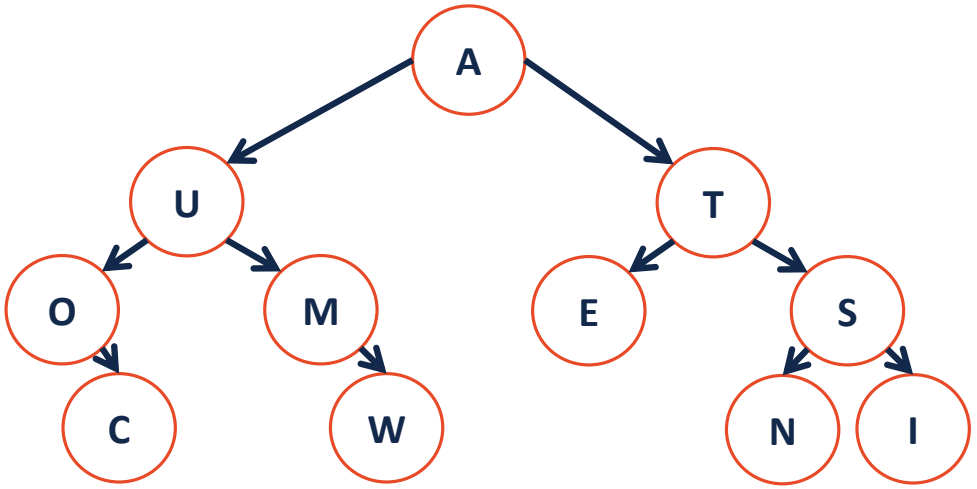
CS 225

Data Structures

October 1 – Binary Search Tree (BST)

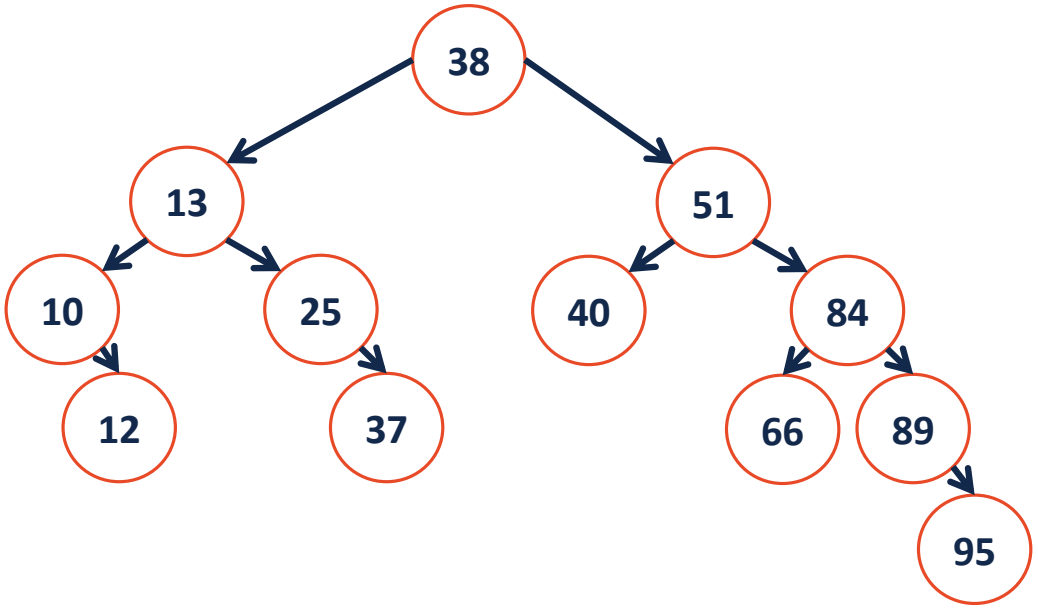
G Carl Evans

Binary Tree as a Search Structure



Binary _____ Tree (BST)

