Tree is an abstract model of a hierarchical structure.

consists of nodes with a parent-child relationship.

Applications - File system.

**//Node: a** data structure for storing data

**//Root**: node without the parent

**//Siblings**: nodes share the same parent

**//Internal node**: node with at least one child (A, B, C, F)

**//External node** (leaf ): node without children (E, I, J, K, G, H, D)

**//Ancestors** of a node: parent, grandparent, grand-grandparent, etc.

**//Descendant** of a node: child, grandchild, grand-grandchild, etc.

**//Depth** of a node: number of ancestors

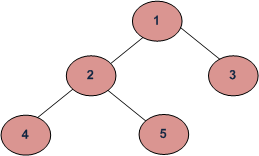
**//Height** of a tree: maximum depth of any node (3)

**//Degree** of a node: the number of its children

**//Degree** of a tree: the maximum number of its node.

**//Subtree**: tree consisting of a node and its descendants

Tree Traversal :



Depth First Traversals:

(a) Inorder (Left, Root, Right) : 4 2 5 1 3

(b) Preorder (Root, Left, Right) : 1 2 4 5 3

(c) Postorder (Left, Right, Root) : 4 5 2 3 1

Breadth-First or Level Order Traversal: 1 2 3 4 5

Types of Trees:

General Trees - 1 Root, Multiple children.

Binary Trees

Each internal node has at most two children (degree of two)

subtrees are ordered

Types-

Strictly Binary Tree, Partial Binary Tree

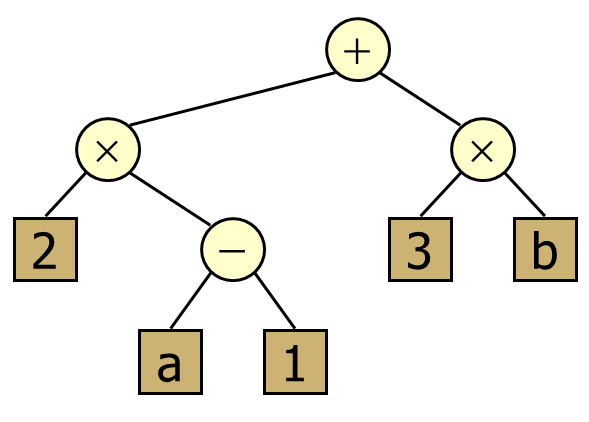
Skewed - one child or no child.

Node has - data, parent node, right, and left child node.

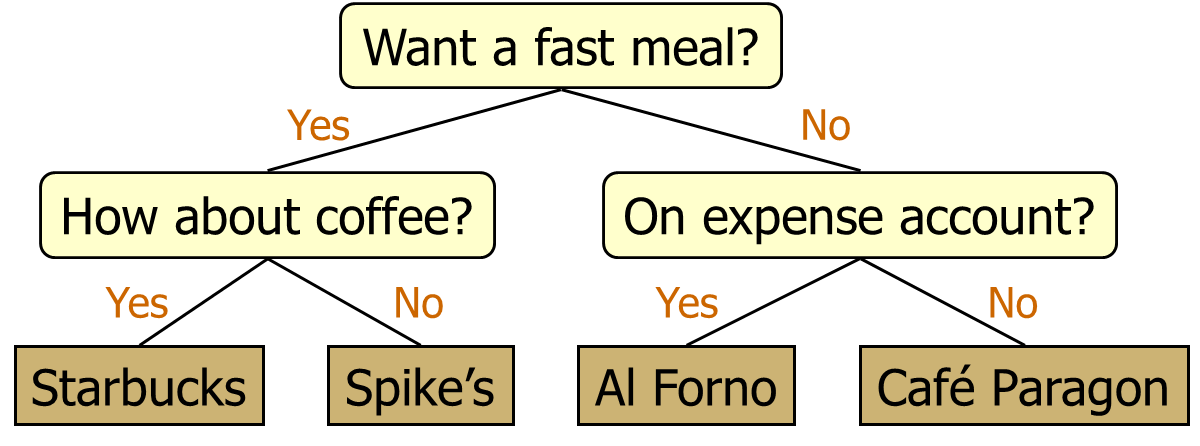
Applications:

arithmetic expressions (In-order)

