CS 225

Data Structures

February 9 – BST

G Carl Evans

Traversal vs. Search

Traversal

Search

Search: Breadth First vs. Depth First

Strategy: Breadth First Search (BFS)

Strategy: Depth First Search (DFS)

Dictionary ADT

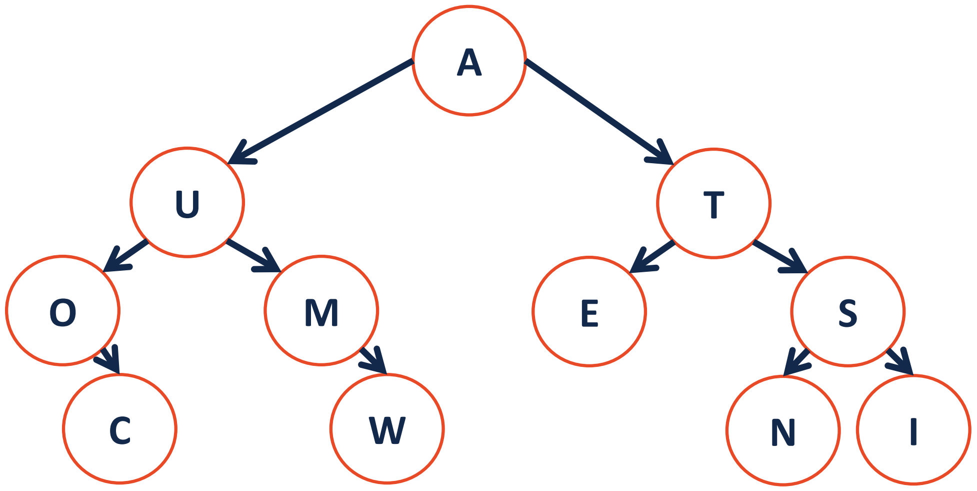
Data is often organized into key/value pairs:

UIN ➔ Advising Record
Course Number ➔ Lecture/Lab Schedule
Node ➔ Incident Edges
Flight Number ➔ Arrival Information
URL ➔ HTML Page
…

Dictionary.h

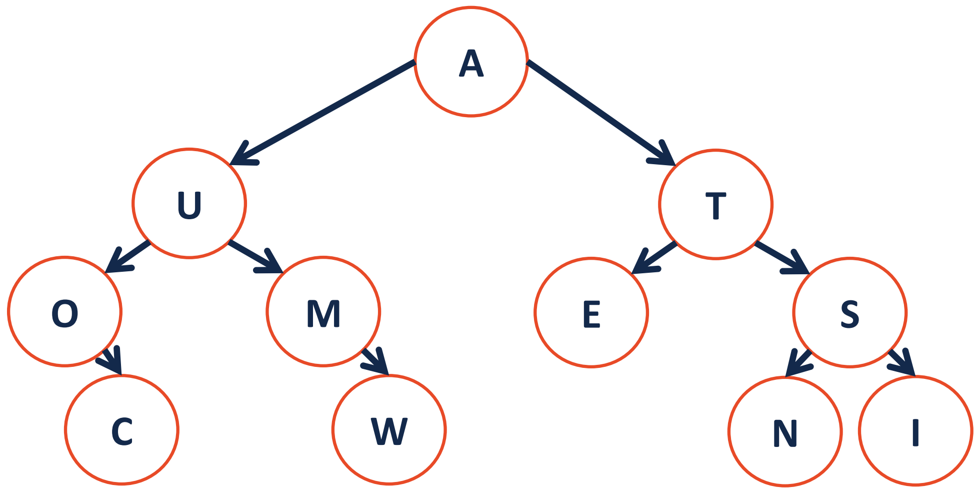
```
#pragma once


class Dictionary {
  public:
    void insert(const K key, V value);
    V remove(const K & key);
    V find(const K & key) const;
    TreeIterator begin();
    TreeIterator end();

  private:
    // ...








};
```

