Data Structure

Adapted By Manik Hosen

Basic Terminology

- Question: What is Data Structure?
- Ans: The logical or mathematical model of particular organization of data is called Data Structure.
- Question: What are the difference between linear and nonlinear data?
- Ans:

Linear Data Structure	Nonlinear Data Structure
1. Data Structures which are linearly connected.	1. Data Structures which are not linearly connected but linked.
2. Example: Array.	2. Example: Graph.
3. Easy to find any data.	3. Hard to find any data.

Depth-First Search

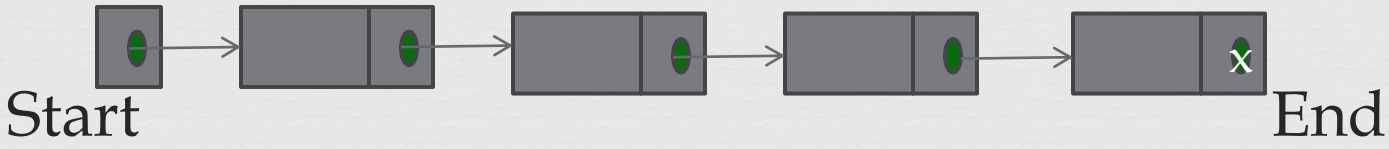
- Question: Explain the depth-first search technique.
- Ans: The general idea behind a depth-first search beginning at a starting node A is as follows. First we examine the starting node A. Then we examine each node N along a path P which begins at A; that is, we process a neighbor of A, then a neighbor of a neighbor of A, and so on. After coming to a "dead end," that is, to the end of the path P, we backtrack on P until we continue along another, path P'. And so on. Again, a field STATUS is used to tell us the current status of a node.

Shorting

- Question: What do you mean by internal and external shorting? List some names of each type.
- Ans:
- Internal Shorting is a shorting process in which all the data to short is stored in memory at the times while shorting is in progress. As example: Bubble Short, Insertion Short, Quick Short, Heap Short, Radix Short, Selection Short.
- External Shorting is a shorting process in which data is stored outside memory and only loaded into memory in small chunks. As example: Distribution shorting, merge shorting.

Linked List

- Question: What is linked list? Discuss with example.
- Ans: A linked list is a linear collection of data elements, called nodes, where the linear order is given by means of pointers.
- In the diagram there are some nodes. Each node has two parts. Left part represents the information part. Righty part represents the address part.



Data Structure

- Question: Suppose 20 elements are maintained by array and another 10 are by linked list. Which method will take longer time to access 9^{th} element? Justify your answer.
- Answer: The Linked List method will take longer time to access 9^{th} element.
- Because in Array we can directly access the 9^{th} element. But in linked list we cannot directly access 9^{th} element, instead of that we need to go through all elements before 9^{th} element.

Linked List

- Question: Briefly discuss inserting mechanism of an item at the beginning, after a given node and at the end.
- Ans:
- At the Beginning:
  a) Checking to see if space is available in the list.
  b) Removing the first node from the list and keep track of the location of new node.
  c) Copying new information into the new node.

Linked List

- After a given node:
  a) Checking to see if space is available in the list.
  b) Removing the first node from the list and keep track of the location of new node.
  c) Saving address of next node, Copying new information into the new node, Setting the location of next node at new node.
- At the End:
  a) Checking to see if space is available in the list.
  b) Removing the first node from the list and keep track of the location of new node.
  c) Copying new information into the new node and setting address of new node at last node.

Polish Notation

- What is polish notation? What are the benefits of polish notation?
- Ans: Polish Notation refers to the notation in which the operator symbol is placed before its two operands. For example: +AB.
- Benefits of Polish Notation: The computer usually evaluates an arithmetic expression written in infix notation in two steps. First it converts the expression to postfix notation, and then it evaluates the postfix expression. Which is Reverse Polish Notation. Hence we need polish notation.

Polish Notation

- Question: Convert the following infix expression to its equivalent prefix and postfix expression.
- (i) A*B+(C*D/E)*F+(G↑H)
- (ii) 1∗ 2/3 + (4 −5 ↑ 6)+7-8
- Ans:
- (i) Prefix: A*B+(C*D/E)*F+(G↑H)

  = A*B+[/*CDE]*F+[↑GH]

  = [*AB]+[*/*CDEF]+[↑GH]

  =++*AB*/*CDEF↑GH

Postfix: A*B+(C*D/E)*F+(G↑H)

= A*B+[CDE*/]*F+[GH↑]

=[AB*]+[CDE*/F*]+[GH↑]

=AB*CDE*/F*+GH↑+

Polish Notation

- (ii) Prefix: $1* 2/3 + (4 -5 \uparrow 6)+7-8$

  $= 1* 2/3 + (4 -[\uparrow 56])+7-8$

  $= 1* 2/3 + [-4 \uparrow 56]+7-8$

  $= [/* 123] + [-4 \uparrow 56]+7-8$

  $= -++/* 123 - 4 \uparrow 5678$

Postfix: $1* 2/3 + (4 -5 \uparrow 6)+7-8$

$= 1* 2/3 + (4 -[56\uparrow])+7-8$

$= 1* 2/3 +[456\uparrow+]+7-8$

$= [12*3/]+[456\uparrow+]+7-8$

$= 12*3/456\uparrow++7+8-$

Polish Notation

- Question: Simulate the postfix expression evaluation algorithm using 12,6,/,6,2,+,*,12,4,/,- by showing Stack's contents as each element is scanned.
- Ans: Consider the following arithmetic expression as P.

  P: 12,6,/,6,2,+,*,12,4,/,-

  First we add a sentinel right parenthesis at the end of P to obtain, P: 12,6,/,6,2,+,*,12,4,/,-)

  The elements of P have been lebeled from left to right for easy reference. Following table shows the contents of Stack as each element of P is scanned. The final number of Stack is the value of P and it will be assigned when ")" is scanned.

Polish Notation

Symbol Scanned	Stack
12	12
6	12,6
/	2
6	2,6
2	2,6,2
+	2,8
*	16
12	16,12
4	16,12,4
/	16,3
-	13

Hence, P=13.

Stack

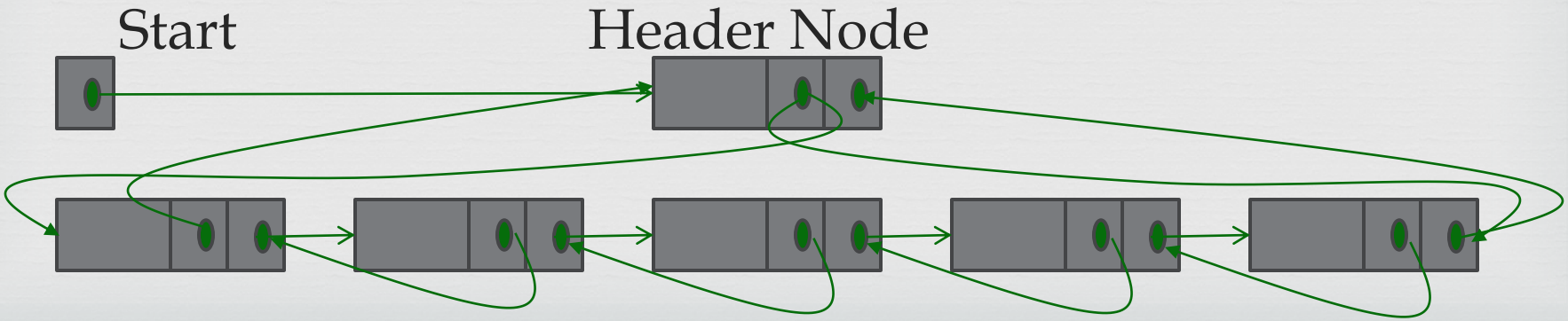
- Question: What is overflow and underflow? How can you handle them?
- Ans: If there is no space in stack for new item then the condition is called overflow.
- If there is no element in stack for delete then the condition is called underflow.
- Underflow depends exclusively upon the given algorithm and the given data, and hence there is no direct control by the programmer. On the other hand, Overflow depends upon the arbitrary choice of the programmer. Generally, the number of elements in a stack fluctuates as elements are added to or removed from stack to handle overflow and underflow.

Two Way Lists

- Question: What is a two way list?
- Ans: A two way list is a linear collection of data elements, called nodes, where each node N is divided into three parts:- information field, Forward link- which points to the next node and Backward link-which points to the previous node.

Two Way Lists

- Question: Why Two Way List is important? Explain with schematic diagram.
- Ans: The importance of a two way list is that, a two way list and a circular header list may be combined into a two way circular header list as the figure below. The list is circular because the two end nodes point back to the header node.



Linked Lists

- Question: What is garbage collection and compaction?
- Ans: The operating system of a computer may periodically collect all the deleted space onto the free storage list. The technique which does this collection is called garbage collection.
- In the second step of garbage of garbage collection, the computer runs through the memory, collecting all untagged spaces onto the free storage list. This is called compaction.

