separate file and submit it together with the other submissions.

c) Do any of the previous two algorithms use tail recursion? Why or why not? Explain your answer. If your answer is ``No'' then

i. design the pseudo code for a tail recursive version of Fibonacci calculator;
ii. implement the corresponding Java program and repeat the same experiments as in part (a) above. You will need to submit both the pseudo code and the Java program, together with your experimental results.

Submit all your answers to written questions in PDF or text formats only. For the Java programs, you must submit the source files together with the compiled executables. The solutions to all the questions should be zipped together into one .zip file and submitted via EAS (Refer to the course outline for more details on submission guidelines).