ASSIGNMENT - 3
Analysis and Design of Algorithms - CSE222
Each question carries 10 points.

Question 1.

1.1. Consider the Change Problem in Pound sterling UK. The input to this problem is an integer U. The output should be the minimum cardinality collection of coins required to make U shillings of change (that is, you want to use as few coins as possible). In the UK, the coins are worth 1, 5, 10, 20, 25, 50 Shillings. Assume that you have an unlimited number of coins of each type. Formally prove or disprove that the greedy algorithm (that takes as many coins as possible from the highest denominations) correctly solves the Change Problem. So for example, to make change for 234 Shillings the greedy algorithms would take four 50 shilling coins, one 25 shilling coin, one 5 shilling coin, and four 1 shilling coins.

1.2. Consider the Change Problem in Binaryland The input to this problem is an integer U. The output should be the minimum cardinality collection of coins required to make U nibbles of change (that is, you want to use as few coins as possible). In Binaryland the coins are worth 1, 2, 2^2 , 2^3 , . . . , 2^{1000} nibbles. Assume that you have an unlimited number of coins of each type. Prove or disprove that the greedy algorithm (that takes as many coins of the highest value as possible) solves the change problem in Binaryland.

Question 2.

You wish to drive from point X to point Y along a highway minimizing the time that you are stopped for gas. You are told beforehand the capacity Z of your gas tank in liters, your rate F of fuel consumption in liters/kilometer, the rater in liters/minute at which you can fill your tank at a gas station, and the locations X = x_1 , . . . , Y = x_n of the gas stations along the highway. So if you stop to fill your tank from 2 liters to 8 liters, you would have to stop for 6/r minutes. Consider the following two algorithms:

2.1. Stop at every gas station, and fill the tank with just enough gas to make it to the next gas station.

2.2. Stop if and only if you don't have enough gas to make it to the next gas station, and if you stop, fill the tank up all the way.

For each algorithm either prove or disprove that this algorithm correctly solves the problem. Your proof of correctness must use an exchange argument.

Question 3.
The Amit food company produces a large variety of different dinner menu items. Unfortunately, they can only produce their foods in limited quantities, so they often run out of popular items, making customers sad. To minimize sadness, Amit food is implementing a sophisticated dinner-ordering system. Customers text in their acceptable choices before dinner time. Then they can use an algorithm to pre-assign dinners to customers. Customers who do not get one of their choices should receive a $5 voucher. Amit food would like to minimize the number of vouchers they give out.

Give an efficient algorithm for Amit food to assign dinners to customers. In general, suppose that, on a given day, Amit food has produced m types of food items b_1 , . . . , b_m , and the quantity of each type of food item b_j is exactly q_j . Suppose that n customer a_1 , . . . , a_n text in their preferences, where each customer ai submits a set A_i of one or more acceptable dinner choices. The algorithm should assign each customer either one of his/her choices or a $5 voucher. The number of vouchers should minimum.

Question 4.
You are given a source string ST[0 . . n − 1] of length n, consisting of symbols x and y. Suppose further that you are given a pattern string PT[0 . . m − 1] of length m « n, consisting of symbols x, y, and ∗, representing a pattern to be found in string ST. The symbol ∗ is a “wild card” symbol, which matches a single symbol, either x or y. The other symbols must match exactly. The problem is to output a sorted list M of valid “match positions”, which are positions j in ST such that pattern PT matches the substring ST[j . . j + |PT| − 1]. For example, if ST = x y x y y x y and PT = x y ∗, then the output M should be [0, 2].

4.1. Describe a straightforward, naive algorithm to solve the problem. Your algorithm should run in time O(nm).

4.2. Give an algorithm to solve the problem by reducing it to the problem of polynomial multiplication. Specifically, describe how to convert strings ST and PT into polynomials such that the product of the polynomials allows you to determine the answer M. Give examples to illustrate your polynomial representation of the inputs and your way of determining outputs from the product, based on the example ST and PT strings given above.

Question 5.
Consider an integer grid. At all vertices of this grid (intersection points of a horizontal and vertical line) a grain of sugar is placed. An ant is placed at one of the grid vertices. The ant walks in a straight line from its initial point to a vertex with a grain of sugar. It then walks again in a straight line to the next vertex with sugar, and so on and returns to its initial position such that the closed path it takes does not cross itself, and between grid points, it walks along a straight line (note that this line need not be vertical or horizontal). Suppose the ant eats k grains of sugar and leaves l grains of sugar in the interior of this closed path uneaten. What is the area enclosed by the closed path?

Question 6. Given n numbers in an array, we want to find the min and max number in this array. Design a divide-and-conquer algorithm to compute both these quantities in $< 2n$ comparisons. Write the recurrence for your algorithm, and solve the resulting recurrence to derive the worst-case number of comparisons your algorithm will take.