VISION & MISSION

VISION

To build strong teaching environment that responds the need of industry and challenges of society.

Mission

Developing strong mathematical & computing fundamentals among the students.

Extending the role of computer science and engineering in diverse areas.

Imbibing the students with a deep understanding of professional ethics and high integrity to serve the nation.

Providing an environment to the students for their growth both as individuals and as globally competent Computer Science professional.

Outreach activities will contribute to the overall wellbeing of society.

Learning Objectives

Understand the basic terminology of trees
Understand the representation of binary trees in memory
Explain the types of binary trees
Demonstrate the concept of array and linked representation of binary trees
Explain various tree traversal methods
Explain threaded binary trees and their traversal
Explain the concept of Huffman algorithm

SYLLABUS

UNIT I

Introduction:

Basic Terminology, Elementary Data Organization, Algorithm, Efficiency of an Algorithm, Time and Space Complexity, Asymptotic notations: Big-Oh, Time -Space trade-off. Abstract Data Types (ADT), Arrays: Definition, Single and Multidimensional Arrays, Representation of Arrays: Row Major Order, and Column Major Order, Application of arrays, Sparse Matrices and their representations.

Linked lists: Array Implementation and Dynamic Implementation of Singly Linked Lists, Doubly Linked List, Circularly Linked List, Operations on a Linked List. Insertion, Deletion, Traversal, Polynomial Representation and Addition, Generalized Linked List

.

UNIT II

Stacks: Abstract Data Type, Primitive Stack operations: Push & Pop, Array and Linked Implementation of Stack in C, Application of stack: Prefix and Postfix Expressions, Evaluation of postfix expression, Recursion, Tower of Hanoi Problem, Simulating Recursion, Principles of recursion, Tail recursion, Removal of recursion Queues, Operations on Queue: Create, Add, Delete, Full and Empty, Circular queues, Array and linked implementation of queues in C, Dequeue and Priority Queue.

UNIT III

Trees: Basic terminology, Binary Trees, Binary Tree Representation: Array Representation and Dynamic Representation, Complete Binary Tree, Algebraic Expressions, Extended Binary Trees, Array and Linked Representation of Binary trees, Tree Traversal algorithms: Inorder, Preorder and Postorder, Threaded Binary trees, Traversing Threaded Binary trees, Huffman algorithm.

UNIT IV

Graphs: Terminology, Sequential and linked Representations of Graphs: Adjacency Matrices, Adjacency List, Adjacency Multi list, Graph Traversal : Depth First Search and Breadth First Search, Connected Component, Spanning Trees, Minimum Cost Spanning Trees: Prims and Kruskal algorithm. Transitive **Closure and Shortest Path algorithm: Warshal Algorithm and Dijikstra Algorithm, Introduction to Activity Networks**

.

UNIT V

Searching : Sequential search, Binary Search, Comparison and Analysis Internal Sorting: Insertion Sort, Selection, Bubble Sort, Quick Sort, Two Way Merge Sort, Heap Sort, Radix Sort.

Presentation Outline

- Basic terminologies of Trees
- Representations of Binary trees in Memory
- Traversal of Binary Trees
- Threaded Binary tree
- Huffman algorithm

Terminologies of Trees

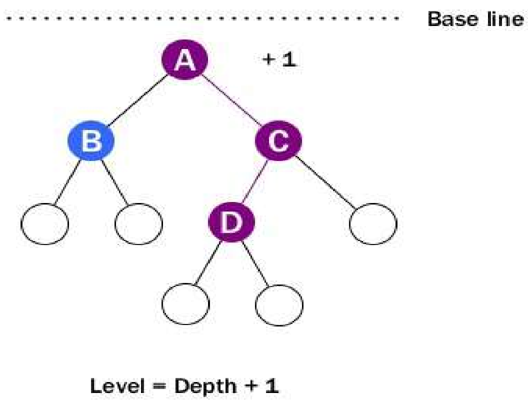
- Trees: Its is a non –linear data structure which is used to represent data containing a heirarchical relationship between two elements. E.g Records, family tree
- Binary trees: A binary tree T is defined as a finite set of elements called nodes such that the:

a)T is empty called null tree or empty tree

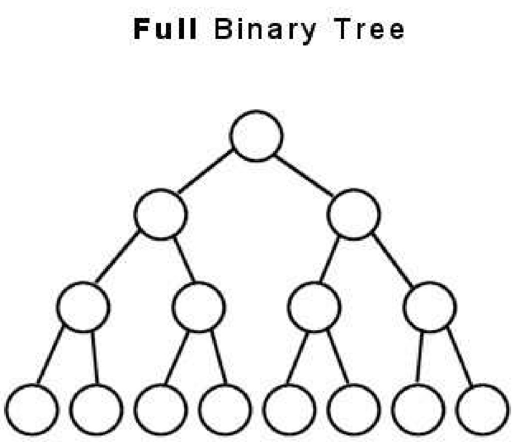
b) If T contains a distinguishable node R called root of the tree T and remaining node form an ordered pair of disjoint binary trees T1 and T2.

- Complete binary tree: A Binary tree is a complete binary tree , if all its level, except possibly the last have the maximum number of possible nodes at the last level appear as far left as possible.
- Extended Binary Tree: A binary tree is said to be extended binary tree if each node has either 0 or 2 children. The node with two children are called internal nodes represented by circle. And the nodes with 0 children are called external nodes represented by square

- Degree of a Tree: It is the maximum degree of nodes in a given tree .
- Level of a Tree: The level of a node is defined by 1 + the number of connections between the node and the root



- Full Binary Tree: A full binary tree (sometimes proper binary tree or 2-tree) is a tree in which every node other than the leaves has two children.

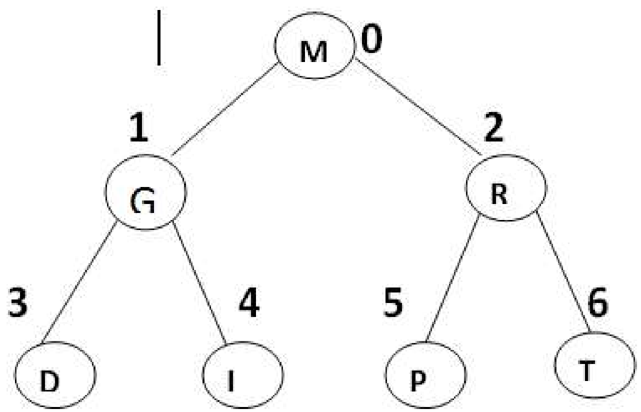


Representation of Binary Tree in Memory

There are two types of representation of binary tree in memory:

- **Array Representation:**
- In array representation binary tree is represented sequentially in memory by using single dimensional array. A binary tree of height n may comprise at most 2(n+1)-1 nodes, hence an array of maximum size 2(n+1)-1 is used for representing such a tree. All the nodes of the tree are assigned a sequence number (from 0 to (2(n+1)-1)-1) level by level.

