CMPSCI 611
Advanced Algorithms
First Midterm Exam Spring 2024

Question 1. For each of the following statements, indicate whether they are TRUE or FALSE by circling the appropriate answer. No justification is required. Each part is worth 2 points.

1. $5n^2 + 10 = O(n^2)$.
   Answer: TRUE.

2. A minimum spanning tree has no cycles.
   Answer: TRUE.

3. Every subset system satisfies the cardinality property.
   Answer: FALSE.

4. Let G be a connected, undirected graph with n nodes. If all edges have length 1, then the shortest path between any two nodes is at most $n - 1$.
   Answer: TRUE.

Question 2. No justification required for your answers. Each part is worth 2 points.

1. We want to sort a list by performing "prefix reversals" operations, i.e., at each step we specify a value k and then reverse the ordering of the first k entries of the list. What is the minimum number of prefix reversals needed to sort $[7, 8, 6, 5, 1, 2, 3, 4]$. **Hint:** We know from homework that it is at most 2×8 but the answer here is less than that.
   Answer: 3, i.e.,

   $$[7, 8, 6, 5, 1, 2, 3, 4] \to [8, 7, 6, 5, 1, 2, 3, 4] \to [4, 3, 2, 1, 5, 6, 7, 8] \to [1, 2, 3, 4, 5, 6, 7, 8]] \ .$$

2. Let $E = \{e_1, e_2, e_3\}$ and $\mathcal{I} = \{\{\}, \{e_1\}, \{e_1, e_2\}, \{e_3\}\}$. What set in $\mathcal{I}$ should you remove if you wanted $(E, \mathcal{I})$ to be a subset system that satisfied the exchange property.
   Answer: Remove $\{e_1, e_2\}$.

3. Let M be the adjacency matrix of the graph with three nodes v_1, v_2, v_3 and two edges (v_1, v_2) and (v_2, v_3). Write out M^2, i.e., the square of the adjacency matrix.
   Answer: $M = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$ and $M^2 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 0 & 1 \end{pmatrix}$.

4. If $T(n) = T(n-1) + n^2$ for $n > 1$ and $T(1) = 1$, then $T(n) = \Theta(n^x)$ for what value of x?
   Answer: $T(n) = T(n-1) + n^2 = T(n-2) + (n-1)^2 + n^2 = \ldots = \sum_{i=1}^{n} i^2 = \Theta(n^3)$. To see the last step note that $\sum_{i=1}^{n} i^2 > \sum_{i=n/2}^{n} i^2 > (n/2)^3$.

Question 3. A company wants to buy some of the existing n gas stations along a highway. For $i \in \{1, 2, \ldots, n\}$, let p_i be the profit they will make from owning the ith gas station. You may assume all p_i values are positive. A state law says forbids the same company from owning two gas stations that are next to each other, e.g., if the company owns the 5th gas station, they can't also own the 4th or 6th gas station.

1. (2 points) Suppose $n = 5$ and $p_1 = 5, p_2 = 3, p_3 = 2, p_4 = 4$ and $p_5 = 1$. What is the maximum possible profit that can be achieved?

   Answer: Pick the first and fourth gas station to get profit $5 + 4 = 9$.

2. (2 points) Consider the algorithm that, at each step, buys the gas station that has max profit amongst those that are not next to a gas station that was already purchased. Suppose $n = 3$. Give examples of values for p_1, p_2, p_3 such that this algorithm would *not* return the best solution.

   Answer: Let $p_1 = 4, p_2 = 5, p_3 = 3$. Then the algorithm just picks the second gas station and gets profit 5 whereas the optimal solution is $4 + 3 = 7$.

3. (4 points) In this part you should not make any assumptions about n or the profits. Design a dynamic programming algorithm that finds the maximum total profit that is possible. Analyze the running time and justify correctness.

   Answer: Let $D(j)$ be the max profit when may only choose amongst the first j gas stations. $D(1) = p_1$ and $D(2) = \max(p_1, p_2)$ because all profits are positive. For $j \geq 3$, $D(j) = \max(p_j + D(j-2), D(j-1))$ because the best subset amongst the first j either includes j (and the max profit is $p_j + D(j-2)$ because you can't pick $j-1$) or it doesn't (and you're free to pick an subset of the first $j-1$ gas stations). This gives the algorithm:

   1. Set $D(1) = p_1$ and $D(2) = \max(p_1, p_2)$
   2. For $j = 3, \ldots, n$: Set $D(j) = \max(p_j + D(j-2), D(j-1))$
   3. Return $D(n)$.

