CS 61BL

Lab 08

Ryan Purpura

Announcements

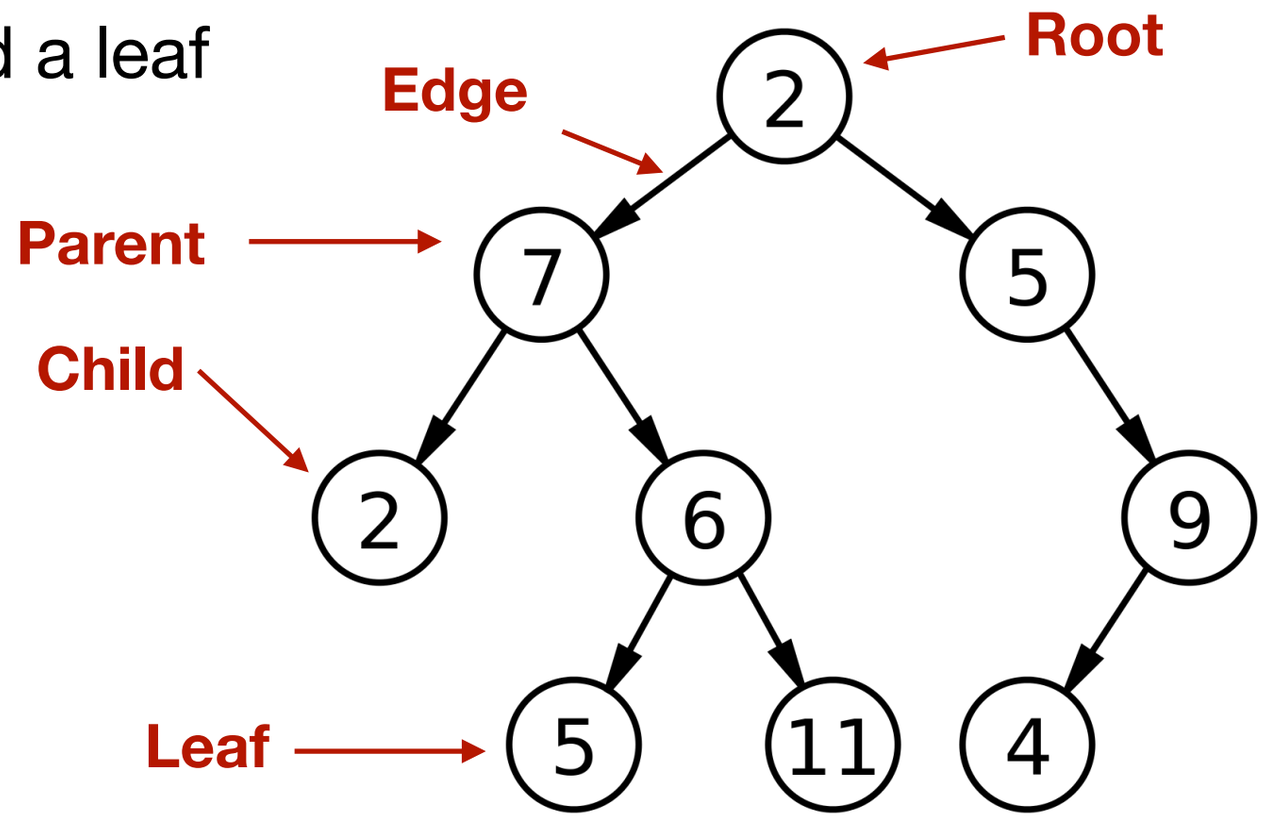
- You are a quarter-way through the course!

- Project 2 (Gitlet) will be released in the upcoming days. This is a partner project.

- No worksheet today.

Trees

- A tree is a hierarchical set of nodes (elements that link to other elements)
- A node has a parent (except the topmost node called a "root node") and can have multiple children
- A node without children is called a leaf



Example Tree

```
class TreeNode {
    public int item;
    public TreeNode parent;
    public ArrayList<TreeNode> children;

    public TreeNode(int item, TreeNode parent) {
        this.item = item;
        this.parent = parent;
        this.children = new ArrayList<>();
    }
}
```

An Arraylist is a collection of items like an array, but will resize automatically (grow or shrink) as items are removed. It is implemented very much like your ArrayDeque!