Binary Tree as a Search Structure



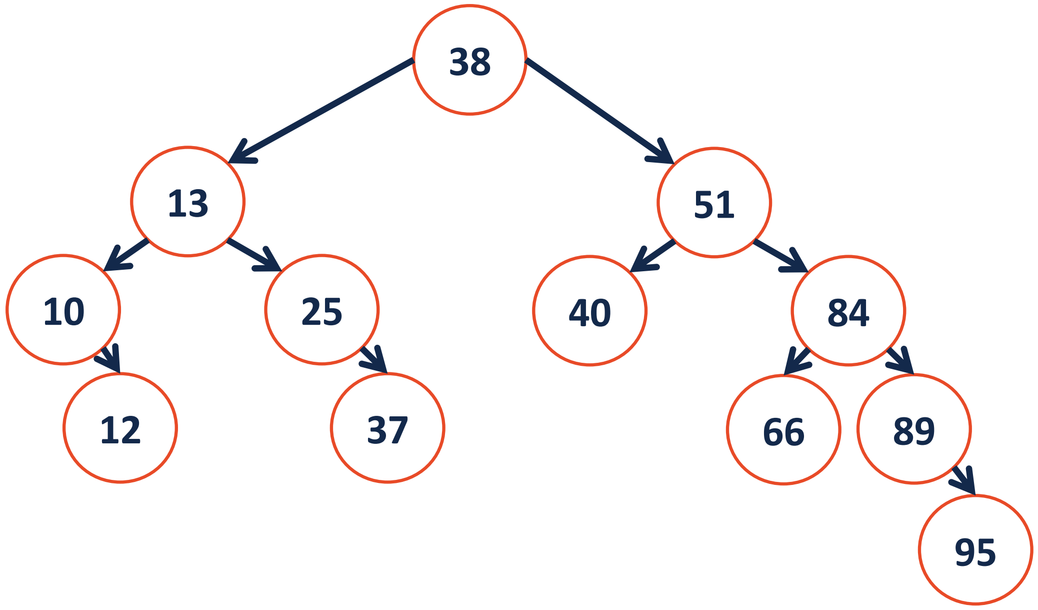
Binary Tree as a Search Structure

Binary Tree Runtimes



Binary _____________ Tree (BST)

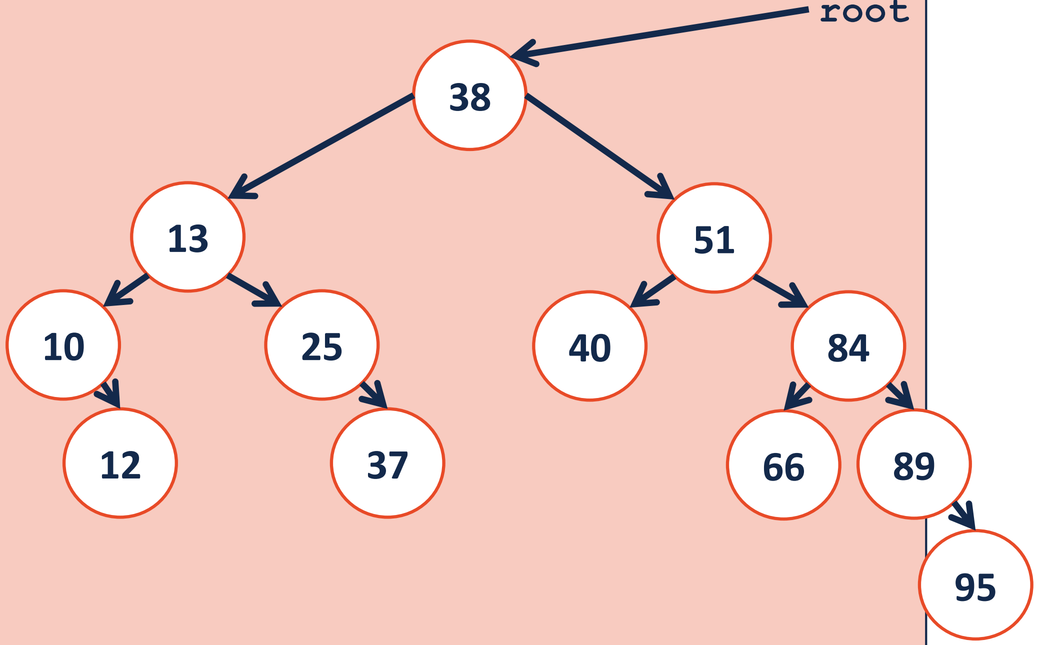
A **BST** is a binary tree **T** such that:

