CSE12 - Lecture 18

Monday, May 15, 2023 8:00 AM

PA6
PA3 Late/Resubmit > due tomorrow

Binary Search Tree (BST)

key value

implements Map<K, V>

```
class BST<K, V> {
  Node<K, V> root;
  BST() {this.root = null);
  BST(Node<K, V> root) { this.root = root; }

  V get(Node<K, V> node, K key) {
    if (node == null) { //throw error }
    if (node.key.equals(key)) {
      return node.value;
    }
    if (node.key < key) {
      return get(node.left, key);
    }
    else {
      return get(node.right, key);
    }
  }

  V get(Key key) {
    return this.get(root, key);
  }
}
```

private
base cases
recursive cases
public
not found
found
greater
lesser

```
class Node<K,V> {
  K key;
  V value;
  Node<K,V> left;
  Node<K,V> right;
  public Node(K key, V value,
              Node<K,V> left,
              Node<K,V> right) {
    this.key = key;
    this.value = value;
    this.left = left;
    this.right = right;
  }
```

Where is the get() method broken?

> does not work

How can we fix the get() method to work with Objects?

Interface → compare(_) 0, 1, -1

- Comparator / also an Object — to pass to the constructor, save as a field
- Comparable
  - compareTo()
    - <0 less than
    - 0 equal
    - >0 greater than
  - public String implements Comparable

What error should we throw in get() if the key isn't found?

NoSuchElementException() / ElementNotFoundException()

What would the code that uses get() look like to prevent the program crashing if the key is missing?

```
Bst<_,_> tree = ....
try {
_ = tree.get(5);
} catch (NoSuchElementException e) {
  // what do we do?
} catch (Exception e) {
  //?
}
```

```
boolean find(E toFind, E value) {
  Comparable comp = (Comparable) toFind;
  while loop
    if (comp.compareTo(value) == 0) {
      return true;
```

runtime error → bad

1. `Bst<K extends Comparable, V>`
   `class Test<E extends Comparable>`
   `while (toFind.compareTo(value))`

Name: _____ PID: _____ Code: 6150

Assume the key and value are identical for this example:

Trace the path for get(4)
How many nodes does it touch?

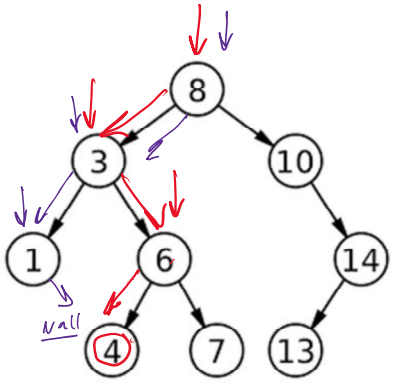
4 nodes

Trace the path for get(2)
How many nodes does it touch?

3 nodes, 4 comparisons

What happens when the node isn't found?

throw an exception
No Such Element
Element Not Found



key smaller node.key
→ go left

key greater node.key
→ go right

key equals node.key
→ found it!

node == null
- did not found it!

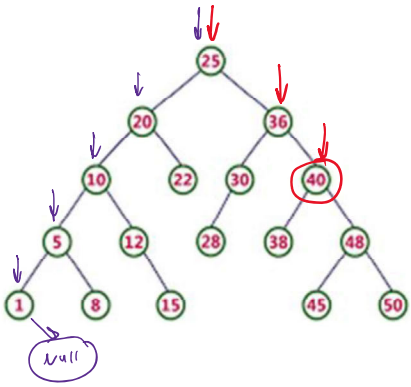
Assume the key and value are identical for this example:

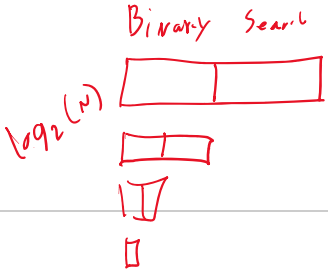
Trace the path for get(40)
How many nodes does it touch?

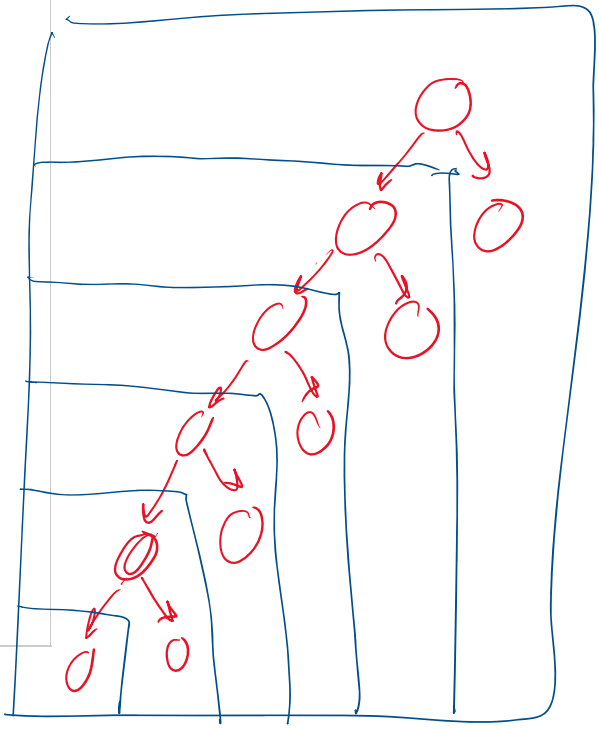
3 nodes

Trace the path for get(4)
How many nodes does it touch?

5 nodes
6 comparisons







recursive data structure