1. (35 pts) An exhibition is going on in a town today from $t = 0$ (9 am) to $t = 720$ (6 pm), and celebrities will attend!! There are n celebrities $C_1, \ldots, C_n$ and each celebrity C_j will attend during a time interval $I_j : [\ell_j, u_j]$ where in $0 \leq \ell_j < u_j \leq 720$. Note: the ends of the interval are *inclusive*. An advertising company for high-end luxury goods wants to broadcast its ad on a large screen television in the exhibition, multiple times during the day. In particular, each celebrity must see the ad but the company also wants to minimize the number of times the ad must be shown.

   For example:

Celebrity	$[\ell, u]$
1	$[2, 53]$
2	$[5, 50]$
3	$[5, 98]$
4	$[102, 150]$
5	$[120, 186]$
6	$[85, 190]$

   Then, if the ad is shown at times $t_1 = 50$ and $t_2 = 150$, then all 6 of the celebrities will see the ad.

   (a) Greedy algorithm $\mathcal{A}$ selects a time instance when the maximum number of celebrities are present simultaneously. An ad is scheduled at this time and the celebrities covered by this ad are then removed from further consideration. The algorithm $\mathcal{A}$ is then applied recursively to the remaining celebrities.
   Give an example where this algorithm shows more ads than is necessary.

   Solution: A counterexample is as follows:
   $C_1 : [0, 2], C_2 : [2, 10], C_3 : [2, 10], C_4 : [2, 10], C_5 : [10, 18], C_6 : [10, 18], C_7 : [10, 18], C_8 : [18, 20]$.
   The greedy algorithm will first select time $t = 10$ since 6 celebs are simultaneously present at that time. Deleting all the intervals covered will delete $C_2, \ldots, C_7$, leaving us C_1, C_8. The algorithm then selects $t = 1, t = 19$ to cover C_1, C_8. This means the ad is shown 3 times. However, the ad just needs to be shown twice: $t = 2$ and $t = 18$.

   (b) Let C_j represent the celebrity who *leaves* first and let $[\ell_j, u_j]$ be the time interval for C_j. Suppose we have some solution $t_1, t_2, \ldots, t_k$ for the ad times that cover all celebrities. Let t_1 be the earliest ad time.
   Prove the following facts for the earliest scheduled at t_1. For each part, your

proof must clearly spell out the argument. Overly long explanations or proofs by examples will receive no credit. You may as the course staff for help with the wordings of your proofs.

S1. Prove $t_1 \leq u_j$. (Three sentences. Hint: proof by contradiction.)
Solution: It is easy to see that $t_1 \leq u_j$. Otherwise, if $t_1 > u_j$, then $t_2, \ldots, t_k > u_j$ as well. Thus, the celebrity C_j will not be covered by any ad.

S2. If $t_1 \leq \ell_j$, then t_1 can be deleted, and the remaining ads still form a valid solution. (Five sentences. Hint: suppose deleting t_1 leaves a celebrity uncovered, when should that celebrity have arrived and left? Prove a contradiction.)
Solution: If $t_1 < l_j$, then suppose deleting t_1 is not a valid solution in that it leaves a celebrity C_k uncovered. Now, $C_k \neq C_j$ since t_1 does not cover C_j. Since t_1 covers C_k therefore C_k must have started before C_j. However, C_j is the first celebrity to leave. Therefore, C_k leaves after C_j. Let t_i be the time of the ad that covers C_j. This will also cover C_k since C_k arrives before and leaves after.

S3. If $t_1 < u_j$, then t_1 can be modified to be equal to u_j, while still remaining a valid solution. (Three sentences. Hint: suppose setting $t_1 := u_j$ leaves a celebrity uncovered, then when should that celebrity have arrived and left? Prove a contradiction.)
Solution: If $t_1 < u_j$, then let us modify the solution to have $t_1 = u_j$. Suppose doing so leaves C_k uncovered. However, this means that $t_1 < u_k < u_j$. But that violates the assumption that C_j was the first celebrity to leave.

(c) Use the insight from (1b) to design a greedy algorithm that is optimal.
(i) Write pseudocode for your algorithm.
(ii) Prove that your algorithm is correct and give its running time complexity.
(iii) Demonstrate the solution your algorithm yields when applied to the $n = 6$ example above.

Solution: The algorithm is as follows:

- Sort all the celebrities in increasing order of the leaving times.
- While the list is not empty:
  - Remove the first interval $[l_i, u_i]$ from the list.
  - Add $t_i = u_i$ as a time when an ad is shown.
  - Remove all intervals that contain t_i from the list.