Algorithm 1 Job scheduling

```
Sort the jobs in non-increasing order of profits: g_1 ≥ g_2 ≥ ... ≥ g_n
d ⟵ max_i d_i
for t : 1..d do
    S(t) ⟵ 0
end for
for i : 1..n do
    Find the largest t such that S(t) = 0 and t ≤ d_i, S(t) ⟵ 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 that after every iteration of the second for loop, the result is promising. After the final iteration, S is still promising, but the only unscheduled tasks are those that cannot extend S at any time. S is promising but it cannot be extended, so it must be optimal.

Identically, assume the last addition made to the schedule is on iteration i. Before it was added, S_{i-1} was promising, and there was exactly 1 task which could extend it. Clearly the profit gained from scheduling this task is the same regardless of when it is scheduled, so every extension of S_{i-1} has the same profit, equal to that of S_i. $\square$