   Note that the $D(j-2)$ and $D(j-1)$ are already computed when we set $D(j)$. There are n entries of D and each requires $O(1)$ time. Hence the running time is $O(n)$.

Question 4. We now consider a simpler (but less efficient) algorithm for multiplying two polynomials than the one discussed in class. Assume n is a power of two. The input is the coefficients of the following polynomials $a(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$ and $b(x) = b_0 + b_1 x + \ldots + b_{n-1} x^{n-1}$ and the goal is to compute the coefficients of the polynomial $c(x) = a(x)b(x)$. Define

$$\begin{aligned} a_L(x) &= a_0 + a_1 x + \ldots + a_{n/2-1} x^{n/2-1} \\ a_H(x) &= a_{n/2} + a_{n/2+1} x + \ldots + a_{n-1} x^{n/2-1} \\ b_L(x) &= b_0 + b_1 x + \ldots + b_{n/2-1} x^{n/2-1} \\ b_H(x) &= b_{n/2} + b_{n/2+1} x + \ldots + b_{n-1} x^{n/2-1} \end{aligned}$$

and note that $a(x) = a_L(x) + x^{n/2} a_H(x)$ and $b(x) = b_L(x) + x^{n/2} b_H(x)$.

1. (2 points) Write $c(x)$ in terms of $a_L(x), a_H(x), b_L(x), b_H(x), x^{n/2}$, and x^n. Your answer should be written as the sum of four terms.

Answer:

$$\begin{aligned} c(x) &= (a_L(x) + x^{n/2}a_H(x))(b_L(x) + x^{n/2}b_H(x)) \\ &= a_L(x)b_L(x) + x^{n/2}a_L(x)b_H(x) + x^{n/2}a_H(x)b_L(x) + x^n a_H(x)b_H(x) \end{aligned}$$

.

2. (2 points) Write $c(x)$ in terms of $p_1(x), p_2(x), p_3(x)$, $x^{n/2}$, and x^n where

$$p_1(x) = a_L(x)b_L(x) \qquad p_2(x) = a_H(x)b_H(x)$$

$$p_3(x) = (a_L(x) + a_H(x))(b_L(x) + b_H(x)) - p_1(x) - p_2(x)$$

Hint: First simplify $p_3(x)$.
Answer: $p_3(x) = a_L(x)b_H(x) + a_H(x)b_L(x)$ and so $c(x) = p_1(x) + x^{n/2}p_3(x) + x^n p_2(x)$.

3. (4 points) Write a divide and conquer algorithm for computing $c(x)$ based on your answers above. Analyze the running time and justify correctness.
Answer: Consider the following algorithm:

Multiply$(a_0, \ldots, a_{n-1}, b_0, \ldots, , b_{n-1})$:

(a) If $n = 1$, output $[a_0 b_0, 0]$

(b) $[p_{1,0}, \ldots, p_{1,n-1}] \leftarrow$ Multiply$(a_0, \ldots, a_{n/2-1}, b_0, \ldots, b_{n/2-1})$

(c) $[p_{2,0}, \ldots, p_{2,n-1}] \leftarrow$ Multiply$(a_{n/2-1}, \ldots, a_{n-1}, b_{n/2-1}, \ldots, b_{n-1})$

(d) $[p_{3,0}, \ldots, p_{3,n-1}] \leftarrow$ Multiply$(a_0+a_{n/2}, \ldots, a_{n/2-1}+a_{n-1}, b_0+b_{n/2}, \ldots, b_{n/2-1}+b_{n-1})$
$- [p_{1,0}, \ldots, p_{1,n-1}] - [p_{2,0}, \ldots, p_{2,n-1}]$

(e) Return

$$[p_{1,0}, \ldots, p_{1,n-1}, \underbrace{0, \ldots, 0}_{n}] + [\underbrace{0, \ldots, 0}_{n}, p_{2,0}, \ldots, p_{2,n-1}] + [\underbrace{0, \ldots, 0}_{n/2}, p_{3,0}, \ldots, p_{3,n-1}, \underbrace{0, \ldots, 0}_{n/2}]$$

The running time is $O(n^{\log_2 3})$ because $T(n) = 3T(n/2) + O(n)$ and $T(1) = \Theta(1)$. The correctness for the base case follows since if $a(x) = a_0$ and $b(x) = b_0$ then $a(x)b(x) = a_0 b_0$. Assuming the algorithm computes the coefficients correctly when multiplying together degree $n/2 - 1$ polynomials the correctness follows because $c(x) = p_1(x) + x^{n/2}p_3(x) + x^n p_2(x)$. Note that inserting $n/2$ or n zero's at the start of the array is equivalent to multiplying the polynomial by $x^{n/2}$ or x^n.