Homework 3.
Due: Monday, September 19, 2022 before 8am EDT.

[DPV] Practice Problems

These are practice problems to help you to become more familiar with the topic, these problems will not be graded. It is not compulsory to finish these problems.

DPV Problems 2.7 (Sum of roots of unity), 2.8, 2.9(a) (FFT practice)

Practice Problem (Types of binary search)

Let A be an arrray of n different elements. By sorted array we mean sorted in non-decreasing order.

(a) Consider $A = \{10, 23, 36, 47, 59, 64, 71, 82, 95, 100, 116, 127, 138, 141, 152, 163\}$. We want to check if the number 36 is an element of A. Explain how the binary search works to find this element.

(b) Design an $O(\log(n))$ algorithm to find the smallest missing natural number in a given sorted array. The given array only has natural numbers. For example, the smallest missing natural number from $A = \{3, 4, 5\}$ is 1 and from $A = \{1, 3, 4, 6\}$ is 2.

See next page for homework problems.

Problem 1 (Integer multiplication using FFT)

(a) Given an $n-$bit integer number $a = a_{n-1}a_{n-2}\dots a_0$ define a polynomial $A(x)$ satisfying $A(2) = a$.

(b) Given two $n-$bit integers a and b, give an algorithm to multiply them in $O(n\log(n))$ time. Use the FFT algorithm from class as a black-box (i.e. don't rewrite the code, just say run FFT on ...). Explain your algorithm in words and analyze its running time. (*Think/investigate why this is not enough to claim a fast multiplication algorithm*).

Problem 2 (Selling stocks)

Certain stock has been fluctuating a lot recently, and you have a share of it. You keep track of its selling value for N consecutive days, and kept those numbers in an array $S = [s_1, s_2, \dots, s_N]$. In order to make good predictions, you decide if a day i is good by counting how many times in the future this stock will sell for a price less than $S[i]$. Design an algorithm that takes as input the array S and outputs and array G where $G[i]$ is the number of days after i that your stock sold for less than $S[i]$.

Examples:
$S = [5, 2, 6, 1]$ outputs $[2, 1, 1, 0]$.
$S = [1]$ outputs $[0]$.
$S = [5, 5, 7]$ outputs $[0, 0, 0]$.

Describe your algorithm with words (do not use pseudocode) and explain why your algorithm is correct. Give the time complexity (using the Master Theorem when applicable).