CS367 Lecture 16

Monday 14 July 2014

Announcements/Reminders:

- Midterm solutions
- HW4 due today
- HW5 assigned
- Readings

Last class:

- Recursion (end)
- Search
- Trees (intro)

Today:

- (Last) Week in Review
- Trees (cont'd)
    - Binary Trees

(Last) Week in Review

- Recursion
    - What and Why
    - Key Questions
    - Examples
    - Analyzing Complexity of Recursive Functions
- Searching
- Trees
    - Intro
    - Implementing
    - Recursive `height()` method

Tree Data Structures

Tree Terminology

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

Implementing Trees (general)

(Tree) Nodes:

```
_________ class TreeNode<E> {

    private E data;

    private _____________<_____________> children;

    ...
```

Tree:

```
public class Tree<E> {
    private TreeNode<E> root;

    ...


    public Tree() {
        root = null;
        ...
    }

    ...
}
```

Working with Trees: Example

Write a method to determine the height of a general tree. (What is the recursive definition?)

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

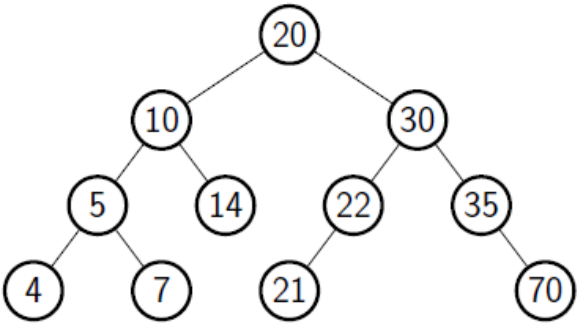
Binary Trees

BinaryTreeNode class:

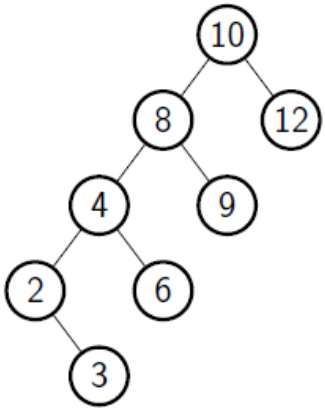
```
class BinaryTreeNode<T> {
    private T data;
    private BinaryTreeNode<T> parent;
    private BinaryTreeNode<T> leftChild;
    private BinaryTreeNode<T> rightChild;

    public BinaryTreeNode(T info) {
        data = info;
        leftChild = null;
        rightChild = null;
    }
    ...
}
```

Types of binary trees

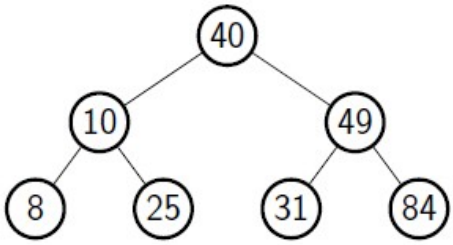
(A)



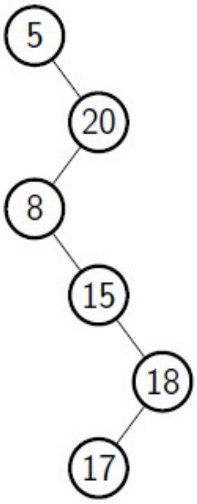
(B)



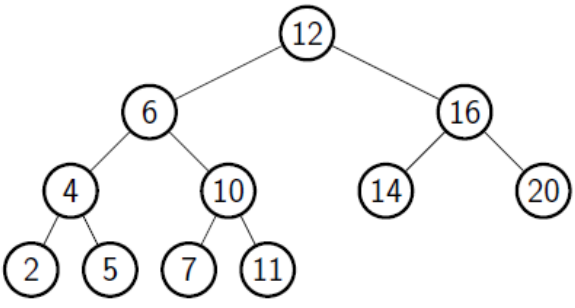
(C)



(D)



(E)



(F)