1. An instance of set cover consists of a family of \( n \) subsets \( S = \{S_1, \ldots, S_n\} \) of universe \( M = \{1, \ldots, m\} \) whose union is all of \( M \). A subfamily \( S' \subseteq S \) covers the universe if its union is the entire \( M \). Our task is to find the smallest subfamily of \( S \) whose union is still \( M \). Your task is to build a primal-dual framework for this problem:

(a) What are the indicator variables of your ILP formulation of this problem.
(b) What is the objective function of your formulation.
(c) What are the constraints of this formulation.
(c) State the relaxation of ILP to LP.
(d) Formulate the dual LP for set cover problem.
(e) What is the interpretation of dual variables for set cover problem.

2. Consider the following geometric problem arising in integrated circuit design:

- **Input**: A collection of rectangles \( I = \{R_1, \ldots, R_n\} \) in the plane such that each rectangle is aligned with the axes (all sides are horizontal or perpendicular). Rectangles may overlap.
- **Feasible solution**: A collection of points \( P = \{p_1, \ldots, p_m\} \) such that each rectangle in \( I \) contains at least one point from \( P \).
- **Goal**: Minimize the number of points \( |P| \)

Provide the best (polynomial time) approximation algorithm you can for this problem.

3. The following problem arises in telecommunications networks. The network consists of a cycle on \( n \) nodes, numbered 0 through \( n - 1 \) clockwise around the cycle. Some set \( C \) of calls is given, each call is a pair \((i, j)\) originating at node \( i \) and destined to node \( j \). The call can be routed either clockwise or counterclockwise around the ring. The objective is to route the calls so as to minimize the total load on the network. The load \( L_i \) on link \((i, i + 1 (\text{mod} n))\) is the number of calls routed through link \((i, i + 1 (\text{mod} n))\), and the total load is \( \max_{1 \leq i \leq n} L_i \). Formulate this problem as an ILP and give a 2approximation algorithm for the ring loading problem.

4. Consider the problem of scheduling \( n \) jobs on \( m \) identical machines, where \( p_j \) is the amount of processing time required for job \( j \). The goal is to minimize the maximum completion time \( C_j \) over all jobs \( j \). It is not hard to show that a simple “list scheduling” algorithm, which takes the list of jobs, and schedules the next job in the list on the least loaded machine, is a 2approximation algorithm. Suppose we begin by sorting the list so that \( p_1 \geq p_2 \geq \ldots \geq p_n \). Prove that this gives a \( \frac{4}{3} \)approximation algorithm. (Hint: Suppose job \( k \) finishes last. What can we say if \( p_k \leq \frac{1}{3} \text{OPT} / 3 \)? Now suppose \( p_k > \frac{1}{3} \text{OPT} / 3 \). Argue that the performance of the algorithm on jobs \( J_1, J_2, \ldots, J_k \) will be no better than that on \( J_1, J_2, \ldots, J_n \). Now argue that the algorithm does well in this case when \( p_j > \frac{1}{3} \text{OPT} / 3 \) for all jobs \( j \).