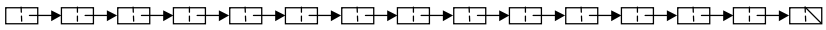
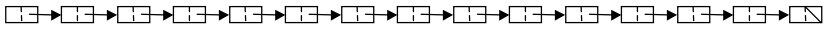


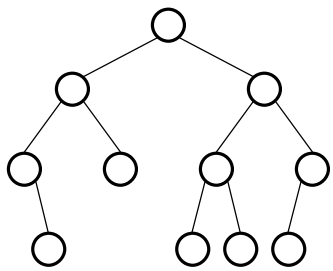
CSE 373: Binary Search Trees

Chapter 4



Binary Trees

Binary Tree: a Tree in which every node has two children or fewer



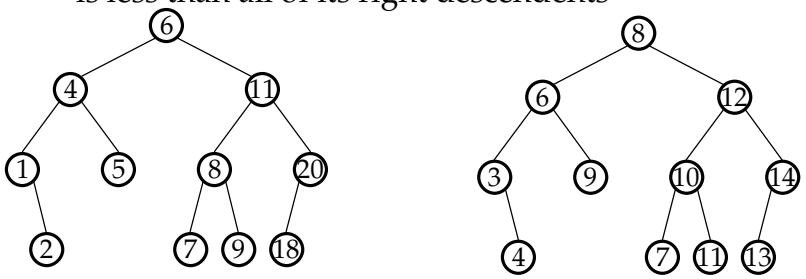
Numerical Trivia for Binary Trees

- Given a binary tree of depth *d*…
  - max number of nodes = min =
  - max number of leaves = min =
- Building a binary tree out of *n* nodes…
  - max depth of tree = min =

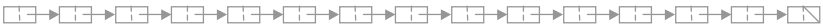
Binary Search Tree

Binary Search Tree: a Binary Tree in which every node…

- is greater than all of its left descendents
- is less than all of its right descendents

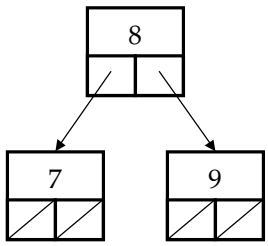


Implementation



Similar to our naive fixed-degree Tree:

```
template <class Comparable>
Class BinaryNode {
  Comparable data;
  BinaryNode* left;
  BinaryNode* right;
};
```



(As with generic trees, could use a parent pointer)

Binary Search Tree Operations

- Search Operations:

```
Comparable& find(Comparable& val);
Comparable& findMin();
Comparable& findMax();
```

- Collection Operations:

```
void insert(Comparable& val);
void remove(Comparable& val);
bool isEmpty();
```

- Creation/Deletion
- Traversals…

Traversals

pre-order:

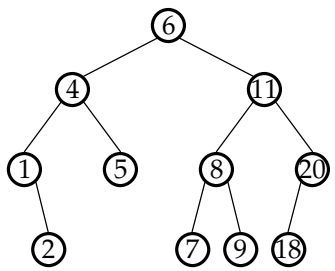
post-order:

in-order:

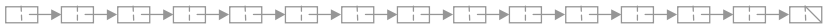
How would these be coded?

Searching

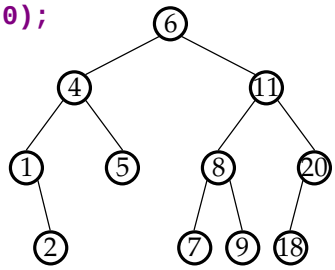
```
T.find(11);
T.find(9);
T.findMin();
T.findMax();
```



insert()

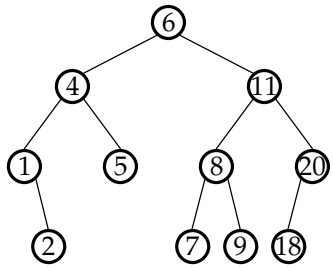


```
T.insert(3);
T.insert(19);
T.insert(0);
```



remove()

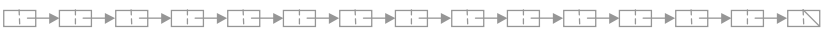
```
T.remove(2);
T.remove(20);
T.remove(11);
```



Asymptotic Analysis

	SearchTree	List	Sorted List (Array)
problem size			
space			
`find()`			
`findMin()`			
`findMax()`			
`insert()`			
`remove()`			
traversals			

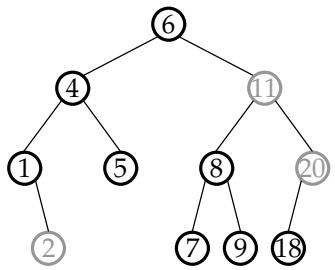
Design Decision: Lazy Deletion



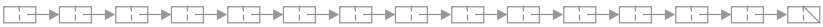
Lazy Deletion: Rather than deleting a node from a tree, merely *mark* it as being deleted

- operate around it as usual
- (just don't return it as the result of a `find()` op)

```
T.remove(2);
T.remove(20);
T.remove(11);
```



Food For Thought



If I read a list of integers from a file and insert them into a Binary Search Tree one by one, what's an example of a worst-case file?