I & C SCI 46 Winter 2023
Lecture 15: Priority Queues and Heaps
Finishing last lecture
Height of a Level-Balanced Tree

\[ n_r = \text{smallest } w/ \text{ root at level } r \]

\[ n_0 = 0 \]
\[ n_1 = 1 \]
\[ n_2 = 2 \]
\[ n_3 = 3 \]
\[ n_4 = 4 \]

\[ n_0, n_1, n_2, \text{ then } \]
\[ n_L = 1 + 2n_{L-2} \]
What is a priority queue?

not a lookup structure

- `insert(e)`: inserts given element into PQ.
- `min()`: returns reference to a smallest elt
- `extractMin()`: remove what `min()` returns.
What is a complete binary tree?

What is the min heap property?
Heap as std::vector

- Where is root?
- Where is i’s left child? $v[2i]$
- Where is i’s right child? $v[2i+1]$
Implementing the Priority Queue

- Complete tree
- Min-heap

- Insert ‘7’
  *Hint: must have heap after*
- Then insert ‘4’ ‘2’ ‘6’
Implementing the Priority Queue

Find min in a heap
Implementing the Priority Queue

- extract-min from heap

*Hint: must have heap after*
XKCD # 835: Tree. Not only is that terrible in general, but you just KNOW Billy’s going to open the root present first, and then everyone will have to wait while the heap is rebuilt.
Common Key

- Two min-heaps, $A$ and $B$
- Do they have a common element?
- Use only the priority queue API

```plaintext
while A and B both non-empty:
    if A.min() == B.min(): return true
    if A.min() < B.min():
        A.extractMin()
    else:
        B.extractMin()

return false
```