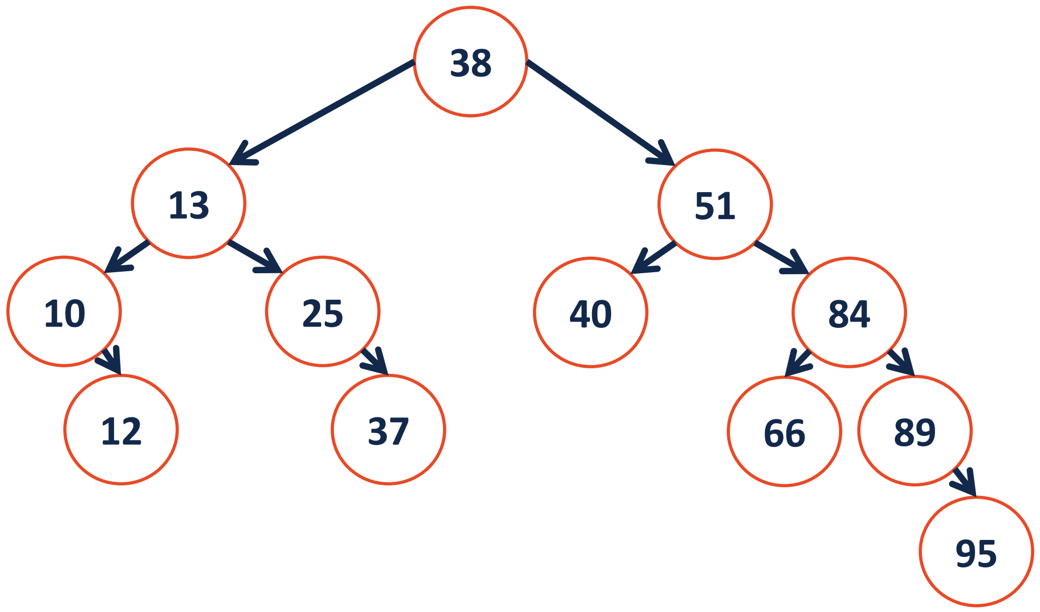


BST.h

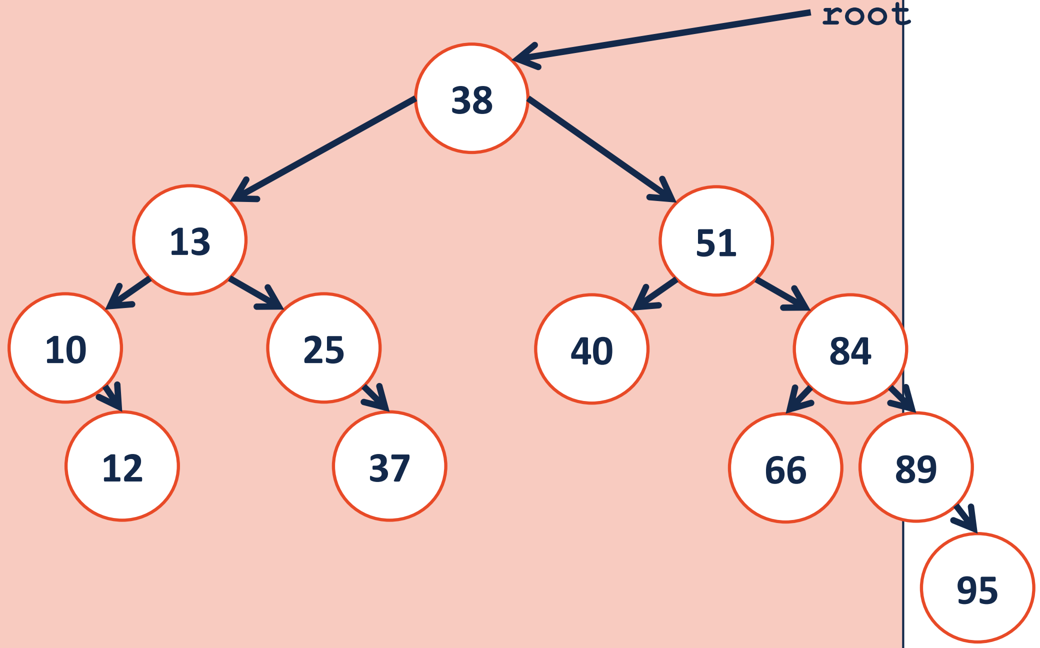
```
#pragma once

template <class K, class V>
class BST {
  public:
    BST();
    void insert(const K key, V value);
    V remove(const K & key);
    V find(const K & key) const;
    TreeIterator begin();
    TreeIterator end();

  private:









};
```

