COMP 3761 Assignment 1

Due Date: Friday **July 17, 2009** at 6:30pm in class **SE12-321**

Instructions:

- No late assignments will be accepted.
- This is an individual assignment. You are welcome to discuss the assignment with others, but you **must** submit your own solutions.
- **Cheating**, as outlined in BCIT Policy 5002, is not allowed. Copying solutions from or to others will result in **zero** for the entire assignment for all involved individuals.
- For the programming parts, please submit your source codes into the BCIT SHARE IN under COMP3761 A1 folder. In addition, please print out your source codes, and submit a hardcopy of the source code together with your test results.
- For all written questions, ensure to show your steps to support your answers.

1. Show whether each of the following is true or false: [5 marks]
   a. $100000n^2 + 5n \in O(n^2)$
   b. $100n^2 + 5n \in O(n^3)$
   c. $100n^2 + 5n \in \Theta(n^3)$
   d. $n^5 \notin O(n^2)$
   e. $2^n \in \Theta(3^n)$.

2. Given the following list of functions:

$$\log n, \quad n, \quad n \log n, \quad n^2, \quad n^3, \quad 2^n, \quad n!.$$

Prove that the functions are listed in increasing order of their order of growth. [10 marks]

3. Order the following functions according to their order of growth (from the lowest to the highest): [5 marks]

$$(n-2)!, \quad 5\lg(n+100)^{10}, \quad 2^{2^n}, \quad 0.001n^4 + 3n^3 + 1, \quad 3^n.$$

4. For each of the following algorithms, show: (1) the actual number of basic operations performed, and (2) the Big-O complexity. [10 marks]

a. for $i = 0..n$
    for $j = 1..10$
      x++;

b. for $i = 1..n$
    for $j = 1..i$
      for $k = 1..i$
        x[j][i]++;

c. for $k = 2..n$
    for $j = 1..3n$
      $x \leftarrow a[k] - b[j]$;

d. for $i = 1..n-1$
    for $j = i+1..n$
      if $a[j] > a[i]$
        temp $\leftarrow a[i]$
        $a[i] \leftarrow a[j]$
        $a[j] \leftarrow$ temp

5. Given three different algorithms whose time complexities are expressed as the following functions:

a. $t(n) = 1000n$

b. $f(n) = 5n^2$

c. $g(n) = 0.0001 * 2^n$.

Suppose the input size is tripled, determine how much longer will it take to execute each of the above algorithms on the same computer? [10 marks]

6. Solve the following recurrence relations. [10 marks]

   a. $T(n) = 2T(\frac{n}{2}) + n, \quad \text{for } n > 1, \quad T(1) = 3.$

   b. $g(n) = g(\frac{n}{3}) + 1, \quad \text{for } n > 1, \quad g(1) = 1.$

7. Implement the following sorting algorithms:

   a. Selection sort

   b. Bubble sort

   To test your programs, generate some random inputs of size n in different magnitudes (e.g, $n = 10, 100, 500, 1000, 5000, 10000, \cdots$), record the actual number of basic operations executed in each case. Analyze your test results and verify the theoretical assertions about their time efficiencies. Submit your source codes and test results. Be prepared to demonstrate your programs in the lab. [20 marks]

8. Consider the problem of polynomial evaluation: Computing the value of the polynomial

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$$

   at a given point x_0. [10 marks]

   a. Design a brute-force algorithm whose worst-case efficiency class is in $\Theta(n^2)$. [4]

   b. Design a linear algorithm for this problem. [5]

   c. Is it possible to design an algorithm with a better than linear efficiency for this problem? Why or why not? [1]

9. Consider the problem of counting, in a given text, the number of substrings that start with an A and end with a B. For example, there are four such substrings in CABAAXBYA. [20 marks]

   a. Design a brute-force quadratic algorithm for this problem. [5]

   b. Design a linear algorithm for this problem. [7]

c. Implement both of your algorithms and verify they work as expected. [8]