CS 341: Algorithms @ Waterloo **Last Updated:** Jan 16, 2023
Homework 1 **Due Date: Jan 27, 2022, 11pm**

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 57 marks (including the bonus). The full mark is 50 (extra marks above 50 will not be carried over). This homework is counted 8% of the course grade.

---

1. **Programming Problem: Polynomial Multiplication** (15 marks)

   The instructions for submitting your programs will be posted on piazza.

   You are asked 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 $f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$. The third line has $n+1$ integers $b_0, b_1, \ldots, b_n$, representing a degree n polynomial $g(x) = b_0 + b_1 x + b_2 x^2 + \ldots + b_n x^n$. 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
   ```

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

   Solve the following recurrence relations:

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

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

3. **Written Problem: Merging Sorted Lists** (12 marks)

We start with a definition. We consider a full binary tree T with k leaves, where the ith leaf has *weight* n_i for all i. The weight of each internal node is defined as the sum of the weights of its two children nodes, and the weight $W(T)$ of the whole tree is the sum of the weights of its internal nodes.

(a) (2 marks) We use this to analyse a divide-and-conquer algorithm to merge k sorted lists. Here, we will do the simplifying assumption that merging two lists of lengths n and n' takes $n + n'$ operations. We consider the following algorithm to merge lists $L_1, \ldots, L_k$:

- if $k = 1$, return L_1
- else,
  - choose $k' \in \{1, \ldots, k-1\}$
  - call the algorithm recursively to merge $L_1, \ldots, L_{k'}$ into a list L and $L_{k'+1}, \ldots, L_k$ into a list L'
  - return the list obtained by merging L and L'

Give the cost of this algorithm in terms of weight of the recursion tree.

(b) (5 marks) Prove that given integers $\ell_1, \ldots, \ell_k$ with $\sum_{i=1}^{k} 1/2^{\ell_i} \leq 1$, you can build a full binary tree with leaves labelled by $1, \ldots, k$ and with the ith leaf of depth at most ℓ_i for all i.

(c) (3 marks) Use part (b) or otherwise, prove that given positive $n_1, \ldots, n_k$, you can build a full binary tree T with k leaves, with leaves having weights $n_1, \ldots, n_k$, and with $W(T) \leq N(H(n_1, \ldots, n_k) + 1)$, where $N = n_1 + \cdots + n_k$ and $H(n_1, \ldots, n_k)$ is the *entropy* defined as

$$H(n_1, \ldots, n_k) = -\sum_{1 \leq i \leq k} \frac{n_i}{N} \log_2\left(\frac{n_i}{N}\right) = \sum_{1 \leq i \leq k} \frac{n_i}{N} \log_2\left(\frac{N}{n_i}\right).$$

You can start by rewriting the weight $W(T)$ of a full binary tree in terms of the weights of the leaves and their respective depths.

(d) (2 marks) Apply part (c) to the case where you merge lists of same lengths $n_1 = \cdots = n_k$: what cost do you get? Same question with $n_1 = n_2 = 1, n_3 = 2, n_4 = 4, \ldots, n_k = 2^{k-2}$.

4. **Written Problem: Finding Maximum Space** (15+5 marks)

This question has two parts.

(a) (10+5 marks) Imagine an advertising company would like to post a huge poster in Toronto harbour front. There are n consecutive buildings there, with positive height $h_1, \ldots, h_n$ and unit width as shown in the figure. Design a divide and conquer algorithm to find the maximum rectangular space to post their poster. Stated mathematically, find $1 \leq i \leq j \leq n$ to maximize $(j - i + 1) \cdot \min_{i \leq k \leq j}\{h_k\}$.

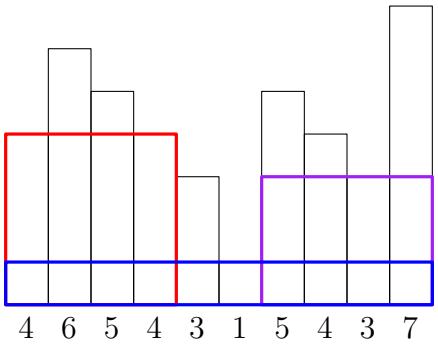

Figure 1: These are three possible solutions. The largest one has area $4 \times 4 = 16$.

You will get full marks if the time complexity is $O(n \log n)$ and the proofs are correct.
Bonus (5 marks): Provide an algorithm with time complexity $O(n)$ with correct proofs.

(b) (5 marks) Imagine the government would like to build a huge park in the city. The city is an $n \times n$ grid, with some units occupied. Use (a) or otherwise, design an algorithm to find the maximum (axis-aligned) rectangular unoccupied space to build the park.

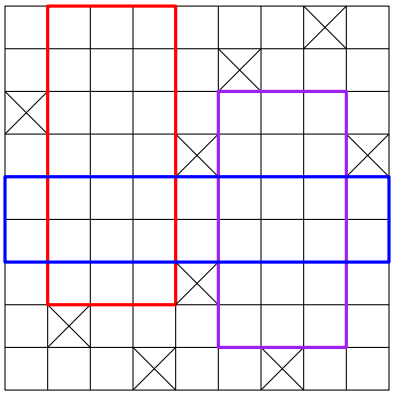

Figure 2: These are three possible solutions. The largest one has area $7 \times 3 = 21$.

You will get full marks if the time complexity is $O(n^2)$ and the proofs are correct. You can assume that there is an $O(n)$ time algorithm for part (a).