Arrays

- Question: For column major order find out the address of the element marks[12,3] from a 25x4 matrix array marks with base value 250 and w=8.
- Ans: We Know, To find the address of A(J,K) from a column major mxn matrix,

Address = Base + w[m(K-1)+(J-1)]

= 250+8[25(3-1)+(12-1)]

= 250+8[50+11]

=250+488

=738

Pointers

- Question: What is pointer and pointer array?
- Ans: A variable P is called a pointer if P points to an element, i.e., if P contains the address of an element.
- An array PTR is called a pointer array if each element of PTR is a pointer.

Pointers

- Question: How a pointer can save memory space to store a 2D array.
- Ans: A 2D array where every column is not fully filled there we need to use a array as the size which column has most elements. As example if there is 4 column where first column has 6 elements, second column has 3 elements, third column has 16 elements and fourth column has 7 elements, then the total number of elements is 32. But to store these data into a 2D array we need 16x4 array. Where 50% space is wasted. But using pointer we can reduce this wastage. We can store the beginning address of next column in the last of previous column and we can reduce the array size. And in this way a pointer can save memory space to store a 2D array.

Sparse Matrix

Question: What is the difference between triangular matrix and tridiagonal matrix?

Triangular Matrix

1. Where all entries above the main diagonal are zero or, equivalently, where nonzero entries can only occur on or below the main diagonal, is called a triangular matrix.
2. Example: $\begin{bmatrix} 1 & 0 & 0 \\ 2 & 3 & 0 \\ 3 & 4 & 5 \end{bmatrix}$

Tridiagonal Matrix

1. Where nonzero entries can only occur on the diagonal or on elements immediately above or below the diagonal, is called tridiagonal matrix
2. Example: $\begin{bmatrix} 1 & 2 & 0 \\ 2 & 3 & 4 \\ 0 & 4 & 5 \end{bmatrix}$

Trees

- Question: Define heap, leaf and depth of tree. For 5009 nodes, find out the depth of the tree.
- Ans:
- Heap: A tree H is called heap if each node N of H has a value is greater than or equal to the value at each child of N(maxheap) or less than or equal to the value at each child of N(minheap).
- Leaf: A terminal node is called a leaf.
- Depth of Tree: The Depth of Tree T is the maximum number of nodes in a branch of T.
- We Know, Depth of tree $D_n = \log_2 n+1$

  $=\log_2(5009)+1$

  $=14$

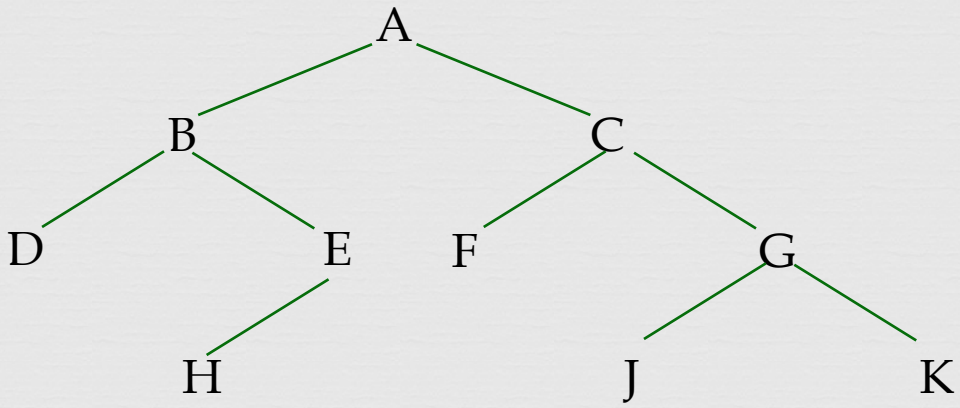
Trees

- Question: Discuss the linked representation of binary tree in memory.
- Ans: Consider a binary tree T which will be represented in memory by means of a linked representation. Which uses three parallel arrays, INFO, LEFT and RIGHT and a pointer variable ROOT as follows. First of all, each node N of T will correspond to a location K such that,
  i. INFO[K] contains the data of node N.
  ii. LEFT[K] contains the location of left child of N.
  iii. RIGHT[K] contains the location of right child of N.

  ROOT will contain location of T. If any subtree is empty then the corresponding pointer will contain null value. If the tree is empty then the ROOT will contain null value.

Trees

- Question: Simulate the preorder traversal algorithm for the following tree.



Trees

- Ans: Consider the tree as T. We simulate the preorder traversal algorithm with T, showing the contents of STACK at each step.

1. Initially push NULL onto STACK. STACK: 0
   Then set PTR:= A.
2. Proceed down the left-most path rooted at PTR=A:
   i. Process A and push C onto STACK. STACK: 0, C
   ii. Process B and push E onto STACK. STACK: 0, C, E
   iii. Process D.
3. Pop the top element E from STACK and set PTR:= E. Now STACK: 0, C
4. Proceed down the left-most path rooted at PTR=E:
   i. Process E.
   ii. Process H.
5. Pop the top element C from STACK and set PTR:= C. Now STACK: 0
6. Proceed down the left-most path rooted at PTR=C:
   i. Process C.
   ii. Process F and push G onto STACK. STACK: 0, G
7. Pop the top element G from STACK and set PTR:= G. Now STACK: 0
8. Proceed down the left-most path rooted at PTR=G:
   i. Process G
   ii. Process J and push K onto STACK. STACK: 0, K
9. Pop the top element K from STACK and set PTR:= K. Now STACK: 0
10. Process K.
11. Pop the top element NULL from STACK and set PTR:= NULL.

Since PTR=NULL, the algorithm is completed.

Graph Terminology

- Question: Define the following graph terms: (i) Adjacent Nodes (ii) Cycle (iii) Connected graph (iv) Weighted graph.
- Ans:

i. Adjacent Nodes: If e=(u,v) where u and v are endpoints of e then u and v are called Adjacent Nodes.

ii. Cycle: A cycle is a closed simple path with length 3 or more.

iii. Connected graph: A graph G is said to be connected if there is a path between any two of its nodes.

iv. Weighted graph: A graph G is said to be weighted if each edge e in G is assigned a nonnegative numerical value.

Sequential Representation

- Question: Discuss the sequential representation of graph with example.
- Ans: There are two standard ways of maintaining a graph G in the memory of computer. One way, called the sequential representation of G, is means of its adjacency matrix A.
- As example: Suppose G is a simple directed graph with 4 nodes then the sequential representation will be like:

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Adjacency Matrix

- Question: Consider the following adjacency matrix:
- $A=\begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$. Now find out A^2, A^3, A^4, B_4 and from that make the path matrix and tell whether this is strongly connected or not.
- Ans:

$$A^2 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

$$A^3 = \begin{pmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{pmatrix}$$

Adjacency Matrix

$$A^4 = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 \end{pmatrix}$$

$$B_4 = \begin{pmatrix} 7 & 7 & 8 & 8 \\ 8 & 8 & 7 & 7 \\ 7 & 7 & 8 & 8 \\ 8 & 8 & 7 & 7 \end{pmatrix}$$

$$P = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Hence, This is strongly connected.

Directed Graph

- Question: What is Directed Graph? Explain.
- A directed graph G is that which has multiple edges and loops and each edge e is identified with an ordered pair (u,v) of nodes in G .
- If G is a directed graph with a directed edge e=(u,v) then,
  i. e begins at u and ends at v.
  ii. u is the origin and v is the destination.
  iii. u is a predecessor of v and v is a successor of u.
  iv. u is adjacent to v and v is adjacent to u.

Traversing a Graph

- Question: How many ways a graph G can be traversed?
- Ans: There are two standard ways a graph G can be traversed systematically.
- Question: What is the significance of STATUS field?
- Ans: STATUS field show the state of N and G during the execution of algorithms.

Depth-First Search

- Question: Consider the adjacency list of the graph G in the following table. Find the nodes that are reachable from node C using Depth-First Search.

Node	**Adjacency**	**Node**	**Adjacency**
A	G, E	E	C
B	C	F	A, B
C	F	G	B, C, E
D	C	H	D

Depth-First Search

- Ans: We want to find all the nodes reachable from the node C. The steps of Depth-First Search are given bellow:

a) Initially, push C onto the stack as follows: STACK: C

b) Pop and print the top element C and then push onto stack all the neighbors of C as follows: Print C STACK: F

c) Pop and print the top element F and then push onto stack all the neighbors of F as follows: Print F STACK: A, B

d) Pop and print the top element B and then push onto stack all the neighbors of B as follows: Print B STACK: A

e) Pop and print the top element A and then push onto stack all the neighbors of A as follows: Print A STACK: G, E

f) Pop and print the top element E and then push onto stack all the neighbors of E as follows: Print E STACK: G

g) Pop and print the top element G and then push onto stack all the neighbors of G as follows: Print G STACK:

Now stack is empty. So depth-first search is complete. The output is C, F, B, A, E, G

Hence, The nodes C, F, B, A, E, G are reachable from C.

