Preview: Heaps
A **heap** is a special binary tree with two additional properties.
A heap is a special binary tree.

1. **Heap Order Invariant:**
   Each element $\geq$ its parent.
A heap is a special binary tree.

2. **Complete**: no holes!
- All levels except the last are full.
- Nodes in last level are as far left as possible.

```
22 38 55 10 20 ← as far left as possible
```

```
21 8
```

```
6
```

```
4
```

```
14
```

```
Heap operations

class Heap<E> {
    boolean add(E e); // insert e
    E peek(); // return min element
    E poll(); // remove/return min element
    void clear();
    boolean contains(E e);
    boolean remove(E e);
    int size();
    Iterator<E> iterator();
}
boolean add(E e);

Algorithm:
• Add e in the wrong place
• While e is in the wrong place
  • move e towards the right place
boolean add(E e);
boolean add(E e);
boolean add(E e);
boolean add(E e);
boolean add(E e);
boolean add(E e);

Algorithm:
• Add e in the wrong place *(the leftmost empty leaf)*
• While e is in the wrong place *(it is less than its parent)*
  • move e towards the right place *(swap with parent)*

The heap invariant is maintained!
What’s the runtime?

• $O(\text{number of swap/bubble operations})$
  $= O(\text{height of tree})$

• Complete $\Rightarrow$ balanced
  $\Rightarrow$ height is $O(\log n)$

• Maximum number of swaps is $O(\log n)$