Example of Array representation

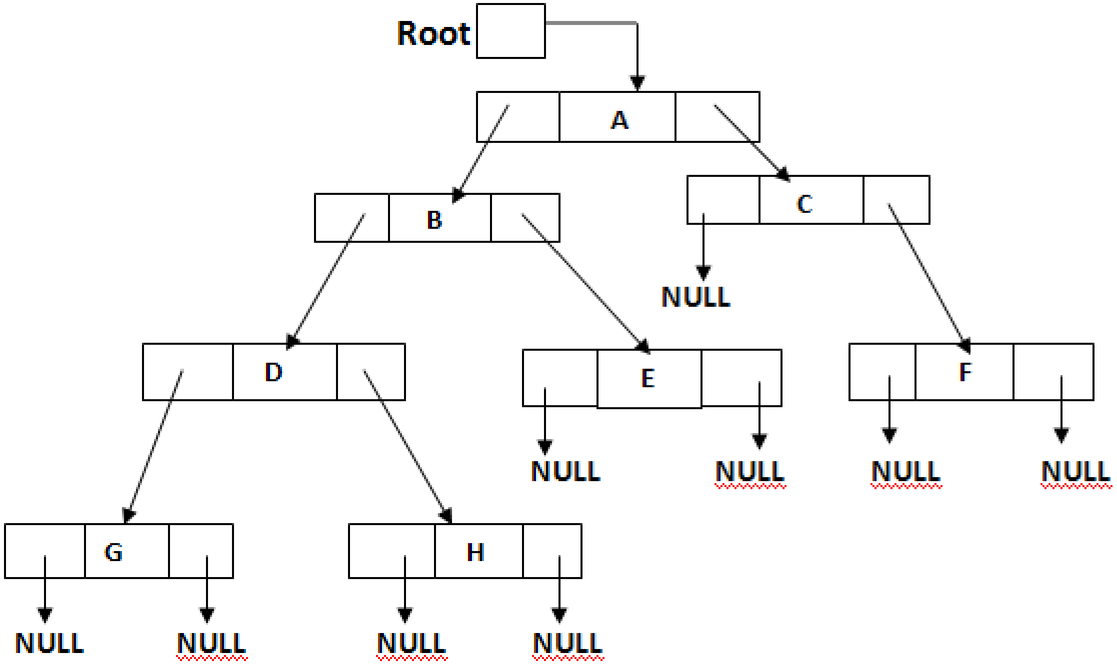


0	1	2	3	4	5	6
M	G	R	D	I	P	T

Nodes of Binary Tree Stored in an Array

- **Linked Representations:** The linked representation of a binary tree is implemented by using a linked list having info part and two pointers. The info part contains the data value and two pointers, left and right are used to point to the left and right sub tree of a node, respectively.

Example of Linked Representaions



Tree Traversals:

- **Preorder:**

i. Process the root ‘r’.
ii. Traverse the left subtree of ‘r’ in preorder.
iii. Traverse the right subtree of ‘r’ in preorder.

- **Postorder:**

i. Traverse the left subtree of ‘r’ in preorder.
ii. Traverse the right subtree of ‘r’ in preorder.
iii. Process the root ‘r’.

Inorder:

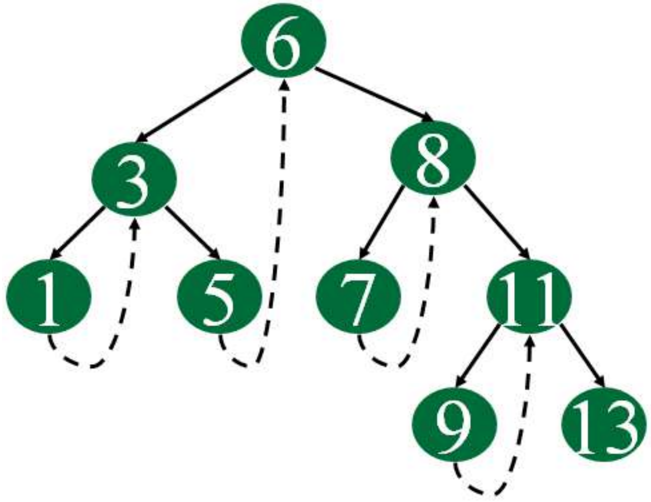
iv. Traverse the left subtree of ‘r’ in preorder.
v. Process the root ‘r’.
vi. Traverse the right subtree of ‘r’ in preorder.

Threaded Binary Tree

- **The idea of threaded binary trees is to make inorder traversal faster and do it without stack and without recursion. A binary tree is made threaded by making all right child pointers that would normally be NULL point to the inorder successor of the node (if it exists).**

- **There are two types of threaded binary trees:**
- **Single Threaded: Where a NULL right pointers is made to point to the inorder successor (if successor exists)**
- **Double Threaded: Where both left and right NULL pointers are made to point to inorder predecessor and inorder successor respectively. The predecessor threads are useful for reverse inorder traversal and postorder traversal**
  .
- **The threads are also useful for fast accessing ancestors of a node.**

Example of Threaded Binary Tree



Huffman Algorithm

1. Create a leaf node for each unique character and build a min heap of all leaf nodes (Min Heap is used as a priority queue. The value of frequency field is used to compare two nodes in min heap. Initially, the least frequent character is at root)

2. Extract two nodes with the minimum frequency from the min heap.

3. Create a new internal node with frequency equal to the sum of the two nodes frequencies. Make the first extracted node as its left child and the other extracted node as its right child. Add this node to the min heap.