Data Structure

- Question: Why Data Structure is needed?
- Ans: A particular data model must be reach enough in structure to mirror the actual relationship of the data in the real world. Moreover data structure save memory space. It also save executation time. So data structure is needed.
- Question: What are the difference between linear and nonlinear data?
- Ans:

Linear Data Structure	Nonlinear Data Structure
1. Data Structures which are linearly connected.	1. Data Structures which are not linearly connected but linked.
2. Example: Array.	2. Example: Graph.
3. Easy to find any data.	3. Hard to find any data.

Data Structure

- Question: Explain linear array representation in memory.
- Ans: By a linear array, we mean a list of a finite number n of similar data elements referenced respectively by a set of n consecutive numbers, usually 1,2,3,……,n. If we choose the name A for the array, then the elements of A are denoted by,

  A[1],A[2],A[3],……,A[n]

Binary Search Tree

- Question: Simulate the binary search algorithm on the following data: 10,20,50,70,80,90,100,110 (suppose we search for item 110).
- Ans:

  Scan all elements in BST[k], where k=0,1,…,7.
  BST={10,20,50,70,80,90,100,110};
  start=0; end=7; mid=3
  Scan the item,
  ITEM=120

1. For each case compare ITEM with A[mid],

   110 > 70, start=4, end=7, mid=5
   110 > 90, start=6, end=7, mid=6
   110 > 100, start=7, end=7, mid=7
   110 = 110,

2. Print item is found at the position 8.

Arrays

- Question: For column major order find out the address of the element score[15,3] from a 20x5 matrix array score with base value 100 and w=4.
- Ans: We Know, To find the address of A(J,K) from a column major mxn matrix,

Address = Base + w[m(K-1)+(J-1)]

= 100+4[20(3-1)+(15-1)]

= 100+4[40+14]

=100+216

=316

Pointers

- Question: How does a pointer array can save memory when stores a variable sized group of data? Discuss with necessary figures.
- Ans: A 2D array where every column is not fully filled there we need to use a array as the size which column has most elements. As example if there is 4 column where first column has 6 elements, second column has 3 elements, third column has 16 elements and fourth column has 7 elements, then the total number of elements is 32. But to store these data into a 2D array we need 16x4 array[See Figure-1]. Where 50% space is wasted. But using pointer we can reduce this wastage. We can store the beginning address of next column in the last of previous column and we can reduce the array size[See Figure-2]. And in this way a pointer array can save memory when stores a variable sized group of data.

Pointers

Column-1	Column-2	Column-3	Column-4
8	8	31	34
23	27	34	24
12	43	5	76
5		3	38
67		6	25
54		7	64
		43	27
		75	
		36	
		46	
		57	
		46	
		35	
		74	
		9	
		30	

Figure-1

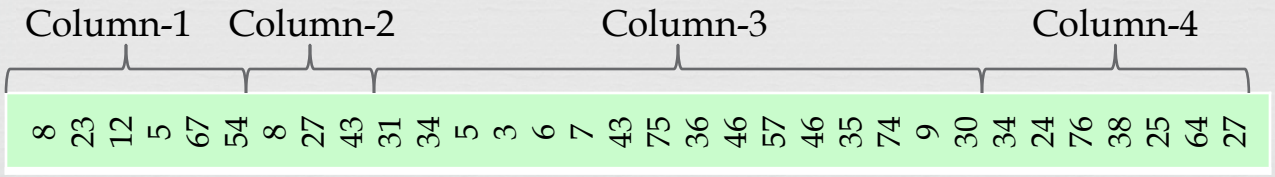


Figure-2

Records

- Question: What is Record?
- Ans: A Record is a collection of related data items.
- Question: What is the difference between a record and a linear array?
- Ans:

Record	Linear Array
1. Record is a collection of related data items.	1. Array is a collection of similar data items.
2. A record can have different types of data.	2. An array only have one data type.
3. Record is indexed by it's field names.	3. Array is indexed by a series of integer numbers.

Linked Lists

- Question: Explain the representation of linked lists in memory.
- Ans: Let LIST be a linked list. First of all LIST requires two linear arrays, let us call them here INFO and LINK, such that INFO[K] and LINK[K] contain respectively, the information part and the nextpointer field of a node of LIST. LIST also requires a variable name- such as START, which contains the location of the beginning of the list, and a nextpointer sentinel denoted by NULL, which indicates the end of the list.

Header Linked Lists

- Question: Discuss header linked list. Describe grounded and circular header list.
- Ans: A header linked list is a linked list which always contains a special node, called the header node, at the beginning of the list. There are two types of widely used header linked lists. They are Grounded Header List and Circular Header List.
- Grounded Header List: A grounded header list is a header list where the last node contains the null pointer.
- Circular Header List: A circular header list is a header list where the last node points back to the header node.

Two Way Lists

- Question: Briefly explain Two Way Linked List.
- Ans: A two way linked list is a linear collection of data elements, called nodes, where each node is divided into three parts:
  i. An information field INFO which contains the data.
  ii. A pointer FORW which contains the location of next node.
  iii. A pointer BACK which contains the location of previous node.
- The list also requires two list pointer variables: FIRST, which points the first node in the list and LAST, which points the last node in the list.

Stacks

- Question: What is stack? What are the operations on stack? Explain with example.
- Ans: A stack is a list of elements in which an element may be inserted or deleted only at one end, called the top of the stack.
- Operations on Stack:
- Push: 'Push' is the term used to insert an element onto a stack. For example: If we push 6 elements in stack then stack is,

  STACK: AAA, BBB, CCC, DDD, EEE, FFF
- Pop: 'Pop' is the term used to delete an element from a stack. For example: If we pop 3 elements from previous stack then stack is,

  STACK: AAA, BBB, CCC

Polish Notation

- Question: Convert the following infix expression to its equivalent prefix and postfix expression.
- (i) A+B*C/D-E+(F/G+H↑ K)
- (ii) (1+2) ↑ 3/4*5 +7-8 ↑9
- Ans:
- (i) Prefix: A+B*C/D-E+(F/G+H↑ K)
  = A+B*C/D-E+(F/G+[↑HK])
  = A+B*C/D-E+([/FG]+[↑HK])
  =A+B*C/D-E+[+/FG↑HK]
  =A+[/*BCD]-E+[+/FG↑HK]
  =+-+A/*BCDE+/FG↑HK

Postfix: A+B*C/D-E+(F/G+H↑ K)
= A+B*C/D-E+(F/G+[HK↑])
= A+B*C/D-E+([FG/]+[HK↑])
= A+B*C/D-E+[FG/HK↑+]
=A+[BC*D/]-E+[FG/HK↑+]
=ABC*D/+E-FG/HK↑++

Polish Notation

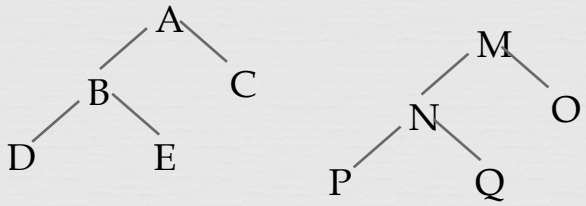
- (ii) Prefix: (1+2) ↑ 3/4*5 +7-8 ↑9
  = [+12]↑ 3/4*5 +7-8 ↑9
  = [↑+123]/4*5 +7-[↑89]
  = [*/↑+12345]+7-[↑89]
  = -+*/↑+123457↑89
- Postfix: (1+2) ↑ 3/4*5 +7-8 ↑9
  = [12+]↑ 3/4*5 +7-8 ↑9
  = [12+3↑]/4*5 +7-[89↑]
  = [12+3↑4/5*]+7-[89↑]
  = 12+3↑4/5*7+89↑ –

Queue

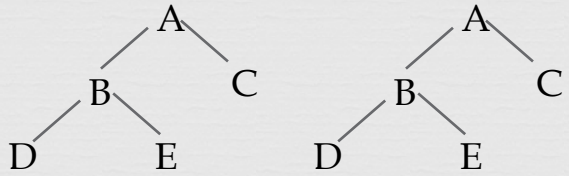
- Question: What is priority queue? Why it is important?
- Ans: A priority queue is a collection of elements such that each element has been assigned a priority and such that the order in which elements are deleted and processed comes from and following rules:
  i. Elements of higher priority is processed first.
  ii. Elements of same priority are processed according to the order in which they were added to the queue.
- Importance of priority queue is it's timesharing system, that is programs of high priority are processed first, and programs with the same priority form a standard queue.

Trees

- Question: Illustrate similar and copies of a tree with examples.
- Ans:
- Similar Trees: Binary trees T and T′ are said to be similar if they have the same shape. For example: T and T′.
- Copies of Tree: The trees are said to be copies if they have same shape and same contents at corresponding nodes. For example: T_1 and T_2.



T Similar T′



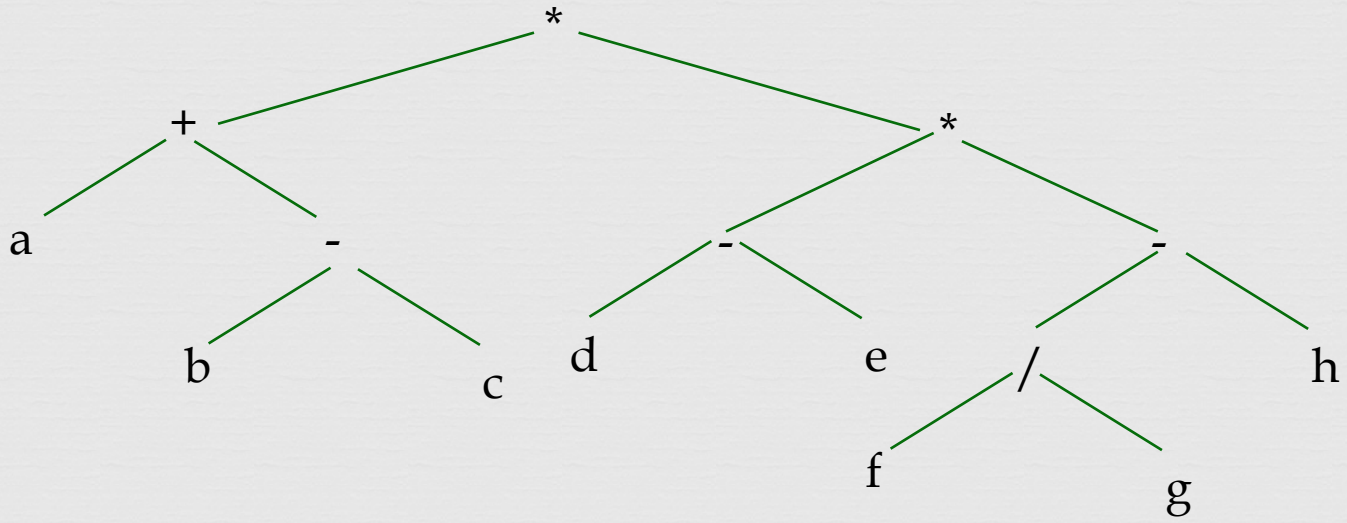
T_1 Copies T_2

Trees

- Question: What is complete binary tree? What is the parent-child relationship?
- Ans: A binary tree T is said to be complete if all its levels, except possibly the last, have the maximum number of possible nodes, and if all the nodes at the last level appear as far left as possible.
- If K is a node then, the left and right child of K are respectively, 2*K and 2*K+1. And the parent of K is the node K/2 as integer.

Tree Traversal

- Question: For the expression: *+a-bc*-de-/fgh draw the tree and perform inorder and postorder traversal.
- Ans: Tree from expression:



Tree Traversal

- Inorder traversal of the expression is:

  a+b-c*d-e*f/g-h

- Postorder traversal of the expression is:

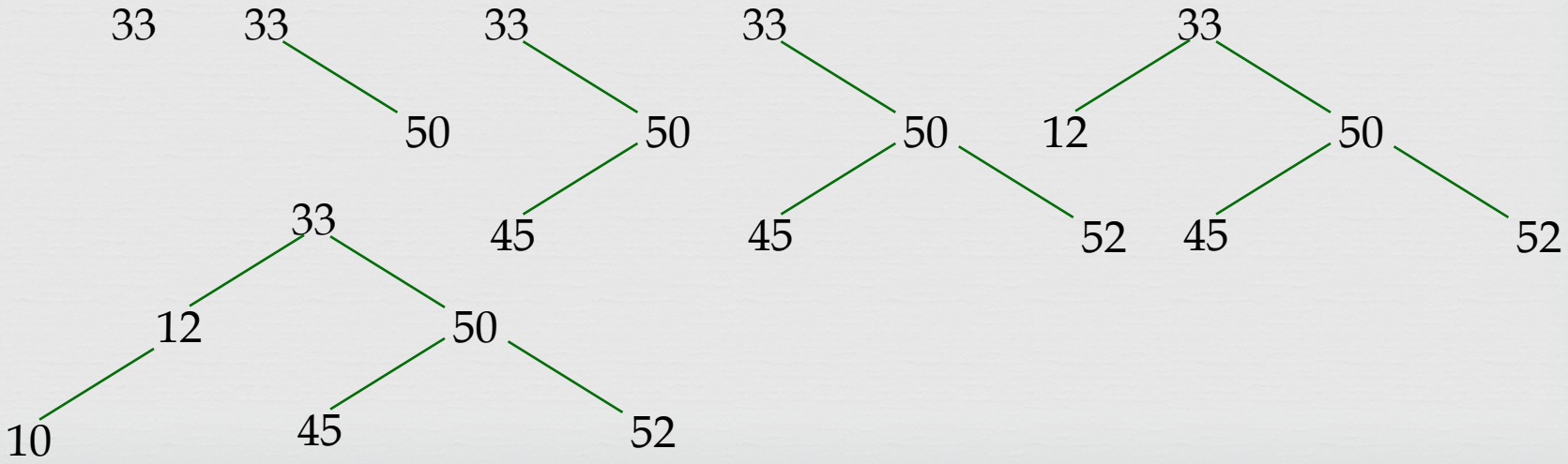
  abc-+de-fg/h-**

Binary Search Tree

- Question: What is Binary Search Tree?
- Ans: A tree T is called Binary Search Tree if each node N of tree T has a value greater than every value in the left subtree of N and less than every value in the right subtree of N.

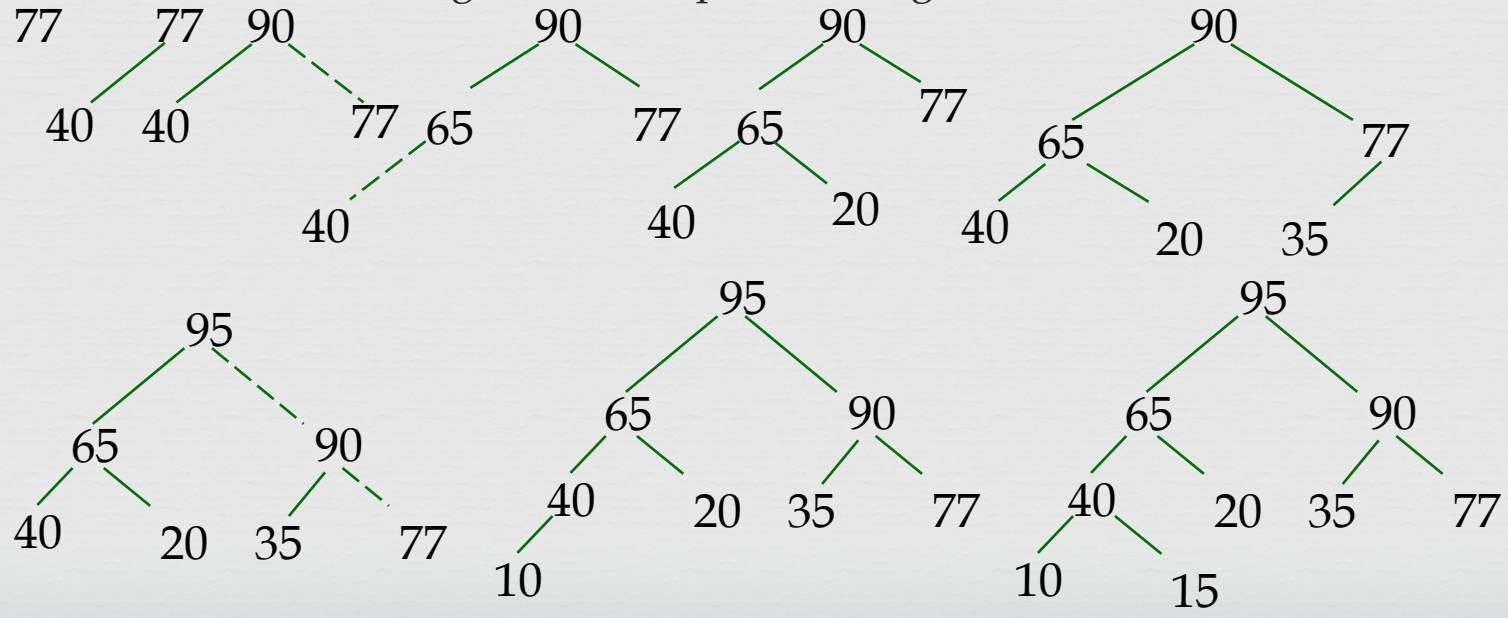
Binary Search Tree

- Question: Suppose the following six numbers are inserted into an empty binary search tree: 33, 50, 45, 52, 12, 10. Show the tree as each number is inserted into a binary search tree.
- Ans:



Binary Search Tree

- Question: Simulate the maxheap algorithm for the following values: 77, 40, 90, 65, 20, 35, 95, 10, 15.
- Ans: This can be accomplished by inserting the eight numbers one after the other into an empty heap H using maxheap algorithm. Pictures below show the heap after inserting each of the nine elements has been inserted. The dotted line indicates that an exchange has taken place during the insertion.



Graph Terminology

- Define the following terms: (i)Degrees of a node (ii)Isolated node (iii)Path (iv)Multi graph.
- Ans:
  i. The degree of a node(u) is the number of edges containing u.
  ii. If u does not belong to any edge then u is called isolated node.
  iii. A path P of length n from a node u to a node v is defined as a sequence of n+1 nodes.
  iv. A graph containing multiple edges and loops is called multi graph.

Adjacency Matrix

- Question: Consider the following adjacency matrix:
- $A=\begin{pmatrix}0 & 1 & 1 & 1\\ 1 & 1 & 0 & 1\\ 1 & 0 & 1 & 1\\ 1 & 0 & 1 & 1\end{pmatrix}$. Now find out A^2,A^3,A^4,B_4 and from that make the path matrix and tell whether this is strongly connected or not.
- Ans:

$$A^2=\begin{pmatrix}3 & 1 & 2 & 3\\ 2 & 2 & 2 & 3\\ 2 & 1 & 3 & 3\\ 2 & 1 & 3 & 3\end{pmatrix}$$

$$A^3=\begin{pmatrix}6 & 4 & 8 & 9\\ 7 & 4 & 7 & 9\\ 7 & 3 & 8 & 9\\ 7 & 3 & 8 & 9\end{pmatrix}$$

Adjacency Matrix

$$A^4 = \begin{pmatrix} 19 & 10 & 23 & 27 \\ 20 & 11 & 23 & 27 \\ 20 & 10 & 24 & 27 \\ 20 & 10 & 24 & 27 \end{pmatrix}$$

$$B_4 = \begin{pmatrix} 28 & 16 & 34 & 40 \\ 30 & 18 & 32 & 40 \\ 30 & 14 & 36 & 40 \\ 30 & 14 & 36 & 40 \end{pmatrix}$$

$$P = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Hence, This is strongly connected.

Linked Representation

- Question: Discuss the linked representation of Graph with example.
- Ans: There are two standard ways of maintaining a graph G in the memory of computer. One way, called the linked representation of G, is means of its adjacency matrix A.
- For Example: Consider a graph G. In the table bellow shows each node in G followed by its adjacency list, which is its adjacency nodes.

Node	**Adjacency**
A	G, E
B	C
C	F

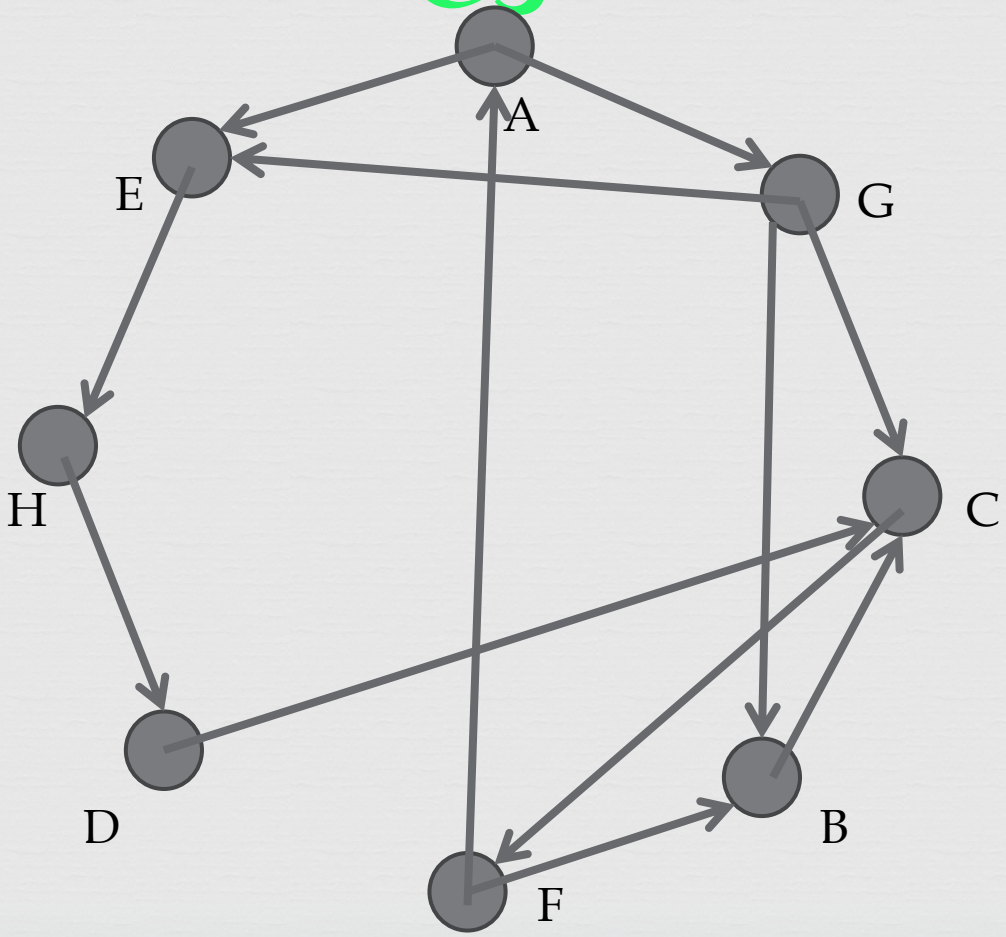
Breadth First Search

- Question: Consider the adjacency list of the graph G in the following table. Draw the graph and find out the path from A to H with minimum number of nodes along that using Breadth First Search nodes that are reachable from node C using Depth-First Search.

Node	**Adjacency**	**Node**	**Adjacency**
A	E, G	E	H
B	C	F	A, B
C	F	G	B, C, E
D	C	H	D

Breadth First Search

Graph:



Breadth First Search

- Ans: We want to find the minimum path P between A and H. The steps of Breadth First Search are given bellow:

a) Initially, add A to QUEUE and add NULL to ORIG as follows:

FRONT=1 QUEUE: A REAR=1 ORIG:0

b) Remove the front element A from QUEUE by setting FRONT:=FRONT+1 and add to QUEUE the neighbors of A as follows:

FRONT=2 QUEUE: A, E, G REAR=3 ORIG: 0, A, A

c) Remove the front element E from QUEUE by setting FRONT:=FRONT+1 and add to QUEUE the neighbors of A as follows:

FRONT=3 QUEUE: A, E, G, H REAR=4 ORIG: 0, A, A, E

We stop as soon as H is added to QUEUE, since H is our final destination. Now we backtrack from H, using the array ORIG to find the path P,

Thus: $A \rightarrow E \rightarrow H$ is the required path.

Data Structure

- Question: Why Data Structure is necessary?
- Ans: A particular data model must be reach enough in structure to mirror the actual relationship of the data in the real world. Moreover data structure save memory space. It also save executation time. So data structure is necessary.
- Question: What do you mean by Linear data structure and Nonlinear data structure? Give example.
- Ans:
- Linear Data Structure: Linear Data Structure means data structures which are linearly connected. For example: Array.
- Nonlinear Data Structure: Nonlinear Data Structure means data structures which are not linearly connected but linked. For example: Graph.

Sparse Matrices

- Question: What is sparse matrix?
- Ans: Sparse Matrix: Matrix with a relatively high proportion of zero entries is called sparse matrix.

Linked Lists

- Question: What are the advantages of linked list?
- Ans: Advantages of linked list:

1. Linked list is dynamic data structure.
2. Link list can grow and shrink during run time.
3. Insertion and deletion operation are easier.
4. Efficient memory utilization.
5. Faster access time.
6. Linear data structures such as stack, queue can be easily implemented.

Linked Lists

- Question: What are the disadvantages of linked list?
- Ans: Disadvantages of linked list:

1. Wastage of memory.
2. No random access.
3. Time Consuming.
4. Heap space restriction.
5. Reverse traversing is difficult.

Data Structure

- Question: Suppose 10 elements are maintained by array and another 10 are by linked list. Which method will take longer time to access 7^{th} element? Justify your answer.
- Answer: The Linked List method will take longer time to access 7^{th} element.
- Because in Array we can directly access the 7^{th} element. But in linked list we cannot directly access 7^{th} element, instead of that we need to go through all elements before 7^{th} element.

Stacks

- Question: Discuss the array representation mechanism of stack.
- Ans: In array representation of stack a array maintains stack. Let's consider the array as STACK. A pointer variable TOP, which contains the location of the top element of the stack. And a variable MAXSTK, which gives the maximum number of elements that can be held by the stack. The condition TOP=NULL will indicate that the stack is empty.

