CS 2230
CS II: Data structures

Meeting 21: trees
Brandon Myers
University of Iowa
Today’s learning objectives

• Define tree (a data structure that can store data hierarchically) and describe trees in terms of attributes
• Interpret code that uses a links to represent a tree
• Interpret code that uses recursion to explore all the branches of a tree
1 minute paper

What is an example of data or real-life system that is organized hierarchically?

https://b.socrative.com/login/student/
room CS2230X ids 1000-2999
room CS2230Y ids 3000+
Your family tree

https://commons.wikimedia.org
Business management structure

Agency Department System

President

Vice President Account Services
- Account Supervisor
  - Account Executive
  - Account Executive

Vice President Creative Services
- Account Supervisor
  - Art / Copy
  - Production

Vice President Marketing Services
- Media
  - Research

Vice President Management Services
- Accounting
  - Purchasing
  - Personnel

https://commons.wikimedia.org
Classification of life

https://commons.wikimedia.org
Inheritance tree for Java exceptions

```
<table>
<thead>
<tr>
<th>X is a Y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>X</td>
</tr>
</tbody>
</table>

```

- **Runnable**
  - **Error**
  - **Exception**
    - **RuntimeException**
      - **NullPointerException**
      - **ArrayOutOfBoundException**
    - **IOException**

X is a Y

Y

X
Motivation

If we store this kind of data in a List, we’ll lose the hierarchical relationships

Your computer’s filesystem

Your family tree

Business management

Agency Department System

Classification of life

https://commons.wikimedia.org
• height

• depth

• leaves

• inner nodes

• root

• children

• parent
1. Pick one of these trees (or your example)
2. Identify what each means for the example: height, depth, leaves, inner nodes, root, children, parent
3. Repeat for another example

**Your computer’s filesystem**

**Your family tree**

**Business management**

**Classification of life**

https://commons.wikimedia.org
Our first implementation of a tree

Eventually we will want a Tree *interface* (your textbook defines one in Ch 8.1), but let’s start with a specific implementation to get the intuition

class LinkedBinaryTree<T> {
    private BinaryTreeNode<T> root;
    private class BinaryTreeNode {
        public T data;
        public BinaryTreeNode left;
        public BinaryTreeNode right;
    }
}
class LinkedBinaryTree\<T> {  
    private BinaryTreeNode\<T> root;  
    public LinkedBinaryTree() {  
        root = null;  
    }  
    
    public static void main(String[] args) {  
        LinkedBinaryTree\<Integer> tree = new LinkedBinaryTree\<>();  
        tree.root = new BinaryTreeNode\<>(2);  
        tree.root.left = new BinaryTreeNode\<>(1);  
        tree.root.left.left = new BinaryTreeNode\<>(4);  
        tree.root.left.right = new BinaryTreeNode\<>(1);  
        tree.root.right = new BinaryTreeNode\<>(3);  
    }  
}

private class BinaryTreeNode {  
    public T data;  
    public BinaryTreeNode left;  
    public BinaryTreeNode right;  
    public BinaryTreeNode(T data) {  
        this.data = data;  
        this.left = null;  
        this.right = null;  
    }  
}
Peer instruction

BinaryTreeNode<Integer> root = new BinaryTreeNode<>((1));
root.right = new BinaryTreeNode<>((2));
BinaryTreeNode<Integer> t1 = root.right;
t1.left = new BinaryTreeNode<>((3));
BinaryTreeNode<Integer> t2 = t1.left;
t2.right = new BinaryTreeNode<>((4));

https://b.socrative.com/login/student/
room CS2230X ids 1000-2999
room CS2230Y ids 3000+
Recursion on binary trees. What does `getXElement()` return when called on this tree?

class LinkedBinaryTree<T> { 
    private BinaryTreeNode<T> root;

    private T getXElement() {
        return root.getXNode().data;
    }

    private class BinaryTreeNode {
        public T data;
        public BinaryTreeNode left;
        public BinaryTreeNode right;
        private BinaryTreeNode<T> getXNode() {
            if (this.left == null) return this;
            else return this.left.getXNode();
        }
    }
}

https://b.socrative.com/login/student/
room CS2230X ids 1000-2999
room CS2230Y ids 3000+
Peer instruction

What is the **worst case** running time of LinkedBinaryTree.getXElement()?

What is the **best case** running time of LinkedBinaryTree.getXElement()?

a) $O(1)$  
b) $O(\log N)$  
c) $O(N)$  
d) $O(N\log N)$  
e) $O(N^2)$

```java
private BinaryTreeNode<T> getXNode() {
    if (this.left == null) return this;
    else return this.left.getXNode();
}
```

https://b.socrative.com/login/student/
room CS2230X ids 1000-2999
room CS2230Y ids 3000+
The real power of recursion on trees

class LinkedBinaryTree<T> {
    private BinaryTreeNode root;

    // only works on LinkedBinaryTree<Integer>
    public int sum() {
        return root.sum();
    }
}

private class BinaryTreeNode {
    public T data;
    public BinaryTreeNode left;
    public BinaryTreeNode right;
    private int sum() {
        int s = 0;
        if (this.left != null) s += this.left.sum();
        if (this.right != null) s += this.right.sum();
        //only works when T is Integer
        s += (int) this.data;
        return s;
    }
}

List the order of BinaryTreeNodes that have sum called on them

https://b.socrative.com/login/student/
room CS2230X ids 1000-2999
room CS2230Y ids 3000+
Peer instruction

```java
private int sum() {
    int s = 0;
    if (this.left != null) s += this.left.sum();
    if (this.right != null) s += this.right.sum();
    //only works when T is Integer
    s += (int) this.data;
    return s;
}
```

What is the running time of `BinaryTreeNode.sum()`?

a) O(1)  
b) O(logN)  
c) O(N)  
d) O(NlogN)  
e) O(N^2)
Today’s learning objectives

• Define *tree* (a data structure that can store data hierarchically)
• Interpret code that uses a links to represent a tree
• Interpret code that uses recursion to explore all the branches of a tree