(2 ´ (a - 1) + (3 ´ b))



decision processes (Preorder)



searching (Binary Search Algorithm)

maximum number of nodes on depth i = 2i

maximum nubmer of nodes on height k, 2k+1-1

Expression Tree

BINARY SEARCH TREE

The value stored at a node is greater than the value stored at its left child and less than the value stored at its right child

AVL - Adelson-Velskii + Landis

height-balanced binary search trees

balance factor = height of left subtree - right

balanced if BF of all nodes is btw [-1 , 1]

For every node, heights of left and right subtrees can differ by no more than 1

If the AVL tree property is violated at node x, single or double rotation will be applied to x to restore the AVL tree property.

4 types of rotations-

LL , RR ,LR ,RL

B-TREE

is a special type of self-balancing tree in which each node can contain more than one key and can have more than two children. It is a generalized form of the binary search tree. It is also known as a height-balanced m-way tree.

self-balancing m-way tree

number of keys in each non-leaf node is one less than the number of its children

order = max children = m

keys (data) = m-1 , child nodes = m,

nodes have minimum [m/2] children.

**B+ Tree:** A B+ Tree is an advanced form of a self-balancing tree in which all values are present at the leaf level. (uses multi leveling indexing)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is Graph?

a collection of vertices (also called nodes) and edges that connect these vertices.

Types: Directed Graph(Digraph), Un-Directed Graph.

There are two common ways of storing graphs in a computer’s memory.

They are:

Sequential representation by using an adjacency matrix.

Linked representation by using an adjacency list

that stores the neighbors of a node using a linked list.

An adjacency list is preferred for representing sparse graphs in a computer’s memory; otherwise, an adjacency matrix is a good choice.

Breadth-first search (BFS) is a graph search algorithm that begins at the source node and explores all the neighboring nodes.

Then for each of those neighboring nodes, the algorithm explores their unexplored neighbor nodes, and so on, until it finds the goal node.

The QUEUE data structure is used to hold the nodes that are waiting to be processed and variable STATUS is used to represent the current status of the node.

BFS Algorithm:

1. Select any random node

2. Add the selected node to queue and list of BFS traversal

3. Add all nodes of a selected node from the adjacency matrix to the queue and the list

of BFS traversal

4. After adding all the nodes from the adjacency list, delete them from the rear in the

queue

5. Consider rear node and add all the adjacent node

6. Repeat the process till all nodes are visited

Depth- First Search algorithm begins at a starting node A which becomes the current node. Then it examines each node N along with a path P which begins at A. That is, we process a neighbor of A, then a neighbor of A, and so on. During the execution of the algorithm, if we reach a path that has a node N that has already been processed, then we backtrack to the current node. Otherwise, the un-visited (un-processed) node becomes the current node.

DFS Algorithm:

1. Select any random node

2. Push the node to the stack and the list of DFS traversal

3. Go to any one of the nodes in the adjacency list of the selected node

4. Repeat this process till all the connections are visited

5. Then pop from the stack

6. Trace the other node in the adjacency list of the top node

7. Repeat till the stack is empty

Application of BFS and DFS:

1. Social Networking Websites: In social networks, we can find people within a given distance ‘k’ from a person using Breadth-First Search till ‘k’ levels.

2. Path Finding We can either use Breadth-First or Depth First Traversal to find if there is a path between two vertices

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Recursion** is an implicit application of a stack. The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called a recursive function.

**Types of Recursion:** Direct, Indirect, Linear, Tree, Tail.

The **Tower of Hanoi** is one of the main applications of recursion. It says, "if you can solve n-1 cases, then you can easily solve the nth case"

The process of locating target data in the list of elements is known as **searching.**

**Linear Search:** simple method of searching a value in an array ( in a sequence)  
It is mostly used to search for a value in an unordered list of elements.

**Binary Search:** Binary search is a searching algorithm that works efficiently in a sorted list.

In the Binary search technique, the list to be searched is divided every time into two lists and lists, and the search is done in only one of the lists. First, Compare Key to be searched with the elements at the middle position, and if it’s found search is successful. Else if the key < middle position element then the search key is in the first half of the list. Else if the key > middle position element then the search key is in the second half of the list. The same process is repeated for one of the halves of the list till the key is found.