Polish Notation

- Question: Convert the following infix expression to its equivalent prefix and postfix expression.
- (i) A*B/C+D ↑(E-F*G)/H
- (ii) 1+2 * 3/4 ↑5*6-7*8
- Ans:
- (i) Prefix: [/(*AB)C]+D↑(E-[*FG]/H
  =[/*ABC]+D↑[-E*FG]/H
  =[/*ABC]+[↑D-E*FG]/H
  =[/*ABC]+[/↑D-E*FGH]
  =+/*ABC/↑D-E*FGH

Postfix: [AB*C/]+D↑(E-[FG*])/H
=[AB*C/]+D↑[EFG*-]/H
=[AB*C/]+[DEFG*-↑]/H
=AB*C/DEFG*-↑H/+

Polish Notation

- (ii) Prefix: 1+2 * 3/4 ↑5*6-7*8
  =1+2*3/[↑45])*6-7*8
  =1+[/*23*↑ 456]-[*78]
  =+1-/*23*↑456*78

Postfix: 1+2 * 3/4 ↑5*6-7*8
=1+2*3/[45↑])*6-7*8
=1+[23*45↑6*/]-[78*]
=123*45↑6*/+78*-

Recursion

- Question: What is recursion? Explain the use of recursion.
- Ans: Recursion: If P is a procedure containing either a Call statement to itself or a Call statement to a second procedure that may eventually result in a Call statement back to the original procedure P then P is Recursion.
- Uses of recursion:
  1. To find Factorial.
  2. To find Fibonacci Sequence.
  3. To solve Tower of Hanoi's problems.

Queue

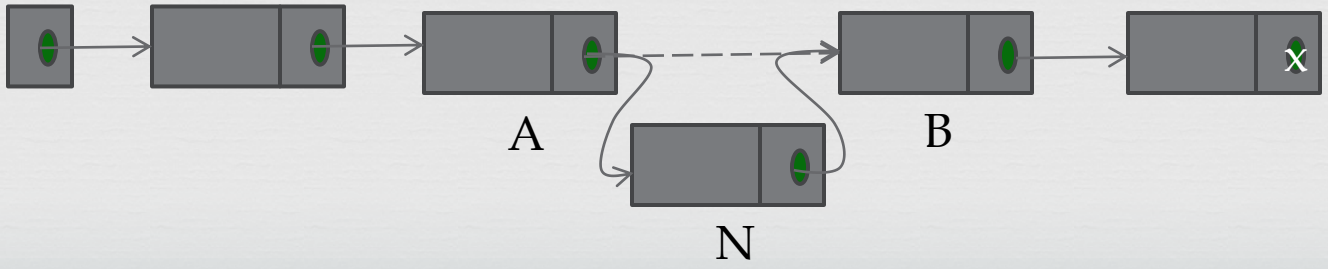
- Question: What is Queue?
- Ans: A queue is a linear list of elements in which deletions can take place only at one end called the front, and insertions can take place only at the other end called rear.
- Question: Explain the operations on queue with example.
- Ans: Two operations occur on queue. They are insertion and deletion.
- Insertion: When insertion occur value of rear increases. For example, if 3 elements are inserted in empty QUEUE where there were FRONT=0 and REAR=0.

  QUEUE: AAA, BBB, CCC ; FRONT=1 ; REAR=3
- Deletion: When deletion occur value of front increases. For example, if 2 elements are deleted from the previous QUEUE where there were FRONT=1 and REAR=3.

  QUEUE: CCC ; FRONT=3 ; REAR=3

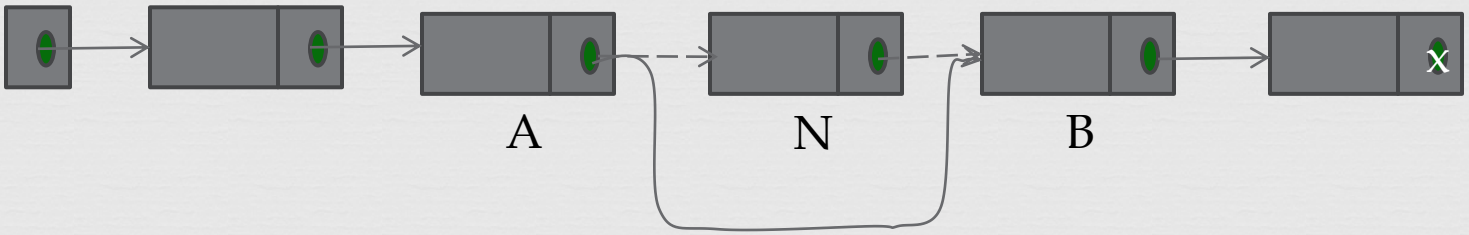
Linked Lists

- Question: Briefly discuss inserting and deleting mechanism of an item in the linked list.
- Ans:
- Inserting: Let LIST be a linked list with successive nodes A and B. Suppose a node N is to be inserted into the list between nodes A and B. After inserting node A now points to the new node N, and node N points to node B, to which A previously pointed.



Linked Lists

- Deleting: Let LIST be a linked list with node N between nodes A and B. Suppose the node N is to be deleted from the list between nodes A and B. After deleting node A now points to the node B, to which N previously pointed.



Queue

- Question: Define deque and priority queue with example.
- Ans: Deque: A deque is a linear list in which elements can be added or removed at any end but not in the middle. For example: DEQUE is a deque.

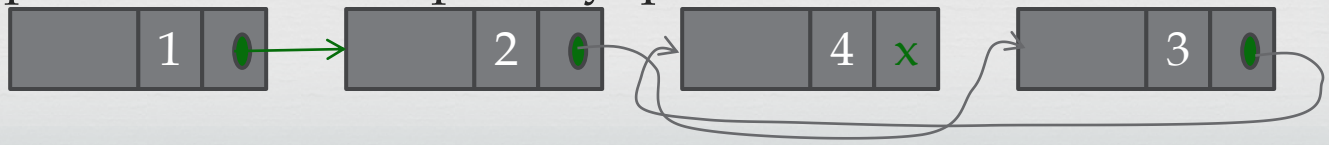
DEQUE:

			AAA	BBB	CCC	DDD	

- A priority queue is a collection of elements such that each element has been assigned a priority and such that the order in which elements are deleted and processed comes from and following rules:
  i. Elements of higher priority is processed first.
  ii. Elements of same priority are processed according to the order in which they were added to the queue.

For example: PQUEUE is a priority queue.

PQUEUE:

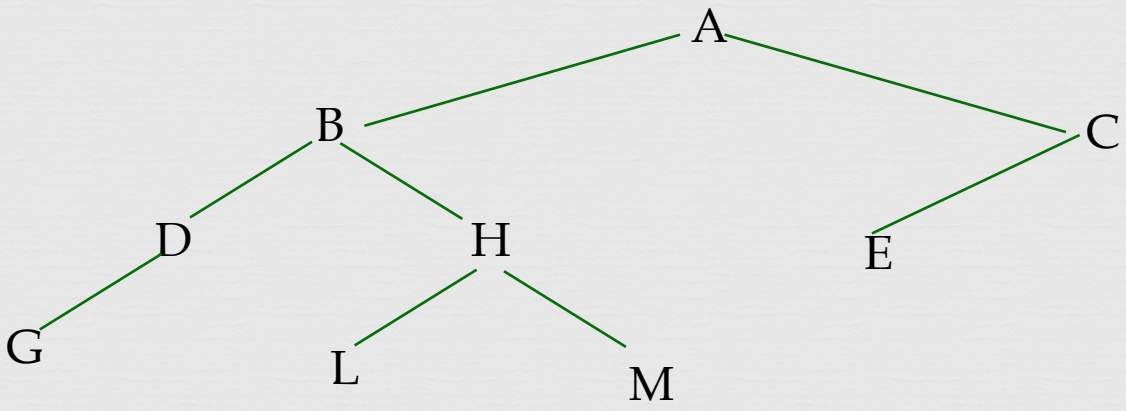


Linked Lists

- Question: What is garbage collection? When does it take place?
- Ans: The operating system of a computer may periodically collect all the deleted space onto the free storage list. The technique to does this collection is called garbage collection.
- The garbage collection take place when there is only some minimum amount of space or no space at all left in the free storage list or when CPU is idle and has time to do the collection.

Trees

- Question: Simulate the preorder traversal algorithm for the following tree.



Trees

- Ans: Consider the tree as T. We simulate the preorder traversal algorithm with T, showing the contents of STACK at each step.