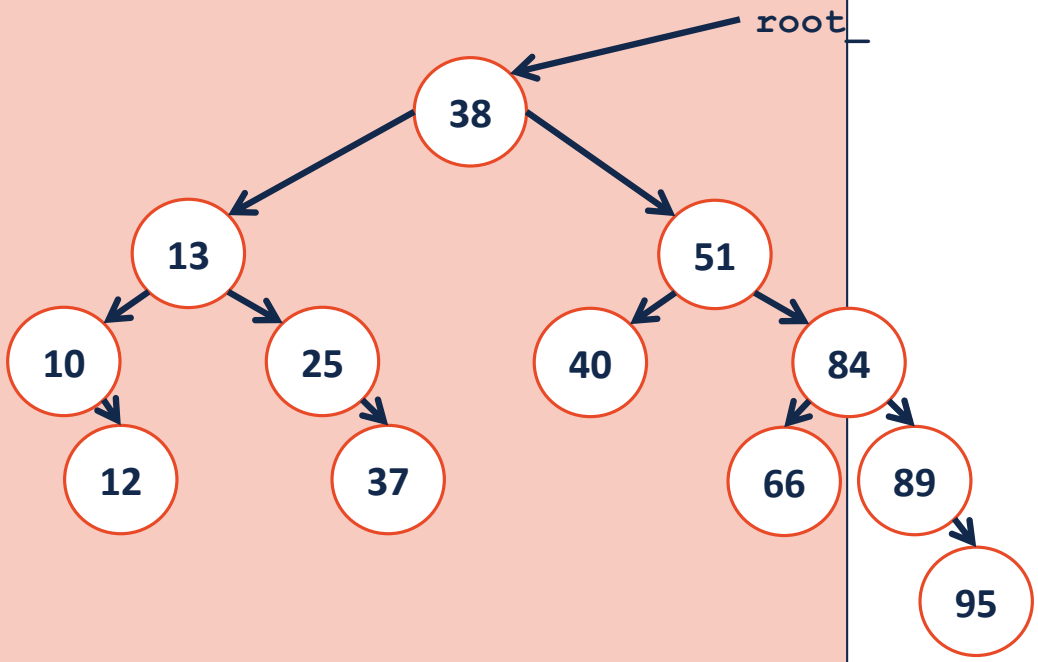
A **BST** is a binary tree **T** such that:



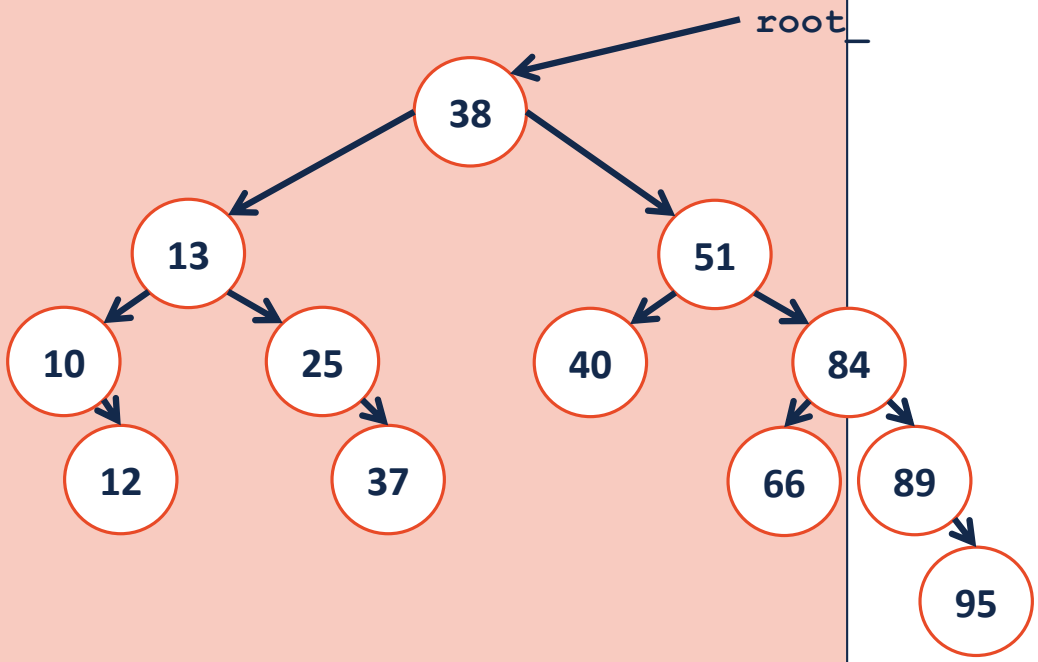
BST.h

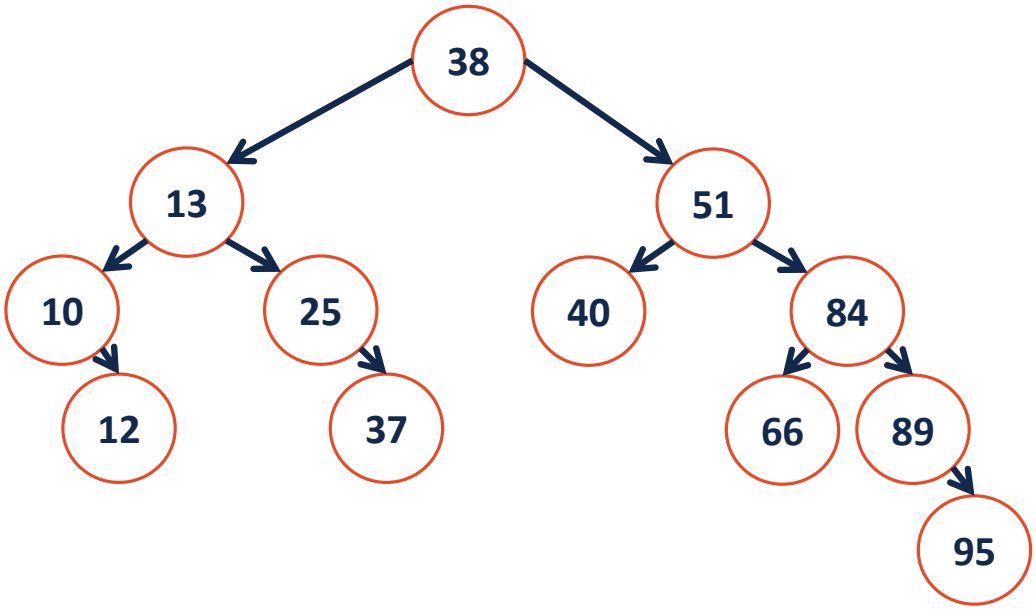
```
 1 #pragma once
 2
 3 template <typename K, typename V>
 4 class BST {
 5   public:
 6     BST();
 7     void insert(const K key, V value);
 8     V remove(const K & key);
 9     V find(const K & key) const;
10     TreeIterator traverse() const;
11
12   private:
13     struct TreeNode {
14        TreeNode *left, *right;
15        K & key;
16        V & value;
17        TreeNode(K & k, V & v) : key(k), value(v), left(NULL),
18           right(NULL) { }
19     };
20
21     TreeNode *head_;
22 };
```

```
1  template<typename K, typename V>
2  _____________________________ find(const K & key) const {
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26 }
```

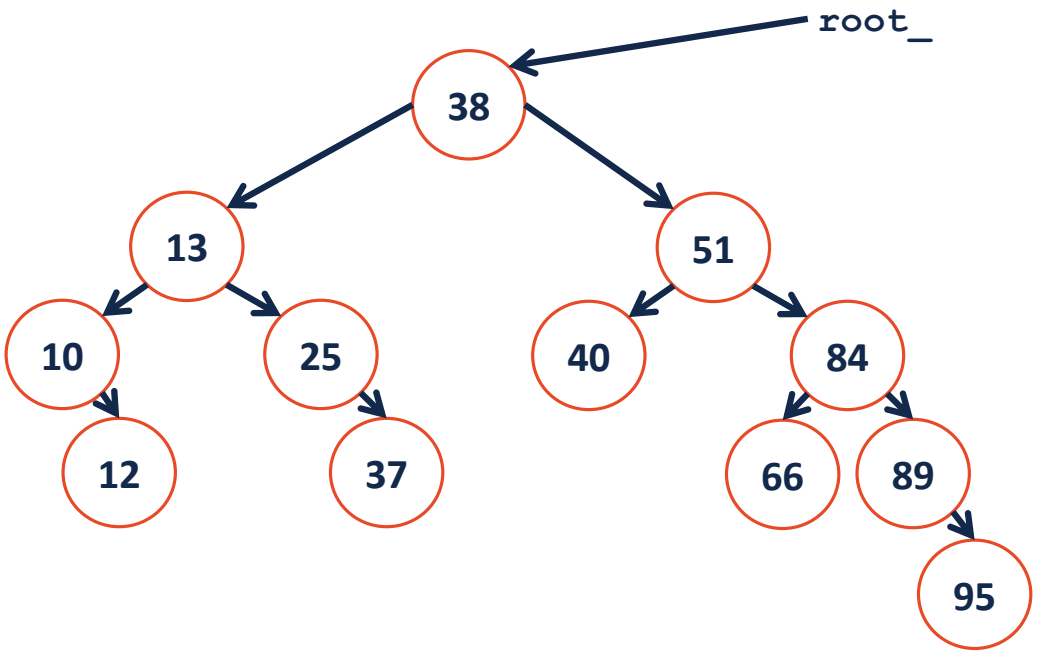


```
1  template<typename K, typename V>
2  _____________________________ _find(TreeNode *& root, const K & key) const {
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26 }
```

