Algorithm 1 Job scheduling

Sort the jobs in non-increasing order of profits: $g_1 \geq g_2 \geq \ldots \geq g_n$
$d \longleftarrow \max_i d_i$
for $t : 1..d$ **do**
    $S(t) \longleftarrow 0$
end for
for $i : 1..n$ **do**
    Find the largest t such that $S(t) = 0$ and $t \leq d_i$, $S(t) \longleftarrow i$
end for

Theorem 2.18. *The greedy solution to job scheduling is optimal. That is, the profit $P(S)$ of the schedule S computed by algorithm 1 is as large as possible.*

Lemma 2.19. *"S is promising" is an invariant for the (second) for-loop in algorithm 1.*

Problem 2.21. *Why does lemma 2.19 imply theorem 2.18? (Hint: this is a simple observation).*

Proof. Assume S_{i-1} (i.e. S after $i-1$ iterations) is promising.
Case 1: If task i is not scheduled on iteration i, then there is no S' that extends S_{i-1} which schedules task i. But S_{i-1} is promising, so there is an optimal S' that extends S_{i-1}; clearly the S' does not schedule task i, so "losing access" to task i does not change the fact that $S_i = S_{i-1}$ is promising.
Case 2: If, on the other hand, task i is scheduled at time t_i, let S'_{i-1} be the optimal extension of S_{i-1}. If S'_{i-1} schedules task i at t_i, then we're done. If t_i is scheduled at a different time t_0, then $t_0 < t_i$, as t_i was the latest available time for task i. So let S'_i be S'_{i-1}, but with the tasks at times t_i and t_0 switched; we know task i may be moved to t_i as t_i is before d_i, and we know the task scheduled at t_i may be moved to t_0 because $t_0 < t_i$. We have found an extension of S_i with the same cost as S'_{i-1}, and S'_{i-1} is optimal. Therefore, S_i is promising. $\square$