Assignment 3

**Date Assigned: 04/13/2016**

**Due: Midnight 04/27/2016 on iLearn**

**Please read turn-in checklist at the end of this document before you start doing exercises.**

**Section 1: Pen-and-paper Exercises**

1. Trace the quick sort on the following array:

{11, 7, 6, 48, 30, 12, 75}

Note: Write down the state of the array after each recursive call. For each recursive call, let the pivot be the leftmost element/first element in the subrange (5 points).

1. Consider the following numerical questions game. In this game, player 1 thinks of an integer in the range 1 to n, where n is an integer. Player 2 has to figure out this number by asking the fewest number of true/false questions.

For example, a question may be “Is your number larger than x?”

Assume that nobody cheats.

(a) What is an optimal strategy if n in known? Describe your algorithm (English description 5 points), and Analyze your algorithm’s running time (5 points).

(b) What is a good strategy if n is not known? Describe your algorithm (English description 5 points), and Analyze your algorithm’s running time (5 points).

1. Given an array A of n elements, find out the maximum difference between any two elements such that larger element appears after the smaller number in A. In other words, find a pair of elements (a\_p, a\_q) with 0<=p < q <n such that (a\_q – a\_p) is the maximum among all such pairs in A. For example, if array is [2, 3, 10, 6, 4, 8, 1] then the maximum difference should be 8 (Diff between 10 and 2). If array is [ 7, 9, 5, 6, 3, 2 ] then the maximum difference should be 2 (Diff between 7 and 9). Design an algorithm of O(n) to solve this problem. Describe your algorithm (English description 5 points + Pseudocode 5 points), and Analyze your algorithm’s running time (5 points). **Note: This problem will be discussed in class. Algorithms that are O(nlogn) or slower will be scored out of 10 points.**
2. Consider the following problem:

Let S1 and S2 be lists of elements. Each contains n elements in an increasing order.

For a given x, the problem is to find whether there exist an element in S1 and an element in S2 whose sum is exactly equal to x. Give an algorithm that solves this problem in O(n) time in the worst case. Describe your algorithm (English description 5 points + Pseudocode 5 points), and Analyze your algorithm’s running time (5 points). Algorithms that are O(nlogn) or slower will be scored out of 10 points.

1. Let L be a list of numbers in non-decreasing order, and x be a given number. Describe an algorithm that counts the number of elements in L whose values are x (English description 5 points + Pseudocode 5 points). For example, if L = {1.3, 2.1, 2.1, 2.1, 2.1, 6.7, 7.5, 7.5, 8.6, 9.0} and x = 2.1 then the output of your algorithm should be 4. Your algorithm should run in O(log n) time (5 points).

**Note: This problem will be discussed in class. Algorithms that are O(n) or slower will be scored out of 10 points.**

**Section 2: Java Implementation**

1. Implement problem 3 in Java (30 points).

Note:

Find a file called Problem3.java in assignment 3 folder.

Complete the method of findmaxdiff().

Test your method in the main method.

1. Implement problem 4 in Java (30 points).

Note:

Find a file called Problem4.java in assignment 3 folder.

Complete the method of checksum().

Test your method in the main method provided.

1. Implement problem 5 in Java (30 points).

Note:

Find a file called Problem5.java in assignment 3 folder.

Complete the method of count().

Test your method in the main method provided.

**TURN-IN CHECKLIST:**

1. **Answers to Section 1 (.doc/.txt), and to Section 2 (all your source Code (.java files)). Remember to include your name, the date, and the course number in comments near the beginning of your code/report.**
2. **Create a folder and name it 'FirstName\_LastName\_assignment\_3'. In the newly created folder copy and paste your files (.doc/.txt/.java files). Then compress the folder, and push it to iLearn.**