

1. [20 marks] You are given two polynomials,

$$P_A(x) = A_0 + A_3x^3 + A_6x^6$$

and

$$P_B(x) = B_0 + B_3x^3 + B_6x^6 + B_9x^9$$

where all A_i 's and all B_j 's are large numbers. Multiply these two polynomials using only 6 large number multiplications.

2. (a) [5 marks] Multiply two complex numbers $(a + ib)$ and $(c + id)$ (where a, b, c, d are all real numbers) using only 3 real number multiplications.
- (a) [5 marks] Find $(a + ib)^2$ using only two multiplications of real numbers.
- (b) [10 marks] Find the product $(a + ib)^2(c + id)^2$ using only five real number multiplications.
3. (a) [2 marks] *Revision:* Describe how to multiply two n -degree polynomials together in $O(n \log n)$ time, using the Fast Fourier Transform (FFT). You do not need to explain how FFT works – you may treat it as a black box.
- (b) In this part we will use the Fast Fourier Transform (FFT) algorithm described in class to multiply multiple polynomials together (not just two). Suppose you have K polynomials P_1, \dots, P_K so that

$$\text{degree}(P_1) + \dots + \text{degree}(P_K) = S$$

- (i) [6 marks] Show that you can find the product of these K polynomials in $O(KS \log S)$ time.
Hint: How many points do you need to uniquely determine an S -degree polynomial?
- (ii) [12 marks] Show that you can find the product of these K polynomials in $O(S \log S \log K)$ time.
Hint: consider using divide-and-conquer; a tree which you used in the previous assignment might be helpful here as well. Also, remember that if x, y, z are all positive, then $\log(x + y) < \log(x + y + z)$
4. Let us define the Fibonacci numbers as $F_0 = 0$, $F_1 = 1$ and $F_n = F_{n-1} + F_{n-2}$ for all $n \geq 2$. Thus, the Fibonacci sequence looks as follows: 0, 1, 1, 2, 3, 5, 8, 13, 21, ...
- (a) [5 marks] Show, by induction or otherwise, that

$$\begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n$$

for all integers $n \geq 1$.

- (b) [15 marks] Hence or otherwise, give an algorithm that finds F_n in $O(\log n)$ time.

5. Your army consists of a line of N giants, each with a certain height. You must designate precisely $L \leq N$ of them to be leaders. Leaders **must** be spaced out across the line; specifically, **every pair** of leaders **must** have **at least** $K \geq 0$ giants standing **in between them**. Given N, L, K and the **heights** $H[1..N]$ of the giants **in the order that they stand in the line** as input, **find the maximum height of the shortest leader among all valid choices of L leaders**. We call this the *optimisation* version of the problem.



For instance, suppose $N = 10, L = 3, K = 2$ and $H = [1, 10, 4, 2, 3, 7, 12, 8, 7, 2]$. Then among the 10 giants, you must choose 3 leaders so that each pair of leaders has at least 2 giants standing in between them. The best choice of leaders has heights 10, 7 and 7, with the shortest leader having height 7. This is the best possible for this case.



- (a) [**8 marks**] In the *decision* version of this problem, we are given an additional integer T as input. Our task is to **decide if there exists** some valid choice of leaders satisfying the constraints whose shortest leader has height no less than T .
Give an algorithm that solves the decision version of this problem in $O(N)$ time.
- (b) [**12 marks**] Hence, show that you can solve the optimisation version of this problem in $O(N \log N)$ time.