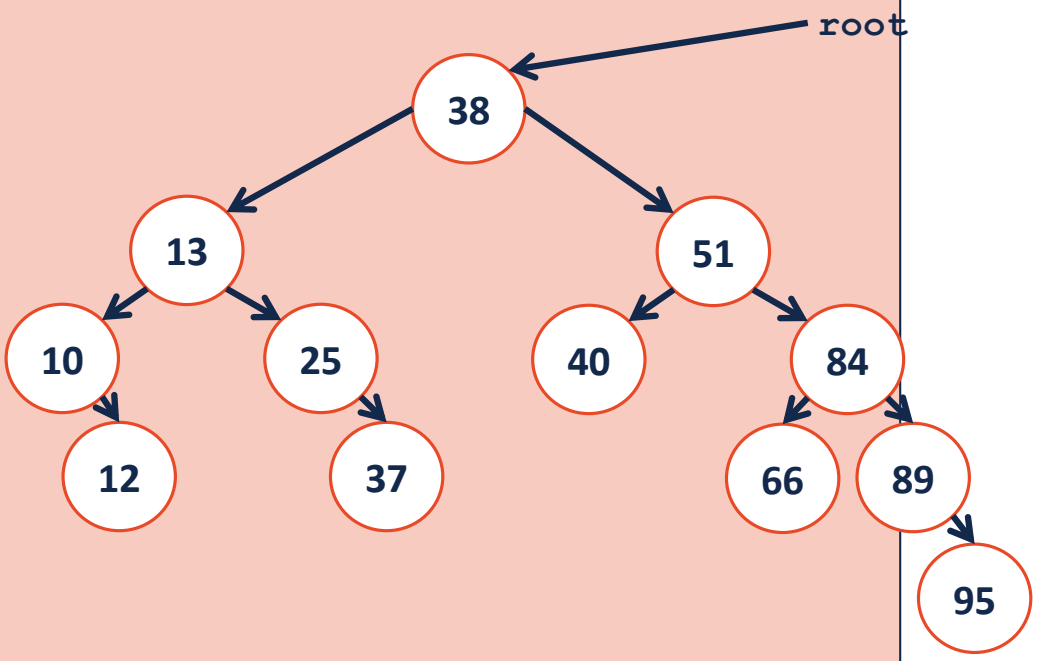


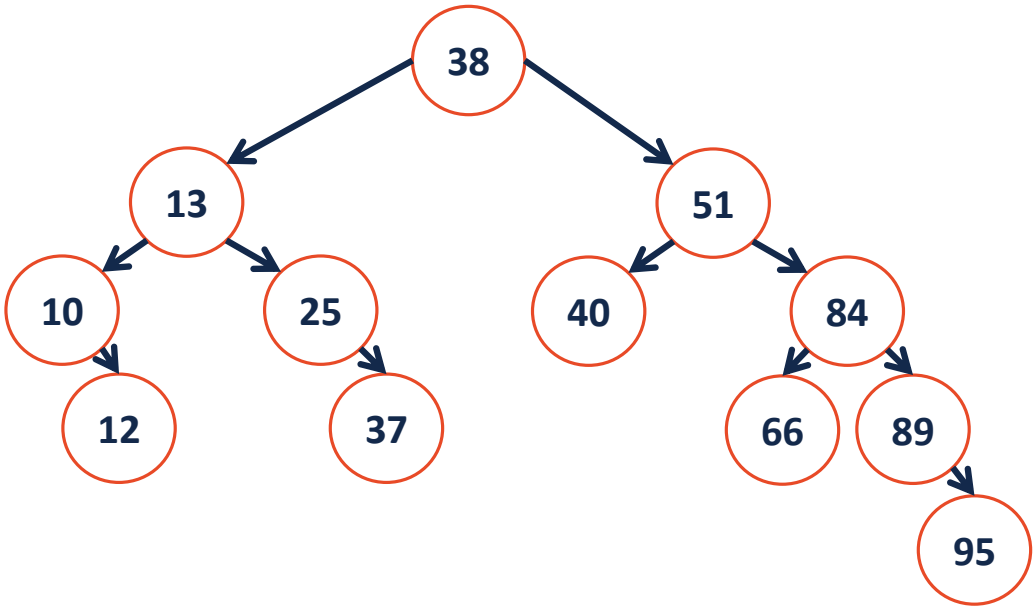

38
13
51
10
25
40
84
12
37
66
89
95

```
1  template<typename K, typename V>
2
3  void BST::_insert(TreeNode *& root, K & key, V & value) {
4    TreeNode * t = _find(root, key);
5    t = new TreeNode(key, value);
6  }
```

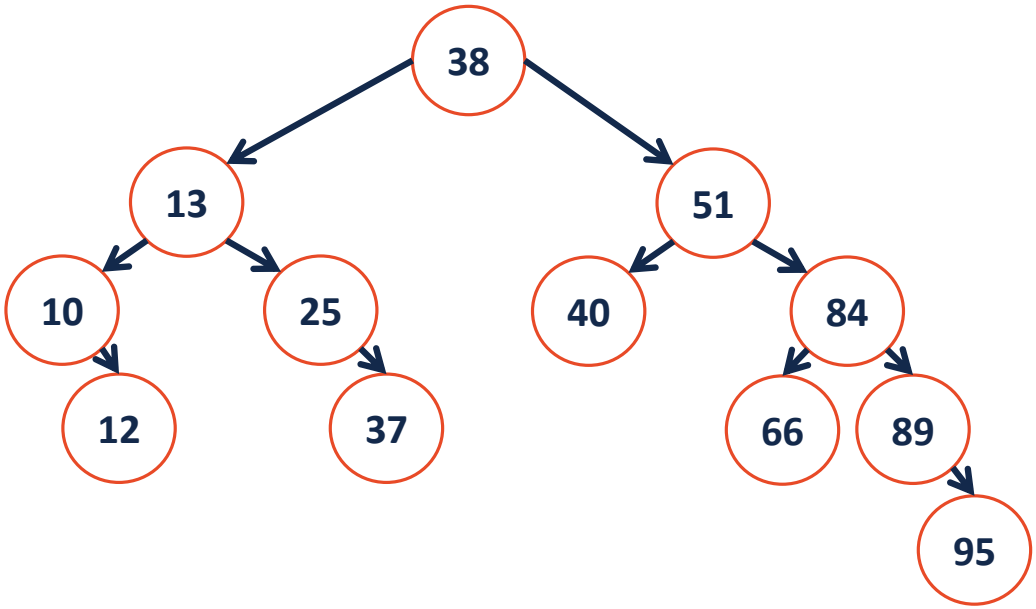


```
1  template<typename K, typename V>
2  _____________________________ _remove(TreeNode *& root, const K & key) {
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26 }
```