1. Initially push NULL onto STACK. STACK: 0
   Then set PTR:= A.
2. Proceed down the left-most path rooted at PTR=A:
   i. Process A and push C onto STACK. STACK: 0, C
   ii. Process B and push H onto STACK. STACK: 0, C, H
   iii. Process D.
   iv. Process G.
3. Pop the top element H from STACK and set PTR:= H. Now STACK: 0, C
4. Proceed down the left-most path rooted at PTR=H:
   i. Process H and push M onto STACK. STACK: 0, C, M
   ii. Process L.
5. Pop the top element M from STACK and set PTR:= M. Now STACK: 0, C
6. Proceed down the left-most path rooted at PTR=M:
   i. Process M.
7. Pop the top element C from STACK and set PTR:= C. Now STACK: 0
8. Proceed down the left-most path rooted at PTR=C:
   i. Process C.
   ii. Process E.
9. Pop the top element NULL from STACK and set PTR:= NULL.

Since PTR=NULL, the algorithm is completed.

Binary Search Tree

- Question: What is Binary Search Tree?
- Ans: A tree T is called Binary Search Tree if each node N of tree T has a value greater than every value in the left subtree of N and less than every value in the right subtree of N.
- Question: Why binary search tree is important?
- Ans: Binary Search Tree enables one to search for and find an element with an average running time $f(n)=O(log_2 n)$. It also enables one to easily insert and delete elements. So binary search tree is important.

Heap

Question: What is the difference between maxheap and minheap?

Maxheap

1. A tree is called maxheap if each node of the tree has a value greater than or equal to the value at each of children of the node.
2. In maxheap root is the largest value.

Minheap

1. A tree is called minheap if each node of the tree has a value less than or equal to the value at each of children of the node.
2. In minheap root is the smallest value.

Graph Terminology

- Question: Define weighted graph and directed graph with example.
- Ans:
- Weighted graph: A graph G is said to be weighted if each edge e in G is assigned a nonnegative numerical value. Fig X is a weighted graph.
- Directed graph: A directed graph G is that which has multiple edges and loops and each edge e is identified with an ordered pair (u,v) of nodes in G . Fig Y is a directed graph.

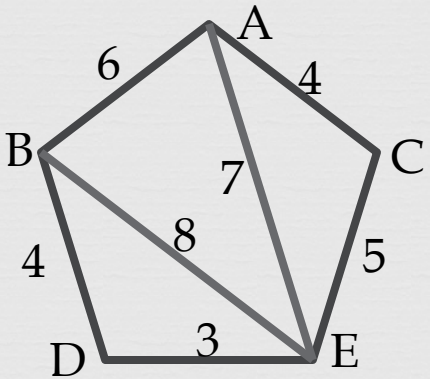


Fig: X

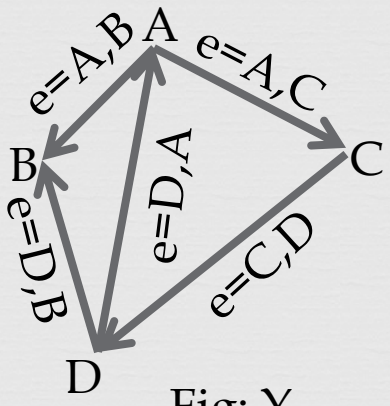


Fig: Y

Adjacency Matrix

- Question: Consider the following adjacency matrix:
- $A=\begin{pmatrix}1 & 0 & 1 & 1\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 1\\ 1 & 1 & 0 & 0\end{pmatrix}$. Now find out A^2,A^3,A^4,B_4 and from that make the path matrix and tell whether this is strongly connected or not.
- Ans:

$$A^2 = \begin{pmatrix}2 & 1 & 2 & 2\\ 0 & 1 & 0 & 0\\ 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1\end{pmatrix}$$

$$A^3 = \begin{pmatrix}4 & 3 & 4 & 4\\ 0 & 1 & 0 & 0\\ 2 & 2 & 2 & 2\\ 2 & 2 & 2 & 2\end{pmatrix}$$

Adjacency Matrix

$$A^4 = \begin{pmatrix} 8 & 7 & 8 & 8 \\ 0 & 1 & 0 & 0 \\ 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 \end{pmatrix}$$

$$B_4 = \begin{pmatrix} 15 & 11 & 15 & 15 \\ 0 & 4 & 0 & 0 \\ 7 & 7 & 8 & 8 \\ 8 & 8 & 7 & 7 \end{pmatrix}$$

$$P = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Hence, This is not strongly connected.

Warshall's Algorithm

- Question: Use the Warshall's Algorithm to find the shortest path matrix of the weighted matrix given below:

$$W = \begin{pmatrix} 6 & 8 & 0 & 0 \\ 3 & 0 & 0 & 9 \\ 5 & 8 & 3 & 6 \\ 6 & 2 & 3 & 0 \end{pmatrix}$$

Ans: Applying the Warshall's Algorithm, we obtain the following matrices Q_0, Q_1, Q_2, Q_3, $Q_4 = Q$.

$$Q_0 = \begin{pmatrix} 6 & 8 & \infty & \infty \\ 3 & \infty & \infty & 9 \\ 5 & 8 & 3 & 6 \\ 6 & 2 & 3 & \infty \end{pmatrix}$$

$$Q_1 = \begin{pmatrix} 6 & 8 & \infty & \infty \\ 3 & 11 & \infty & 9 \\ 5 & 8 & 3 & 6 \\ 6 & 2 & 3 & \infty \end{pmatrix}$$

$$Q_2 = \begin{pmatrix} 6 & 8 & \infty & 17 \\ 3 & 11 & \infty & 9 \\ 5 & 8 & 3 & 6 \\ 6 & 2 & 3 & 11 \end{pmatrix}$$

$$Q_3 = \begin{pmatrix} 6 & 8 & \infty & 17 \\ 3 & 11 & \infty & 9 \\ 5 & 8 & 3 & 6 \\ 6 & 2 & 3 & 11 \end{pmatrix}$$

$$Q_4 = \begin{pmatrix} 6 & 8 & 20 & 17 \\ 3 & 11 & 11 & 9 \\ 5 & 8 & 3 & 6 \\ 6 & 2 & 3 & 11 \end{pmatrix};$$

Hence Q_4 is the shortest path matrix.

Arrays

- Question: What is linear array? How can you find the no of elements in any linear array?
- Ans: A linear array is a list of a finite number n of homogeneous data elements which are referenced by an index and stored in successive memory locations respectively.
- I can find the no of elements in any linear array using the following formula,

  Length = UB – LB + 1

Binary Search Tree

- Question: Simulate the binary search algorithm on the following data: 11,22,33,44,55,66,77,88,99,110,121,132,143 (suppose we search for item 120).
- Ans:

  Scan all elements in BST[k], where k=0,1,…,12.
  BST={11,22,33,44,55,66,77,88,99,110,121,132,143};
  start=0; end=12; mid=6
  Scan the item,
  ITEM=120

1. For each case compare ITEM with A[mid],
   120 > 77, start=7, end=12, mid=9
   120 > 110, start=10, end=12, mid=11
   120 < 132, start=10, end=10, mid=10
   120 ≠ 121,
2. Print item is not found.

Arrays

- Question: What is 2D array? How can you represent 2D array in memory?
- Ans: A 2D mxn array is a collection of m.n data elements such that each element is specified by a pair of integers (such as J,K) with the properties that,

  $1 \leq J \leq m \qquad and \qquad 1 \leq K \leq n$

- Let A be a 2D mxn array. Programming language will store the array A either (1) column major order and (2) row major order. Although A is pictured as a rectangular array of elements with m rows and n columns, the array will be represented in memory by a block of m.n sequential memory locations.

Arrays

- Question: Room(1:8, -4:1, 6:10) is a 3D array with base=400, w=2, calculate Room[4, -2, 7] address in a row major order and column major order.
- Ans: From the array Room(1:8, -4:1, 6:10) we get,

  L_1=8-1=7, L_2=1+4=5, L_3=10-6=4
- And E_1=4-1=3, E_2=-2+4=2, E_3=7-1=6
- Now, location of Room in column major order,

  Room[4,-2,7] =Base+w[(L_2E_3 +L_1)E_2+E_1]
  =400+2[(5.6+7)2+3]
  =554
- location of Room in row major order,

  Room[4,-2,7] =Base+w[(L_2E_1 +L_1)E_2+E_3]
  =400+2[(5.3+7)2+6]
  =500

N.B. Room(1:8, -4:1, 6:10) is a array of FORTRAN language.

Sparse Matrix

- Question: How can you locate element a_{ij} of a sparse matrix from a 1D array?
- Ans: To locate a_{ij} in array B we need the formula that gives us the integer L in terms of i and j where,

$$B[L]=a_{ij}$$

- Observe that L represents the number of elements in the list up to and including a_{ij}. Now they are,

$1+2+3+\ldots\ldots+(i-1)=\frac{i(i-1)}{2}$ elements in the row above a_{ij}, and there are j elements in the row j up to and including a_{ij}, Accordingly,

$$L=\frac{i(i-1)}{2}+j$$

- And L is the location of a_{ij} in 1D array.

