CS 341: Algorithms @ Waterloo **Last Updated:** May 9, 2017
Homework 1 **Due Date: May 22, 2017, 11:00pm**

You are allowed to discuss with others but are not allowed to use any references other than the course notes and the three reference books. Please list your collaborators for each question. You must write your own solutions. See the course outline for the homework policy.

There are totally 55 marks (including the bonus). The full mark is 50. This homework is counted 5% of the course grade.

---

1. **Programming Problem: Counting Triangles** (20 marks)

   This question has two parts. The instructions for submitting your programs will be posted on piazza.

   (a) (10 marks) The first part is to write a program for fast polynomial multiplication, by extending the Karatsuba's $O(n^{1.59})$ algorithm for integer multiplication.

   Input: The first line has an integer n. The second line has $n+1$ integers $a_0, a_1, \ldots, a_n$, representing a degree n polynomial $f(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$. The third line has $n+1$ integers $b_0, b_1, \ldots, b_n$, representing a degree n polynomial $g(x) = b_0 + b_1x + b_2x^2 + \ldots + b_nx^n$. You can assume that $0 \leq n \leq 100000$ and $|a_i| \leq 100000$ for $0 \leq i \leq n$ and $|b_j| \leq 100000$ for $0 \leq j \leq n$.

   Output: One line with $2n+1$ numbers, the coefficients of the degree $2n$ polynomial $f(x) \cdot g(x)$.

   Sample Input:

   ```
   5
   0 20 139 -78 137 -11
   -7 88 923 -342 179 0
   ```

   Sample Output:

   ```
   0 -140 787 31238 113634 -103819 177040 -70969 28285 -1969 0
   ```

   (b) (10 marks) The second part is to apply the first part to solve a combinatorial counting problem. We have many sticks of different lengths. Our goal is to count how many ways to pick three sticks so that the three sticks can form a triangle. Three sticks can form a triangle if and only if the total length of any two sticks is strictly greater than the remaining one. For example, (4, 5, 8) can form a triangle but (1, 1, 2) and (1, 2, 4) cannot.

   Input: The first line has a positive integer n. The second line has n integers $a_1, \ldots, a_n$, where $a_i \geq 0$ is the number of sticks of length i. You can assume that $1 \leq n \leq 100000$ and $\sum_{i=1}^{n} a_i \leq 100000$.

   Output: The number of different triples that can form a triangle.

   Sample Input:

   ```
   7
   0 0 2 1 1 1 1
   ```

   Sample Output:

   ```
   16
   ```

   Explanation: There are two sticks of length 3 and one stick of length 4,5,6,7. The only invalid triples are (3, 3, 6), (3, 3, 7), (3, 4, 7) and (3, 4, 7), hence the answer is $\binom{6}{3} - 4 = 16$.

   Hint: Represent the input as a polynomial. Use polynomial multiplication to get useful information about various combinations of two sticks.

2. **Written Problem: Solving recurrences** (10 marks)

   Solve the following recurrence relations:

   (a) $T(n) = T(2n/3) + T(n/3) + n^2$.

   (b) $T(n) = \sqrt{n} \cdot T(\sqrt{n}) + n$.

   The base case is that constant size problems can be solved in constant time.

   You can either prove by induction or use the recursion tree method. Note that the master theorem as stated in L02 does not apply to these two problems.

3. **Written Problem: Maximum Sum Rectangle** (10 marks)

   Given an $n \times n$ 2D-array A where each entry $A[i][j]$ is an integer, our goal is to find a rectangle, specified by four indexes i_1, i_2, j_1, j_2, so as to maximize

$$\sum_{i=i_1}^{i_2} \sum_{j=j_1}^{j_2} A[i][j].$$

   Your task is to give an algorithm to solve the maximum sum rectangle problem. You will get full marks if the time complexity is $O(n^3 \log n)$ and the proofs are correct.

-9	-5	4	1	-7
10	-6	8	-10	3
3	12	-9	2	-4
-8	-4	-7	-5	2
9	2	5	3	-1

Figure 1: These are the two possible solutions with maximum total sum 19.

4. **Written Problem: Binary Tree Reconstruction**

(10 marks) Given a binary tree, the *pre-order* traversal is obtained by first printing the root, and then the left subtree recursively and then the right subtree recursively. The *in-order* traversal is obtained by first printing the left subtree recursively, then the root and then the right subtree recursively. See the following picture for an example and see the wikipedia page for "Tree Traversal" for the definition.

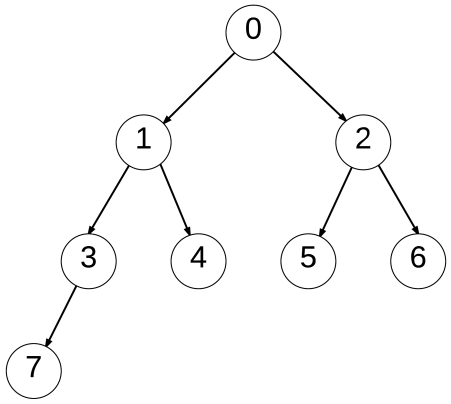

Figure 2: The pre-order traversal of the tree is 0 1 3 7 4 2 5 6, and the in-order traversal is 7 3 1 4 0 5 2 6.

Given a pre-order traversal, we could not uniquely recover a binary tree, as there are multiple binary trees with the same pre-order traversal. However, if we are given both the pre-order traversal and the in-order traversal of a binary tree, then we can uniquely recover the binary tree.

In this question, your task is to give an algorithm to reconstruct the binary tree given its pre-order traversal and in-order traversal. You will get full marks if the time complexity of your algorithm is $O(n^2)$ and the proofs are correct, where n is the number of nodes in the tree.

Bonus: (5 marks) You will get the bonus marks if the time complexity of your algorithm is $O(n)$ and the proofs are correct.