In lecture we saw the following problem:

Let's consider a problem where each interval i is a task that must be completed; each has a designated time t_i, but we can designate any start time for it. Each interval also has a deadline d_i, which can be different for each interval.

We must assign each interval a start time in such a way that no two intervals overlap. Ideally, we would like to schedule everything to be finished before its deadline, but this is not always possible. We say the lateness l_i of a job is how late it is finished compared to its deadline, $s_i + t_i - d_i$, or 0 if it has been completed by the deadline. Our goal is to minimize the *maximum* lateness: the amount by which the most late job exceeds its deadline.

Show that scheduling tasks in non-decreasing order of deadlines produces an optimal schedule.

The following is how I'd suggest writing it if it were a homework question. If it were an exam question, you can probably get away with a bit less narration, as long as it's clear to the reader where you're going with it, because it's a handwritten exam on a time constraint.

. . . .

Consider any alternative permutation of the tasks[1]. There are a pair of tasks, $i, j = i + 1$, that are consecutive but $d_i > d_j$. I will show that the same permutation, with i and j swapped, has a maximum lateness no worse than this permutation.

Consider when tasks i and j finish in this permutation. Task i is scheduled at some start time s_i and finishes at $f_i = s_i + t_i$. Because task j is next, it starts when task i finishes, and finishes at time $f_j = s_i + t_i + t_j$.

Now consider the permutation ALT, which is the same sequence, but i and j are swapped. Because the quantity $t_i + t_j$ doesn't change, no tasks other than i and j have their finish time altered, so only i and j can change the maximum lateness. What happens to the lateness of tasks i and j?

Task j finishes at time $s_i + t_j$, earlier than it would have before the swap. The lateness from j is either the same (if it was already on time) or less (if it wasn't). So this does not increase the maximum lateness.

Task i finishes at time $f'_i = s_i + t_j + t_i = f_j$, the same time task j would have finished in the original permutation. However, $d_i > d_j$, so the lateness is *no worse than j's would have been.* Thus, the swap does not make the permutation's maximum lateness worse.

Therefore, any permutation with two tasks out of order relative to EDF can be altered to produce a permutation with one fewer pair out of order without increasing the maximum lateness.

. . . .

The full proof would demonstrate that this applies inductively: we can make this incremental change to any permutation other than EDF, so no permutation can have been better than it.

[1]In lecture, I proved that, given any sequence, any alternative permutation has an adjacent pair that is out of order relative to the preferred permutation. If this were a homework or exam question, you would not need to re-prove that. You don't need to remember which lemma number, or which day in class we discussed it, either. There are also some problems where you don't need the fact that the alternative pair is adjacent, although in this problem you do.