CS2100

Priority Queues
Heaps
- No class next week
- HW due Sunday after break
- Lab due today

No Office hours today
Last time:

Priority queues:

Store 2 things:
- value
- key

Operations:

Given priority queue PQ:

• insert(e, k): adds e to PQ with priority k
• get Max(): returns maximum value in PQ
• remove Max():
  (plus size & empty)

Note: Many possible implementations!

We talked about 2 using vectors.
One (best?) way: heaps

**Heap**: A binary tree where:
- We use one way to do a priority queue
- For every node \( v \) (other than \( r \)) the key stored at \( v \) is \( \leq \) key stored at \( v \)'s parent

- The tree is complete:
  - Levels 0...h-1 are full
  - \( h \) is filled in left to right

**Example**: 

```
keys
```

```
levels
```

```
Example:
```

```
```

```
```

```
```

```
```

```
```
3 functions: insert, get Max, remove Max.

\[ \text{while (curr < either child) Swap w/ larger child} \]

\[ \text{while (curr > parent) move up} \]
Now: Code! (on webpage)

Recall: Array based trees:

- Array based:
  - $\text{left}(v) = \frac{2v + 1}{2}$
  - $\text{right}(v) = \frac{2v + 2}{2}$
  - $\text{parent}(v) = \left\lfloor \frac{v}{2} \right\rfloor - 1$

\[ A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\} \]
Runtimes:

- empty & size: $O(1)$
- get Max: $O(1)$
- Insert & remove:
  - while loop:
    - traverses a root-to-leaf path in the tree once
    - n nodes height $\lceil \log_2 n \rceil + 1$
    - $O(\log_2 n)$