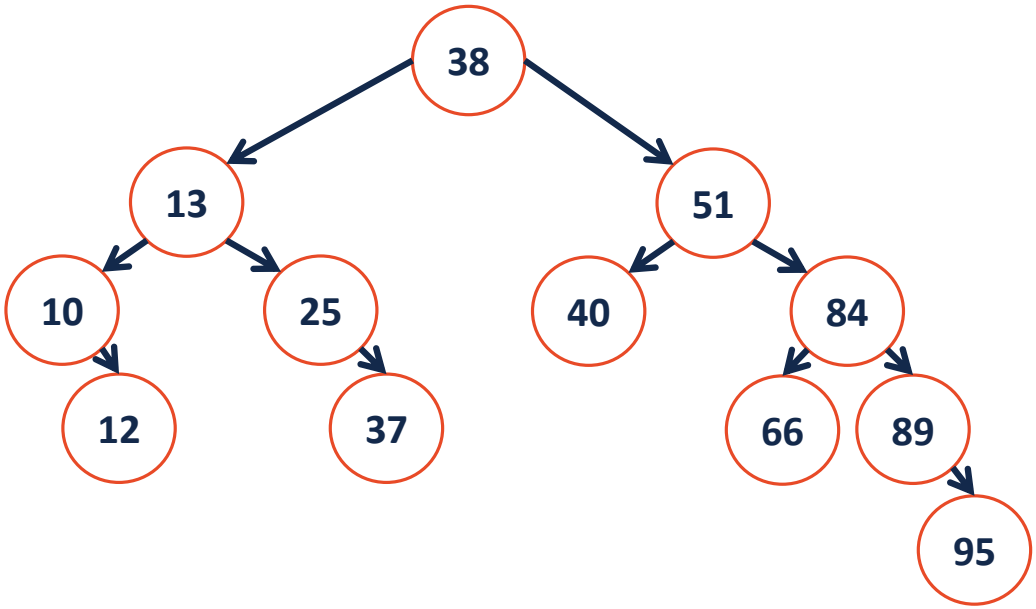




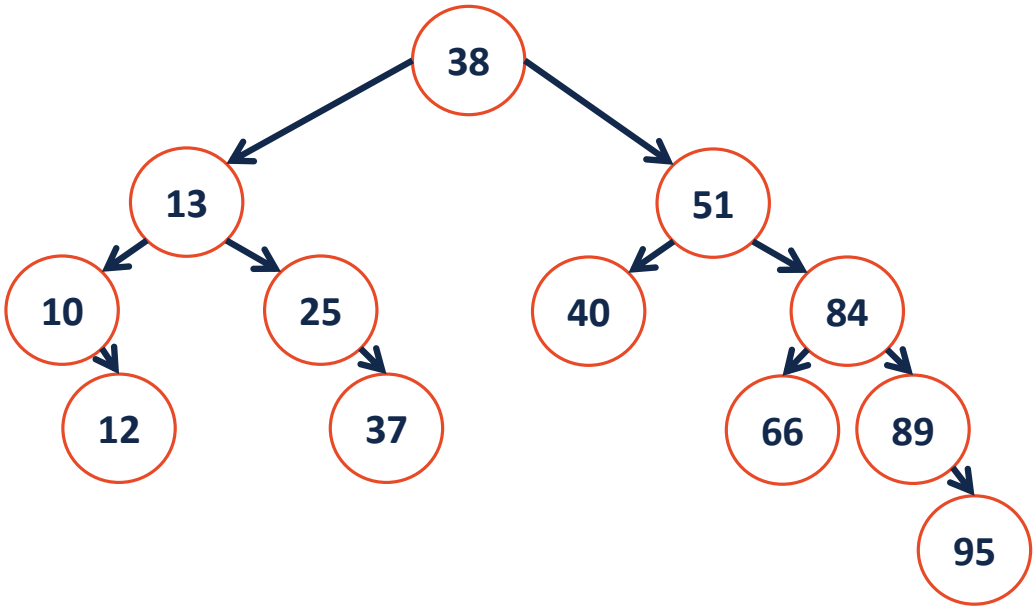

```
remove(40);
```




```
remove(25);
```




```
remove(10);
```




```
remove(13);
```

BST Analysis – Running Time

Operation	BST Worst Case
find	
insert	
delete	
traverse	

BST Analysis

Every operation that we have studied on a BST depends on the height of the tree: **O(h)**.

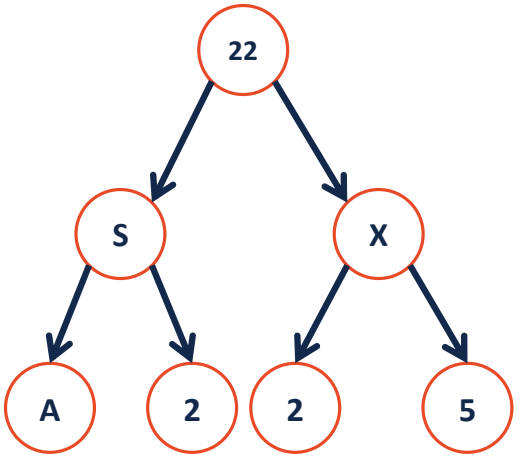
…what is this in terms of **n**, the amount of data?

We need a relationship between **h** and **n**:

$f(h) \leq n \leq g(h)$

BST Analysis

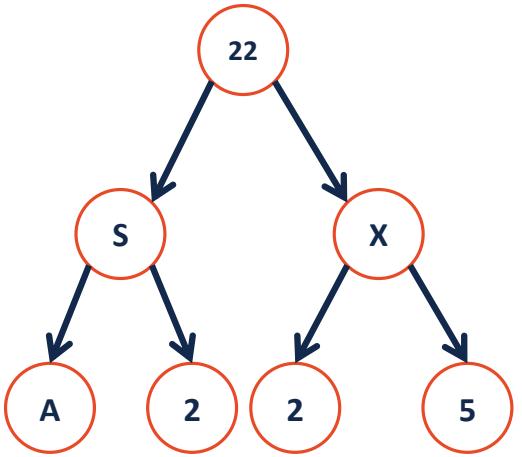
Q: What is the maximum number of nodes in a tree of height **h**?



BST Analysis

Q: What is the minimum number of nodes in a tree of height **h**?

What is the maximum height for a tree of **n** nodes?



BST Analysis

Therefore, for all BST:

Lower bound:

U**pper bound:**

BST Analysis

The height of a BST depends on the order in which the data is inserted into it.

ex: 1 3 2 4 5 7 6 vs. 4 2 3 6 7 1 5

Q: How many different ways are there to insert keys into a BST?

Q: What is the average height of all the arrangements?

BST Analysis

Q: How many different ways are there to insert keys into a BST?

Q: What is the average height of all the arrangements?

BST Analysis – Running Time

Operation	BST Average case	BST Worst case	Sorted array	Sorted List
find				
insert				
delete				
traverse				