1. Give an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers.

2. Find an optimal parenthesizing of a matrix-chain product whose sequence of dimensions is [50,10,3,12,50,5,6]

3. You are driving from Tehran to Shiraz. There are gas stations along the way at distance $x_1,x_2,...,x_n$ from Tehran. Because of different wait times and pump speeds, filling up at a gas station takes minutes (the gas costs the same everywhere, so we ignore its cost). Your car can hold enough gas to go miles, and you start with a full tank of gas. If you decide to stop at a gas station, you have to fill your entire tank up. Give a dynamic programming algorithm that finds where you should stop to spend the minimum amount of time at gas stations during your trip.

4. Suppose we want to make change for n cents, using the least number of coins of denominations 1, 10, and 25 cents. Consider the following greedy strategy:
Suppose the amount left to change is m; take the largest coin that is no more than m; subtract this coin's value from m, and repeat.
Either give a counterexample to prove that this algorithm can output a non-optimal solution or prove that this algorithm always outputs an optimal solution.

5. Suppose that a data file contains a sequence of 8-bit characters such that all 256 characters are about equally common: the maximum character frequency is less than twice the minimum character frequency. Prove that Huffman coding in this case is no more efficient than using an ordinary 8-bit fixed-length code.

6. Suppose that you and your friends have decided to hike Mount Everest this summer. You want to hike as much as possible per day but, for obvious reasons, not after dark. On a map, you have identified a large set of good stopping points for camping, and you are considering the following system for deciding when to stop for the day. Each time you come to a potential stopping point, you determine whether you can make it to the next one before nightfall. If you can make it, then you keep hiking; otherwise, you stop. Despite many significant drawbacks, you claim this system does have one good feature. "Given that we are only hiking in the daylight, it minimizes the number of camping stops we have to make." Is this true? The proposed system is a greedy algorithm, and we wish to determine whether it minimizes the number of stops needed. We can state the question as follows: Is your greedy algorithm - hiking as long as possible each day - optimal, in the sense that it finds a valid set whose size is as small as possible? If so, prove it.