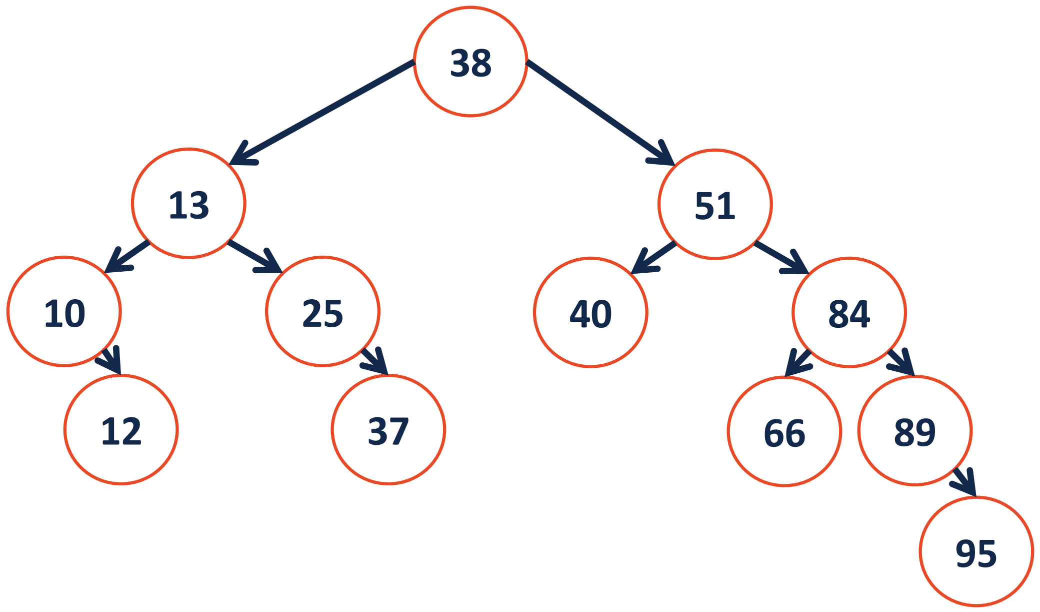
```
template<class K, class V>

_________________________ _find(TreeNode *& root, const K & key) {
























}
```



