CS367 Announcements
Wed, July 17th, 2013

- P2 due Today, Wed 11:59pm
- H5 due Mon 6pm

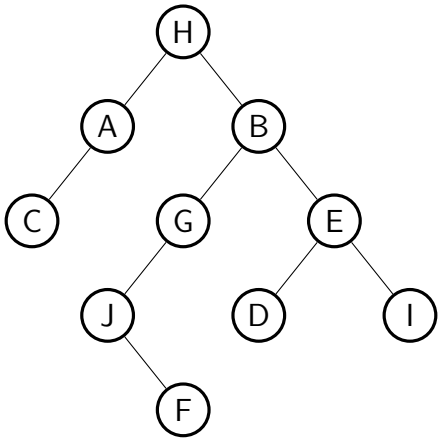
Last Time

- Finish Recursion
- Intro to Search
- Intro to Trees

Today

- Trees (Cont.)

Tree Terminology



1. What is the **root**?
2. How man **leaves** are there?
3. What is the **height** of the tree?
4. What is the **depth** of J?
5. How many **children** does G have (**degree** of G)?
6. How many **decendents** does B have?
7. What are the **ancestors** of D?
8. What is the **length** of the **path** from B to D?
9. What are the **subtrees** of B?

General Tree Implementation

Tree nodes:

```
class Treenode<T> {
    private T data;
    private ListADT<Treenode<T>> children;
    ...
```

Tree:

```
class Tree<T> {
    private Treenode<T> root;
    private int size;

    public Tree() {
        root = null;
        size = 0;
    }
    ...
```

Determining Height of a General Tree

```
public int height() {
```

Binary Tree Implementation

Tree nodes:

```
class BinaryTreenode<T> {
    private T data;
    private BinaryTreenode<T> leftChild;
    private BinaryTreenode<T> rightChild;

    public BinaryTreenode(T item) {
        data = item;
        leftChild = null;
        rightChild = null;
    }
    ...
```

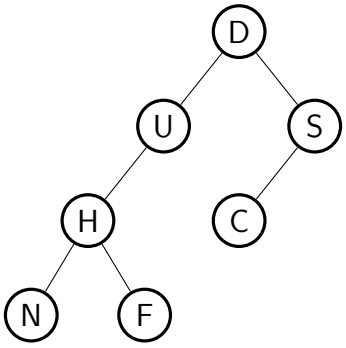
Tree:

```
public class BinaryTree<T> {
    private BinaryTreenode<T> root;
    private int size;

    public BinaryTree() {
        root = null;
        size = 0;
    }
    ...
```

Tree Traversals

Goal: visit every node in the tree exactly once



- Level-order

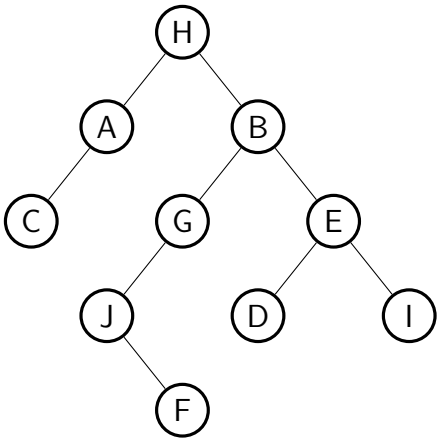
- Pre-order

- Post-order

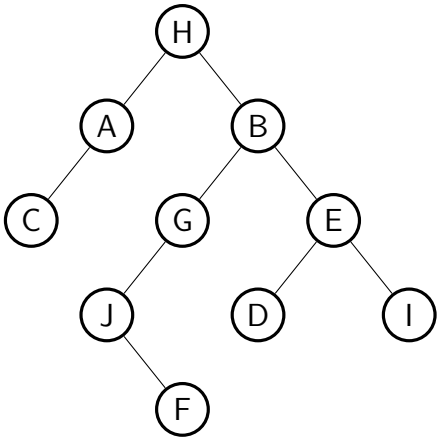
- In-order

Tree Traversals Practice

1. List nodes using a **pre-order traversal**:



2. List nodes using a **post-order traversal**:



3. List nodes using an **in-order traversal**: