Lecture 17: Iterators, Array Representation of Trees and Priority Queues

CS 62
Fall 2015
Michael Bannister & Kim Bruce

Midterm?

- Friday? Wednesday? Exact date?!?

Tree Iterators

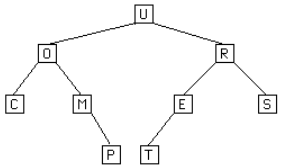
- Pre-order: root, left subtree, right subtree
- Post-order: left subtree, right subtree, root
- In-order: left subtree, root, right subtree.

- Example on board

Array Representations of Trees

Array Representation

- data[0..n-1] can hold values in trees
  - left subtree of node i in 2*i+1, right in 2*i+2,
  - parent in (i-1)/2



Indices:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
data[]:	U	O	R	C	M	E	S	-	-	-	P	T	-	-	-

Array Representation: Efficiency

- Tree of height h, takes $2^{h+1}-1$ slots, even if only has O(h) elements
  - Bad for long, skinny trees
  - Good for full or complete trees.
- Recall complete tree is full except possibly bottom level and has all leaves at that level in leftmost positions.

Priority Queues

PriorityQueue

```
public interface PriorityQueue<E extends Comparable<E>>
{

    /**
     * @pre !isEmpty()
     * @return The minimum value in the queue.
     */
    public E remove();
    public E getFirst();
    public void add(E value);
    public boolean isEmpty();
    public int size();
    public void clear();
}
```

First Implementation

- As regular queue (array or linked) where either keep in order or search for lowest to remove:
  - One of add or remove will be O(n)