I. Core / D. Dynamic programming and greedy algorithms

- Dynamic programming review and overview (last week Wednesday)
- Dynamic programming practice (last week Friday and **today**)
- Greedy algorithms (Wednesday, Friday, and next week Monday)

Today:

- Finish sawmill problem
- Work through company party problem

A lumberjack has an ℓ-yard long log of wood he wants cut at n specific places L_1, L_2, $\ldots L_n$, represented as the distance of that cut point from one end of the log. (We can also consider the ends as trivial "cut points" $L_0 = 0$ and $L_{n+1} = \ell$.) The sawmill charges \$$x$ to cut a log that is x yards long (regardless of where that cut is). The sawmill also allows the customer to specify the ordering and location of the cuts.
For example, if $\ell = 20$ and we want cuts at 3 yards, 6 yards, and 10 yards from the left end, then if we cut them from left to right the cost would be

$$20 + (20 - 3) + (20 - 6) = 20 + 17 + 14 = 51$$

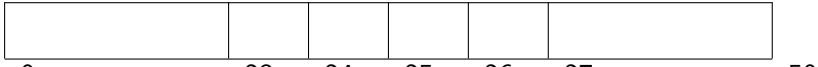
But making the same cuts from right to left would cost

$$20 + 10 + 6 = 36$$

Devise and implement an algorithm to minimize the cost, and analyze its running time.

Don't be greedy:

Let $\ell = 50$ and $L = [0, 23, 24, 25, 26, 27, 50]$



Cutting in half:

$$50 + 25 + 2 + 25 + 2 = 104$$

Trimming edges:

$$50 + 27 + 4 + 3 + 2 = 86$$

15-6 Professor Stewart is consulting for the president of a corporation that is planning a company party. The company has a hierarchical structure; that is, the supervisor relation forms a tree rooted at the president. The personnel office has ranked each employee with a conviviality rating, which is a real number. In order to make the party fun for all attendees, the president does not want both an employee and his or her immediate supervisor to attend.

Professor Stewart is given the tree that describes the structure of the corporation *using the left-child right-sibling representation described in Section 10.4*. Each node of the tree holds, in addition to the pointers, the name of an employee and that employee's conviviality ranking. Describe an algorithm to make up a guest list that maximizes the sum of the conviviality ratings of the guests. Analyze the running time of your algorithm.

For next time:

Read Sec 16.(1&2).

(Nothing to turn in, but Ex 16.2-4 will be assigned for next time.)