DS 320 Algorithms for Data Science	**Lecture #7 Worksheet**
Spring 2026	Prof. Kira Goldner and Jeffrey Considine

Greedy II: Interval Scheduling

Suppose you are given n jobs to schedule on a machine. Each job i (where $i \in \{1, \ldots, n\}$) has a start time $s(i)$ and a finish time $f(i)$. You would like to schedule *as many* jobs as possible given that the machine can only process one job at a time, and the jobs must run from their start time to finish time uninterrupted to be processed. That is, the machine cannot process two jobs that overlap.

What *greedy* algorithm should you use to schedule the jobs? By what metric is it greedy? (See **Step 2**.)

Prove that your algorithm is optimal by a **Greedy-Stays-Ahead** proof.

Step 1: Define your solutions. Describe the form your greedy solution takes, and what form some other solution takes (possibly the optimal solution). For example, let A be the solution constructed by the greedy algorithm, and let O be a (possibly optimal) solution.

Step 2: Find a measure. Find a *measure* by which greedy stays ahead of the other solution you chose to compare with. Let $a_1, \ldots, a_k$ be the first k measures of the greedy algorithm, and let $o_1, \ldots, o_m$ be the first m measures of the other solution ($m = k$ sometimes).

Step 3: Prove greedy stays ahead. Show that the partial solutions constructed by greedy are always just as good as the initial segments of your other solution, based on the measure you selected.

- For all indices $r \leq \min(k, m)$, prove (often by induction) that $a_r \geq o_r$ or that $a_r \leq o_r$, whichever the case may be. Don't forget to use your algorithm to help you argue the inductive step.

Step 4: Prove optimality. Prove that since greedy stays ahead of the other solution with respect to the measure you selected, then it is optimal.

Step 5: Analyze runtime.