Announcements

MP4 available, due 10/17, 11:59p.

Perfect Binary tree -

Perfect tree of height \( h \), \( P_h \):
- if \( h = -1 \), then \( P_h \) is \( \{\} \)
- if \( h > -1 \), then \( P_h \) is \( \{r, T_L, T_R\} \), where \( T_L \) and \( T_R \) are \( P_{h-1} \).

\[
\begin{align*}
P_0 & : \\
P_2 & : \\
P_1 & : 
\end{align*}
\]
Complete Binary tree: for any level $k$ in $[0, h-1]$, level $k$ has $2^k$ nodes, and on level $h$, all nodes are pushed to the left.

Complete tree of height $h$, $C_h$:
- if $h = -1$, then $C_h$ is $\{\}$
- if $h > -1$, then $C_h$ is $\{r, T_L, T_R\}$, and either:
  - $T_L$ is _______ and $T_R$ is _______
  - OR
  - $T_L$ is _______ and $T_R$ is _______

http://xlinux.nist.gov/dads//HTML/completeBinaryTree.html

Check for understanding:
Is every full tree complete?
Is every complete tree full?
Rooted, directed, ordered, binary trees

Tree ADT:
- insert
- remove
- traverse

```cpp
template <class T>
class tree{
public:
...
private:
    struct treeNode{
        T data;
        treeNode * left;
        treeNode * right;
    };
    treeNode * root
...}
```
Theorem: if there are $n$ data items in a binary tree, then there are ____ null pointers.
Traversal — scheme for processing all the data in the structure...
Traversal – scheme for visiting every node.
Traversals – scheme for visiting every node.

- At each node, two choices for direction (left, right)
- After both subtrees of a node are complete, move back up tree
- Each node is “visited” 3 times in a traversal.
- Each of those visit times corresponds to a particular kind of traversal.

http://nova.umuc.edu/~jarc/idsv/lesson1.html
Traversals:

```
template<class T>
void binaryTree<T>::_______Order(treeNode * croot){
    if (croot != null){
        _______Order(croot->left);
        _______Order(croot->right);
    }
}
```

```
        +
       / \
      -   /
     /   c
    /   /  \
   b   d   e
```