Last time we saw the scheduling of activities for a conference room.

Objective: maximize the # of activities scheduled to a conf. room (that are compatible).

Given S = start time of availability of room
list of activities w/ (s_i, f_i)
start time, finish time.

example:

(1,4), (3,6), (1,7), (5,8), (3,9), (5,10), (6,11), (8,12)
= 8 activ.

S = 1

Last time we did the dyn. prog. solution but an easier Greedy solution exists:

```
Schedule(S, activ.list):
1. maxAct ← 0
2. sort activitylist by finish times (low to high)
3. for i ← 1 to n
4.     if actlist[i].s >= S
5.         sched. actlist[i]
6.         maxAct ++
7.         S ← actlist[i].f
8. return maxAct
```

Running time? $\Theta(n \lg n)$ for sort + $\Theta(n)$

$\Rightarrow \Theta(n \lg n)$.

Will the greedy sol. always work?
(that is always give us MAX ACTIV.s)

Define Greedy Choice = take the earliest finish time that is still compat. w/ the first set of greedy choices.

Proof by induction.

BASE CASE: We can always start w/ greedy choice.

Let S be any optimal sol. (that is it has MAX # of compat. acts.) ordered by finish time.

$\langle s_1, s_2, \ldots s_t \rangle$

s_1 can be replaced by the greedy choice

Notice

g_1 can replace s_1 (g_1 is the 1ST greedy choice, i.e. it ends the earliest).

so g_1 can replace s_1 which ends later.

Ind. hypothesis Assume S is an optimal sol. that begins w/ k greedy choices.

$S = \langle g_1, g_2, g_3 \ldots, g_k, s_{k+1} \ldots s_t \rangle$

can replace s_{k+1} w/ g_{k+1} b/c by defn it is compatible w/ g_k & it ends earlier than s_{k+1} //

New topic.

Binary Min Heaps. What are they?

a binary tree w/ parent <= children at every node

no relationship btwn L & R children.

also always complete. (balanced).

Binary Min Heap is a common implementation of a PQ. which needs these operations

1. Extract Min (get & remove the min)
2. Insert
3. Decrease Key

Compare a Binary Min Heap to a BST.

- a heap has lighter restrictions on ordering than a BST. We give up search but have easy balancing.