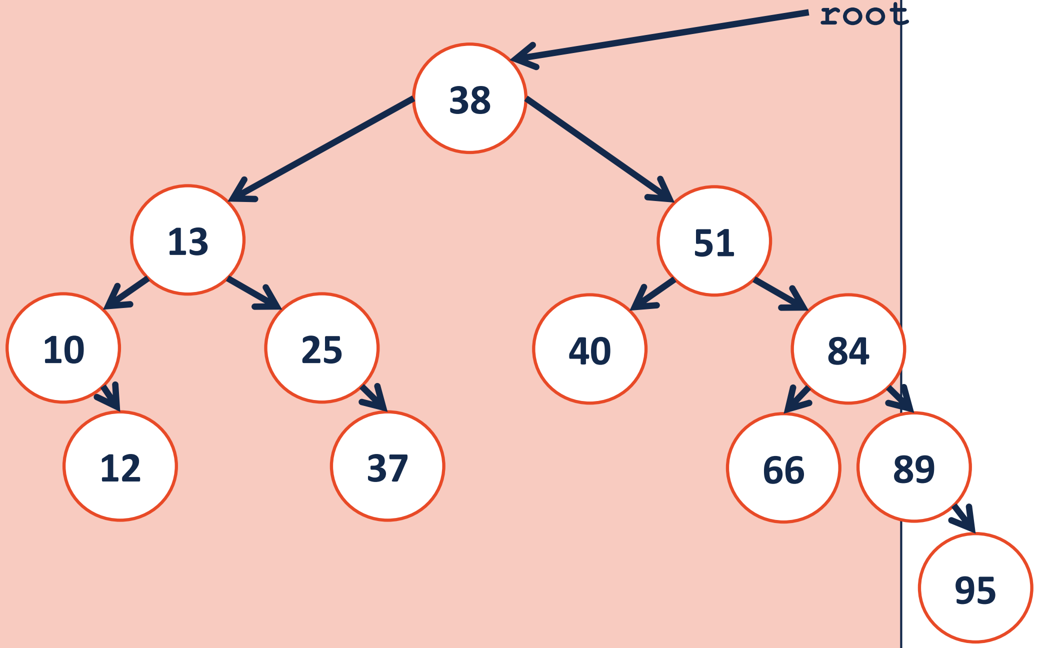

38
13
51
10
25
40
84
12
37
66
89
95

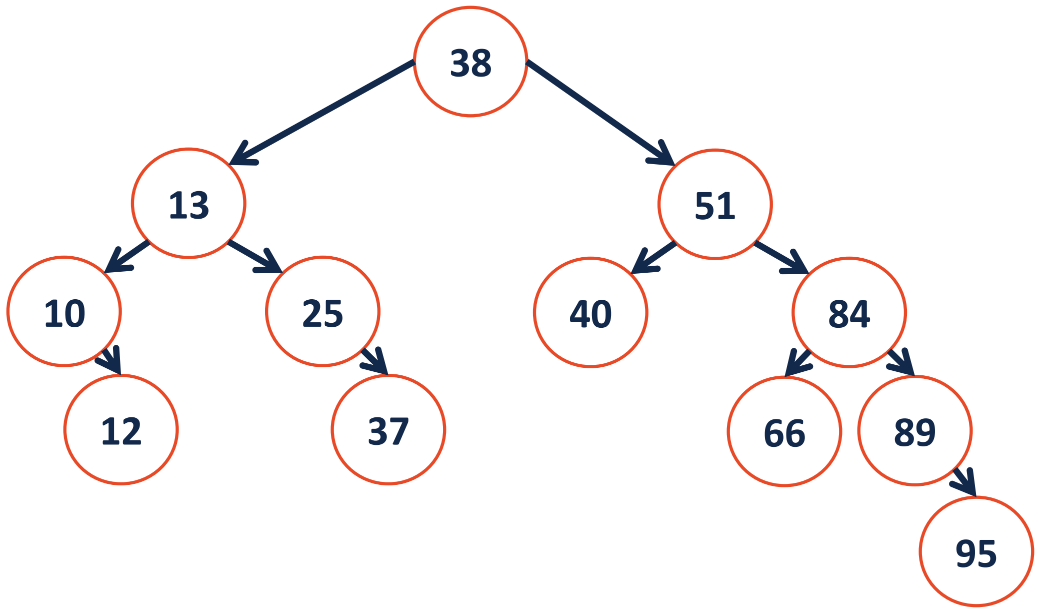
```
template<class K, class V>

_________________________ _insert(TreeNode *& root, const K & key) {
























}
```



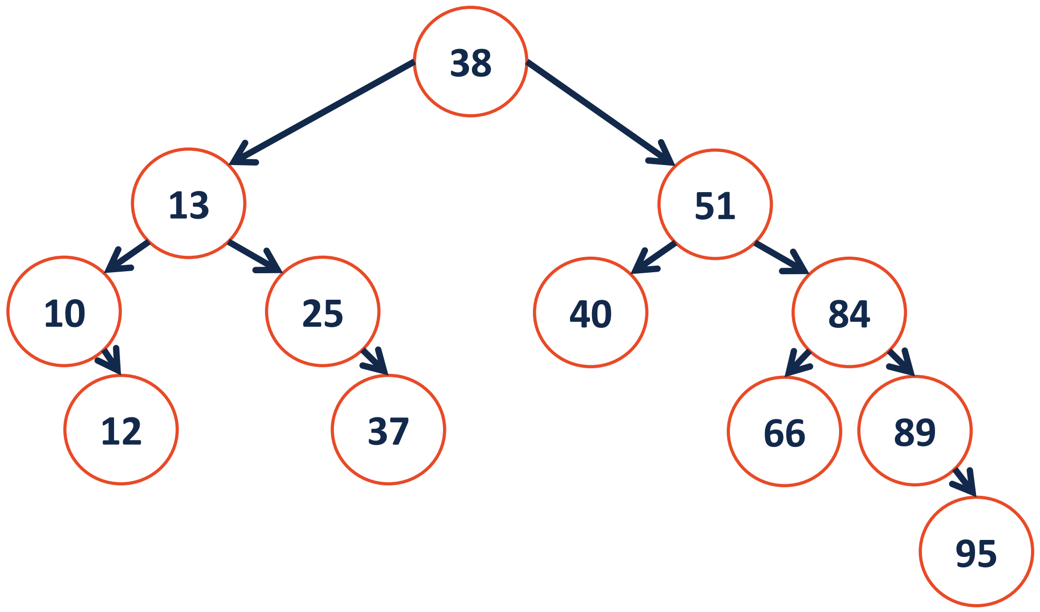

38
13
51
10
25
40
84
12
37
66
89
95

```
template<class K, class V>

_______________________ _remove(TreeNode *& root, const K & key) {
























}
```

