1. We consider the following problem:

**INPUT:** A list \( I_1 = (a_1, b_1), \ldots, I_n = (a_n, b_n) \) of intervals on the real line. For simplicity assume that all \( 2n \) numbers are distinct.

**OUTPUT:** A maximum cardinality collection \( S \) of disjoint intervals.

That is, you want to pick as many intervals as possible, with the proviso that no pair of picked intervals overlap. We consider greedy algorithms of the following form:

(a) From the remaining intervals, pick the interval \( I_j \) with some special property.
(b) Add \( I_j \) to the solution \( S \)
(c) Remove \( I_j \), and any intervals that intersect \( I_j \) from future consideration.
(d) If there are interval left then repeat.

One can get different algorithms depending on what the special property is. Here we consider two such algorithms:

**LargestLeft Algorithm:** Here \( I_j \) is picked to be the interval with the largest left endpoint. That is, the interval where \( a_j \) is the largest.

**LowestOverlap Algorithm:** Here \( I_j \) is picked to be the interval that intersects a minimal number of other remaining intervals.

Show that one of these two greedy algorithms is not correct (5 points). For full credit, it should be clear from your writeup what one needs to provide to demonstrate the incorrectness of an algorithm. Prove that the other greedy algorithm is correct using an exchange argument (15 points). For full credit, it should be clear from your writeup that you know what an exchange argument is, and why it establishes the correctness for the algorithm in question.

2. (20 points) Consider the following problem.

**INPUT:** Positive integers \( r_1, \ldots, r_n \) and \( c_1, \ldots, c_n \).

**OUTPUT:** An \( n \) by \( n \) matrix \( A \) with 0/1 entries such that for all \( i \) the sum of the \( i \)th row in \( A \) is \( r_i \) and the sum of the \( i \)th column in \( A \) is \( c_i \), if such a matrix exists. If such a matrix doesn’t exist, you can output anything.

Think of the problem this way. You want to put pawns on an \( n \) by \( n \) chessboard so that the \( i \)th row has \( r_i \) pawns and the \( i \)th column has \( c_i \) pawns.

Consider the following greedy algorithm that constructs \( A \) row by row. Assume that the first \( i - 1 \) rows have been constructed. Let \( a_j \) be the number of 1’s in the \( j \)th column in the first \( i - 1 \) rows. Now the \( r_i \) columns with with maximum \( c_j - a_j \) are assigned 1’s in row \( i \), and the rest of the columns are assigned 0’s. That is, the columns that still needs the most 1’s are given 1’s.

Formally prove that this algorithm is correct using an exchange argument.