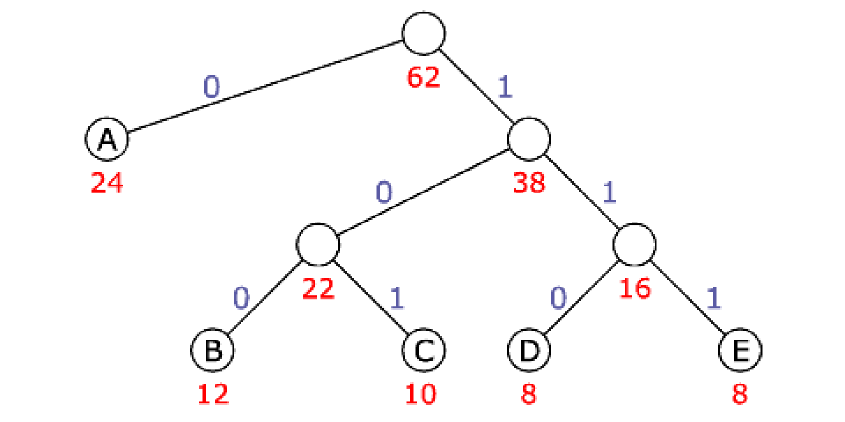
4. Repeat steps 2 and 3 until the heap contains only one node. The remaining node is the root node and the tree is complete.

- Symbol Frequency

Symbol	Frequency
A	24
B	12
C	10
D	8
E	8

Solve the above using huffman algorithm

Tree obtained



Code table

Symbol	Frequency	Code	Code Length	total Length
A	24	0	1	24
B	12	100	3	36
C	10	101	3	30
D	8	110	3	24
E	8	111	3	24

----

ges. 186 bit   tot. 138 bit

ASSIGNMENT

Q1. Two Binary Trees are similar if they are both empty or if they are both nonempty and left and right sub trees are similar. Write an algorithm to determine if two Binary Trees are similar.

Q2. What are expression trees? Represent the following expression using a tree. Comment on the result that you get when this tree is traversed in Preorder, Inorder and postorder. (a-b) / ((c*d)+e).

Q3. How do you rotate a Binary Tree? Explain right and left rotations with the help of an example.

- Q4. Construct a binary tree whose nodes in inorder and preorder are given as follows:
- Inorder : 10, 15, 17, 18, 20, 25, 30, 35, 38, 40, 50

  Preorder: 20, 15, 10, 18, 17, 30, 25, 40, 35, 38, 50
- Q5. Given the following inorder and preorder traversal reconstruct a binary tree

  Inorder sequence D, G, B, H, E, A, F, I, C

  Preorder sequence A, B, D, G, E, H, C, F, I
- Q6. What is a Binary Tree? What is the maximum number of nodes possible in a Binary Tree of depth d. Explain the following terms with respect to Binary trees (i) Strictly Binary Tree (ii) Complete Binary Tree (iii) Almost Complete Binary Tree.
- Q7. Construct the binary tree for the following sequence of nodes in preorder and inorder respectively.

  Preorder : G, B, Q, A, C, K, F, P, D, E, R, H

  Inorder: Q, B, K, C, F, A, G, P, E, D, H, R

  Q8. Let a binary tree 'T' be in memory. Write a procedure to delete all terminal nodes of the tree.

  Q9. What is the maximum total number of nodes in a tree that has N levels? Note that the root is level (zero).

TUTORIAL

1 What is the maximum number of nodes at level 'h' of a binary tree is 2^{h-1}?

2 In a Binary Tree with N nodes, minimum possible height or minimum number of levels is ceiling (Log_2 (N+1)). Prove.

3 How to construct tree from In-order sequence: D B E A F C and Preorder sequence: A B D E C F?

4 What is complete binary tree and how is it different from almost complete binary tree?

Tutorial contd..

5 Create min heap tree with keys: 72, 52, 45, 12, 65, 72, 125, 64, 1, 15, 25, 100, 167, 144 and 27.

6 How many different binary trees are possible with n distinct keys?

7 What will be the Huffman codes for the values given below:

A= 0.45
B= 0.11
C= 0.02
D= 0.24
E= 0.18

Course Outcome

The basic concept of trees, their representation in memory and traversal techniques	Illustrate basic terminology of trees, tree traversal methods and solve numerical related to Huffman algorithm.

<u>THANK YOU</u>