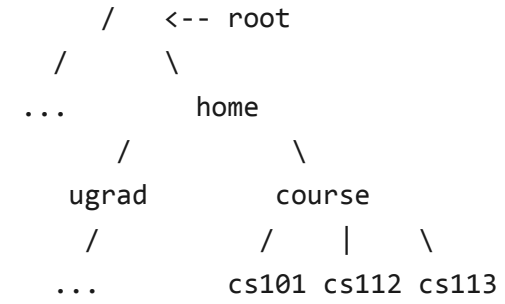
Applications of tree data structure

Difficulty Level: Rookie

Why Tree?

Unlike Array and Linked List, which are linear data structures, tree is hierarchical (or non-linear) data structure.

1. One reason to use trees might be because you want to store information that naturally forms a hierarchy. For example, the file system on a computer:
   file system
   ————



2. If we organize keys in form of a tree (with some ordering e.g., BST), we can search for a given key in moderate time (quicker than Linked List and slower than arrays). Self-balancing search trees like AVL and Red-Black trees guarantee an upper bound of O(Logn) for search.
3. We can insert/delete keys in moderate time (quicker than Arrays and slower than Unordered Linked Lists). Self-balancing search trees like AVL and Red-Black trees guarantee an upper bound of O(Logn) for insertion/deletion.
4. Like Linked Lists and unlike Arrays, Pointer implementation of trees don't have an upper limit on number of nodes as nodes are linked using pointers.

Other Applications :

1. Heap is a tree data structure which is implemented using arrays and used to implement priority queues.

2. B-Tree and B+ Tree : They are used to implement indexing in databases.
3. Syntax Tree: Used in Compilers.
4. K-D Tree: A space partitioning tree used to organize points in K dimensional space.
5. Trie : Used to implement dictionaries with prefix lookup.
6. Suffix Tree : For quick pattern searching in a fixed text.

As per Wikipedia, following are the common uses of tree.

1. Manipulate hierarchical data.
2. Make information easy to search (see tree traversal).
3. Manipulate sorted lists of data.
4. As a workflow for compositing digital images for visual effects.
5. Router algorithms

References:

http://www.cs.bu.edu/teaching/c/tree/binary/

http://en.wikipedia.org/wiki/Tree_%28data_structure%29#Common_uses

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recommended Posts:

Tango Tree Data Structure

Applications of Minimum Spanning Tree Problem

Advantages of Trie Data Structure

Overview of Data Structures | Set 2 (Binary Tree, BST, Heap and Hash)

Complexity of different operations in Binary tree, Binary Search Tree and AVL tree

Print Binary Tree levels in sorted order | Set 3 (Tree given as array)

Convert an arbitrary Binary Tree to a tree that holds Children Sum Property

Given level order traversal of a Binary Tree, check if the Tree is a Min-Heap

Check if a given Binary Tree is height balanced like a Red-Black Tree

Convert a given Binary tree to a tree that holds Logical AND property

Check whether a binary tree is a complete tree or not | Set 2 (Recursive Solution)

Sub-tree with minimum color difference in a 2-coloured tree

Convert a Binary Tree into its Mirror Tree

Check if the given binary tree has a sub-tree with equal no of 1's and 0's | Set 2

Minimum swap required to convert binary tree to binary search tree

Article Tags : Tree Self-Balancing-BST

Practice Tags : Tree

4

To-do Done

1.3

Based on **223** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments Share this post!

A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY

About Us
Careers
Privacy Policy
Contact Us

LEARN

Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE

Company-wise
Topic-wise
Contests
Subjective Questions

CONTRIBUTE

Write an Article
Write Interview Experience
Internships
Videos