course_content

top





## Binary Trees

# Week 8

Time:
1 min

# Trees

- A tree is a **non-linear** data structure
- It has a **root, branches** and **leaves**
- The **root** (only one) is at the **top,** branches come down with leaves at the bottom
- Each item in a tree is called a **node**
- Examples: Animal Kingdom Hierarchy, Family Tree, File System, XML/HTML data, Heaps, compilers (for managing syntax),

Time:
1 min

# Trees

- What do the following mean in the context of trees?
  - Node, Edge, Root, Path, Children, Parent
  - Sibling, Subtree, Leaf Node, Level, Height
- You can think of a tree in 2 ways:
  - A root + one-way edges coming out of each node to new nodes such that there are no ‘cycles’ ---- the intuitive understanding
  - A tree is either empty or a root connecting to zero or more subtrees (which are also trees) ----- the recursive definition

Time:
2 min

# Trees

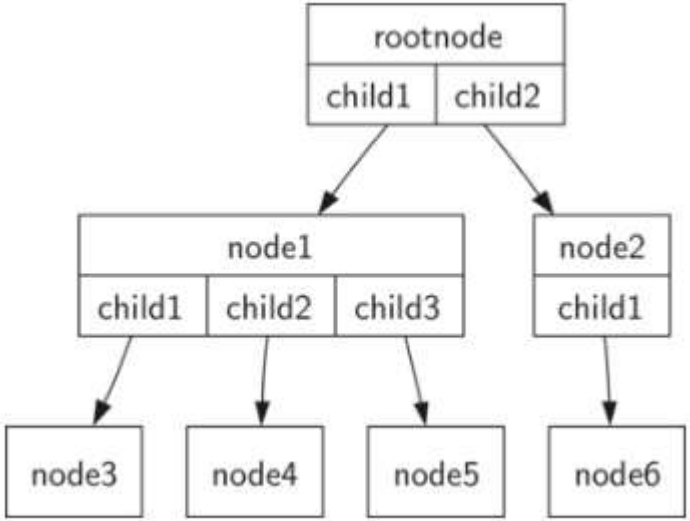


Figure 3: A Tree Consisting of a Set of Nodes and Edges

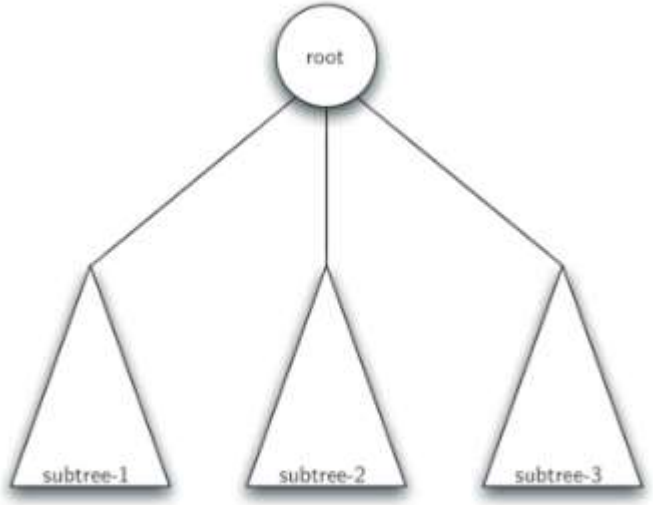


Figure 4: A recursive Definition of a tree

Time:
5 min

# Trees - List of Lists

- List of List Implementations
  - E.g. [3,[1],[2,[1],[3]]]
  - E.g. [3,[1,[2,[4,[6]]]]]
  - E.g. [3]
  - E.g. [] (if empty trees are allowed)
- Are all list of lists trees?
- How would you parse through a list to check if it is in fact a tree?
- Advanced - how would you check it is a binary tree?
- (Write pseudocode to do this)

Time:
15 min

# Trees - Implementation 2

- Similar to a linked list
- One needs to define a node class which contains a few key attributes:
  - Parent (to go back up tree), Value (to see what value is at that node)
  - Children (collective data structure of your choice)
  - Left, Right - For binary tree instead of children
- A tree is then just a pointer to the root node.
- What are the primary node operations?
- What are the primary tree operations?
- Write pseudocode to find the height of a tree.

Time:
5 min

# Trees - Traversals

- There are many different traversal methods:
  - Depth-First Search
    - Pre-order traversal
    - In-order traversal
    - Post-order traversal
  - Breadth-First Search
- What do they mean? How do they work?
- Can you write the method recursively?
- Can you write the method iteratively? Using other DS to help you.

Time:
15 min

# Trees - Binary Search Tree

- We previously looked at sorted arrays and found that maintaining them was hard
- A binary search tree can be a better way of doing this
- Values are inserted into their correct position. Smaller than root means left sub-tree, larger than root means right sub-tree.
- Construct a BST (binary search tree) with the following 5,10,7,8,1,3,6.
- What are the runtimes of the following in a BST?
    - Insert, Search, Delete, Create (with a list of given values)

Time:
15 min

# Trees - AVL Tree

- The operations can in the worst case be very long if they are inputted poorly.
- E.g. 10,9,8,7,6,5 will lead to a one-sided BST
- $balanceFactor = height(leftSubTree) - height(rightSubTree)$

- An AVL Tree has balanceFactor 1,0 or -1 (for all roots)
- Once the balance factor leaves this range, it has a procedure to fix it.
- This a fairly involved procedure that you can read about but it involves a ‘rotation’ of your tree by changing the root and one of the branches.

Time:
1 min

# Questions?