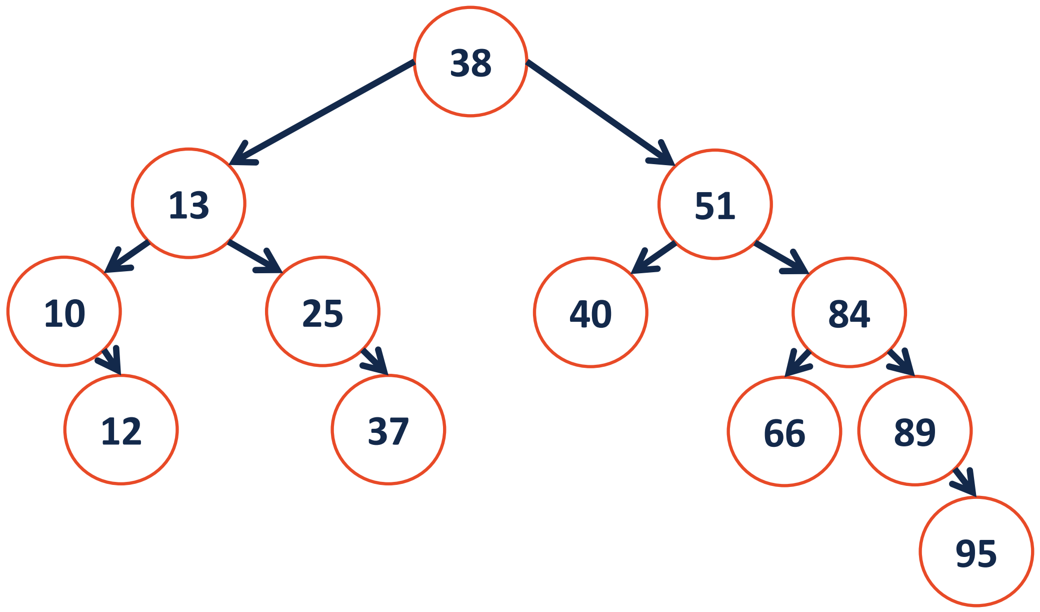




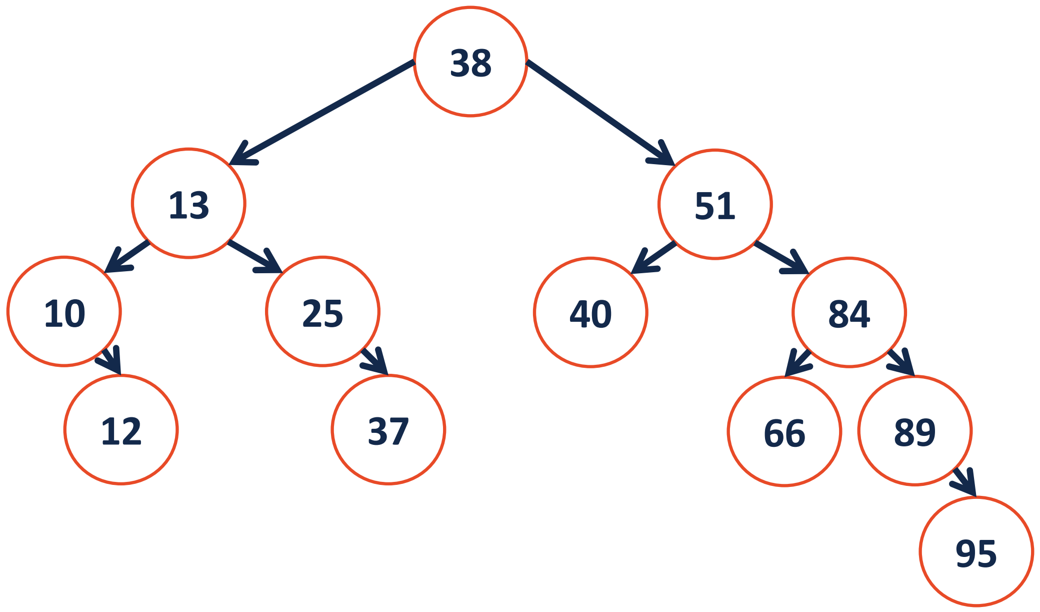
```
remove(40);
```



```
remove(25);
```



```
remove(10);
```



```
remove(13);
```