Lecture 15: Loop invariants

- How to develop loops, guaranteeing correctness
- Also: finishing heaps, start sorting

Announcements

- P3 extension (again)
- No T/W discussion
- Two surveys (sorry!)

Heap implements PriorityQueue:
- insert (value, priority), - removeMax ()
- find Max()

Heap implementation:

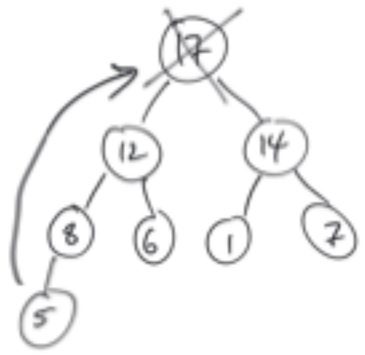
binary tree satisfying: (in priority)
1. each node is larger than its children
2. the tree is full.

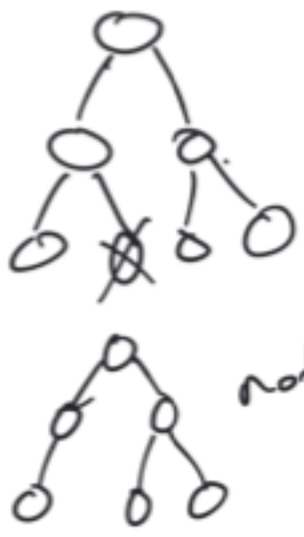
} class invariants

- max is at root
- to insert: put new val in last pos
  - swap with parent & repeat as necessary to satisfy inv. 1

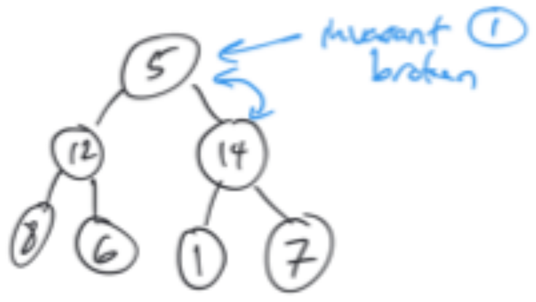
- remove Max():

1. replace the root with the last leaf

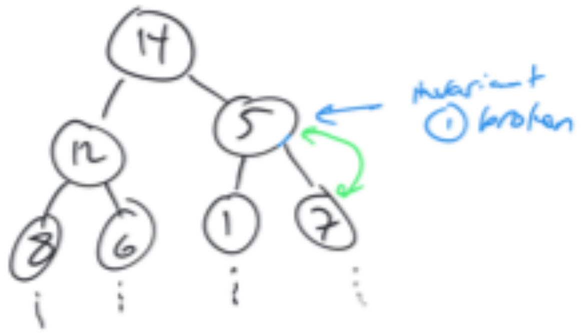


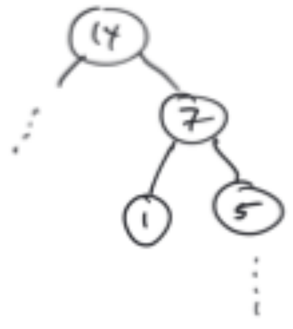

not full!

2. swap root with larger of its children



repeat until invariant holds





loop invariant: something you know is true in each iteration of a loop.

Example: remove from heap.

- save root value to return
- swap last value in for root (remove it).

To develop a loop (4 loopy questions):

- INIT: does it start right?

  set variables so that precondition makes invariant true

- TERMINATE: does it end right?

  create loop gaurd that, with invariants, guarantees postcondition

- PROGRESS: will it end?

  start loop body by making progress towards term. cond.

- PRESERVATION: is it invariant?

  finish loop body by ensuring that invariant still holds after