Implementing Tree Methods

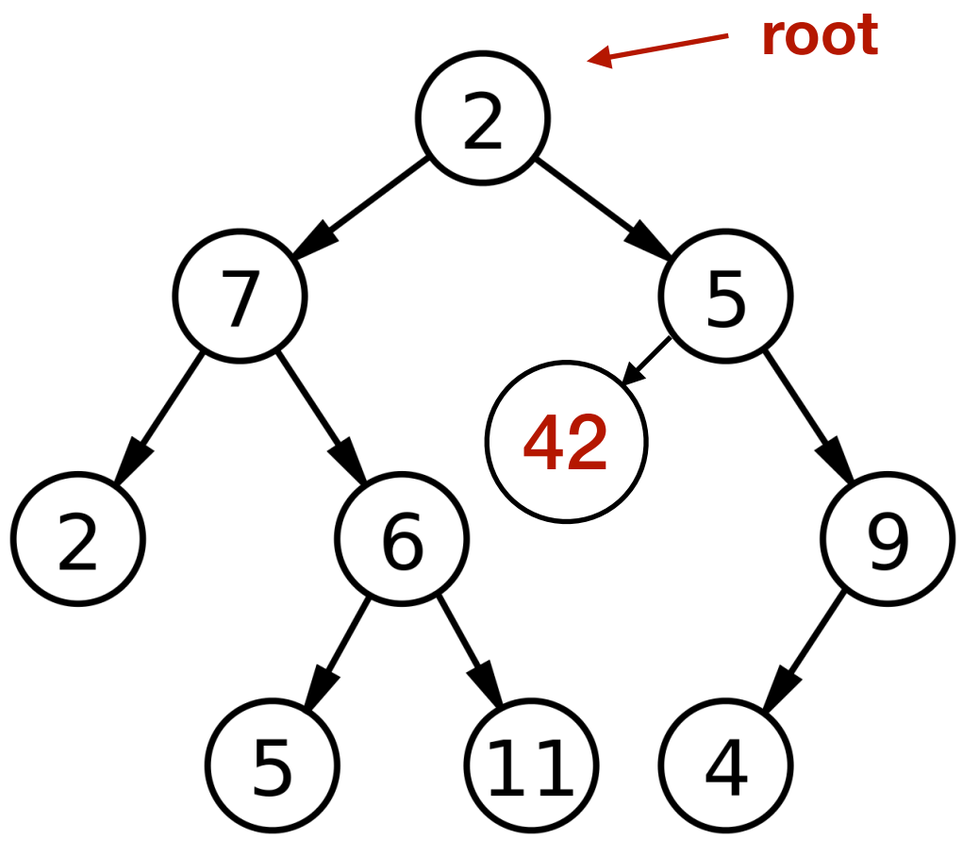
- Usually a combination of recursion (to go through each level) and iteration (to check all of your children) is the way to go.

Implementing add

Problem: We want to add a node to the tree, where we say where the node should go by specifying its intended parent.

```
public void add(int newItem, int parentItem);
```

root.add(42, 5)



Implementing add

Problem: We want to add a node to the tree, where we say where the node should go by specifying its intended parent.

```
public int item;
public TreeNode parent;
public ArrayList<TreeNode> children;

public TreeNode(int item, TreeNode parent) {
    this.item = item;
    this.parent = parent;
    this.children = new ArrayList<>();
}

public void add(int newItem, int parentItem) {









}
```

Implementing add

```
public int item;
public TreeNode parent;
public ArrayList<TreeNode> children;

public TreeNode(int item, TreeNode parent) {
    this.item = item;
    this.parent = parent;
    this.children = new ArrayList<>();
}

public void add(int newItem, int parentItem) {
    if (this.item == parentItem) {

    }




}
```

Implementing add

```
public int item;
public TreeNode parent;
public ArrayList<TreeNode> children;

public TreeNode(int item, TreeNode parent) {
    this.item = item;
    this.parent = parent;
    this.children = new ArrayList<>();
}

public void add(int newItem, int parentItem) {
    if (this.item == parentItem) {
        children.add(new TreeNode(newItem, this));
    }



}
```

Implementing add

```
public int item;
public TreeNode parent;
public ArrayList<TreeNode> children;

public TreeNode(int item, TreeNode parent) {
    this.item = item;
    this.parent = parent;
    this.children = new ArrayList<>();
}

public void add(int newItem, int parentItem) {
    if (this.item == parentItem) {
        children.add(new TreeNode(newItem, this));
    } else {



    }
}
```

Implementing add

```
public int item;
public TreeNode parent;
public ArrayList<TreeNode> children;

public TreeNode(int item, TreeNode parent) {
    this.item = item;
    this.parent = parent;
    this.children = new ArrayList<>();
}

public void add(int newItem, int parentItem) {
    if (this.item == parentItem) {
        children.add(new TreeNode(newItem, this));
    } else {
        for (TreeNode child : children) {

        }
    }
}
```

Implementing add

```
public int item;
public TreeNode parent;
public ArrayList<TreeNode> children;

public TreeNode(int item, TreeNode parent) {
    this.item = item;
    this.parent = parent;
    this.children = new ArrayList<>();
}

public void add(int newItem, int parentItem) {
    if (this.item == parentItem) {
        children.add(new TreeNode(newItem, this));
    } else {
        for (TreeNode child : children) {
            child.add(newItem, parentItem);
        }
    }
}
```