Sparse Matrix

- Question: What is memory saving if we store a sparse matrix in a 1D array rather than a 2D array?
- Ans: Most of the entries of a sparse matrix is zero. When we store a sparse matrix into a 2D array there need a array of the same size of the matrix either we save zero entries or not. But if we store a sparse matrix in a 1D array we can ignore zero entries and can save only nonzero entries. Hence almost half space is needed for a sparse matrix to store in 1D array rather than store in 2D array.

Linked Lists

- Question: Briefly discuss inserting mechanism of an item to a sorted list.
- Ans: Suppose ITEM is to be inserted into a sorted linked LIST. Then ITEM must be inserted between nodes A and B so that, $INFO(A) < ITEM \leq INFO(B)$.
- First we have to find the LOC of node A. Using a pointer variable PTR and comparing ITEM with INFO[PTR] at each node. While traversing, keep track of the location of the preceding node by using a pointer variable SAVE. The traversing continues as long as INFO[PTR]>ITEM. Then PTR points to node B, so SAVE will contain the location of node A.

Stacks and Queues

Question: Differentiate between stack and queue.

Stacks

1. A stack is a list of elements in which an element may be inserted or deleted only at the end, called top of the stack.
2. Stack allows Last In First Out process.
3. In stack insertion and deletion occur at same end.

Queues

1. A queue is a list of elements in which an element may be inserted at one end called rear and deleted from other end called front.
2. Queue allows First In First Out process.
3. In queue insertion and deletion occur at different end.

Polish Notation

- Question: Convert the following infix expression to its equivalent prefix and postfix expression.
- (i) A+(B-C)*D/E+F/G↑H
- (ii) $(1+2) * 3 - 4/5 * 6 \uparrow 7$
- Ans:
- (i) Prefix: A+(B-C)*D/E+F/G↑H
  =A+[-BC]*D/E+F/G↑H
  =A+[-BC]*D/E+F/[↑GH]
  =A+[/*-BCDE]+[/F↑GH]
  =++A/*-BCDE/F↑GH
- Postfix: A+(B-C)*D/E+F/G↑H
  =A+[BC-]*D/E+F/G↑H
  =A+[BC-]*D/E+F/[GH↑]
  =A+[BC-D*E/]+[FGH↑/]
  =ABC-D*E/+FGH↑/+

Polish Notation

- (ii) Prefix: $(1+2) * 3 - 4/5 * 6 \uparrow 7$
  $=[+12]* 3 - 4/5 * 6 \uparrow 7$
  $=[+12]* 3 - 4/5 * [\uparrow 67]$
  $=[*+123]-[*/45\uparrow 67]$
  $=-*+123*/45\uparrow 67$
- Postfix: $(1+2) * 3 - 4/5 * 6 \uparrow 7$
  $=[12+]* 3 - 4/5 * 6 \uparrow 7$
  $=[12+]* 3 - 4/5 * [67 \uparrow]$
  $=[12+3*]-[45/67 \uparrow *]$
  $=12+3*45/67 \uparrow *-$

Polish Notation

- Question: Simulate the postfix expression evaluation algorithm using 20,5,2,*,/,2,3,↑,↑,4,/,- by showing Stack's contents as each element is scanned.
- Ans: Consider the following arithmetic expression as P.

  P: 20,5,2,*,/,2,3,↑,↑,4,/,-

  First we add a sentinel right parenthesis at the end of P to obtain, P: 20,5,2,*,/,2,3,↑,↑,4,/,-)

  The elements of P have been lebeled from left to right for easy reference. Following table shows the contents of Stack as each element of P is scanned. The final number of Stack is the value of P and it will be assigned when ")" is scanned.

Polish Notation

Symbol Scanned	Stack
20	20
5	20,5
2	20,5,2
*	20,10
/	2
2	2,2
3	2,2,3
↑	2,8
↑	256
4	256,4
/	64
-	64 [expression error]

Hence, P=64.

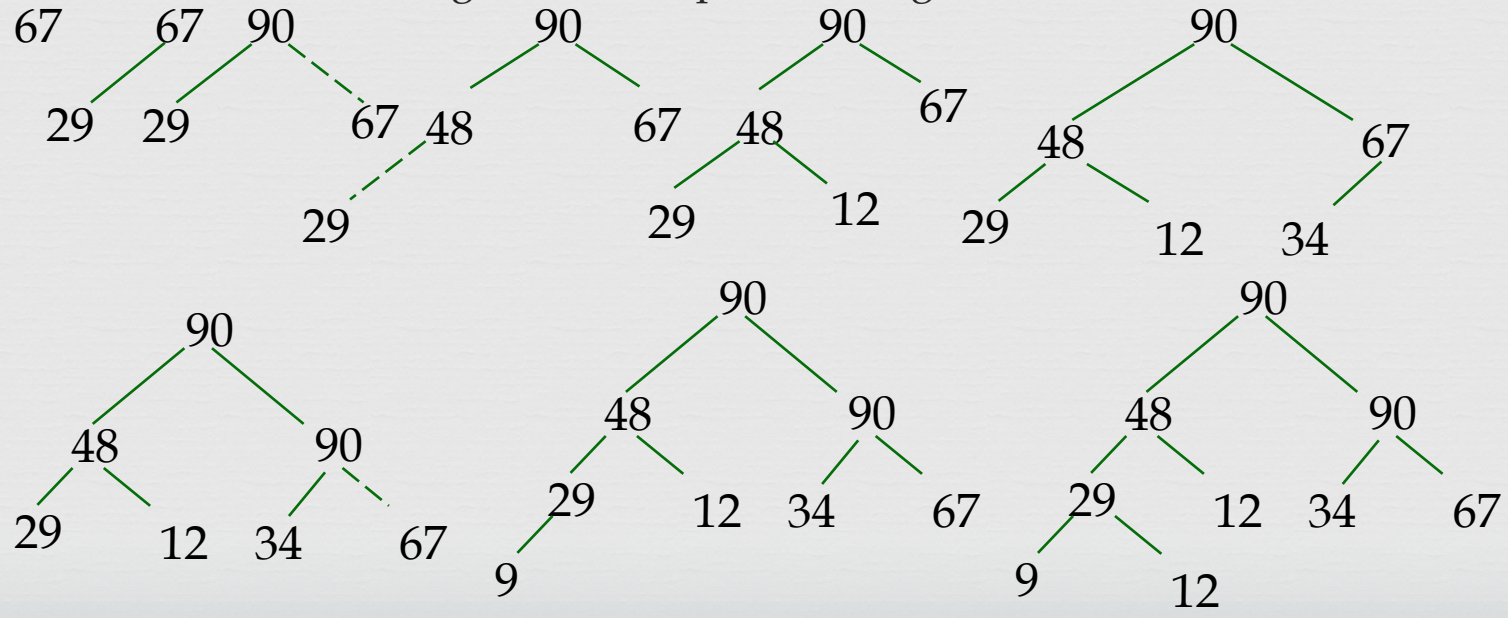
Trees

- Question: A complete binary tree has 1129 nodes, find out the depth of the tree.
- Ans: We Know, Depth of tree,

$$D_n = \log_2 n+1$$
$$=\log_2(1129)+1$$
$$=12$$

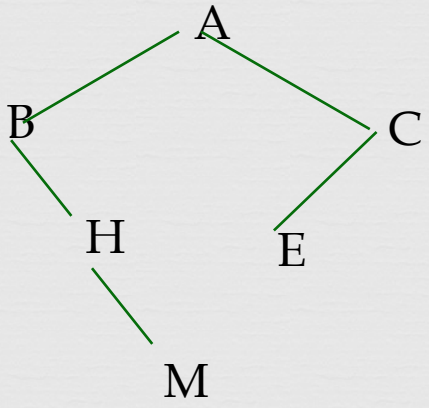
Binary Search Tree

- Question: Simulate the maxheap algorithm for the following values: 67, 29, 90, 48, 12, 34, 90, 9, 12.
- Ans: This can be accomplished by inserting the eight numbers one after the other into an empty heap H using maxheap algorithm. Pictures below show the heap after inserting each of the nine elements has been inserted. The dotted line indicates that an exchange has taken place during the insertion.

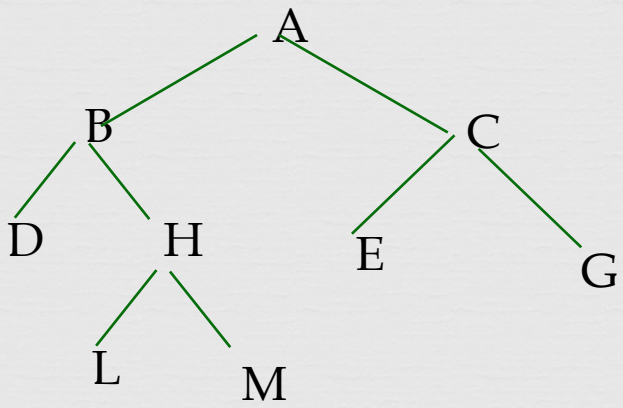


Trees

- Question: What is extended tree?
- Ans: A tree T is said to be an extended tree if each node N has either 0 or 2 children.



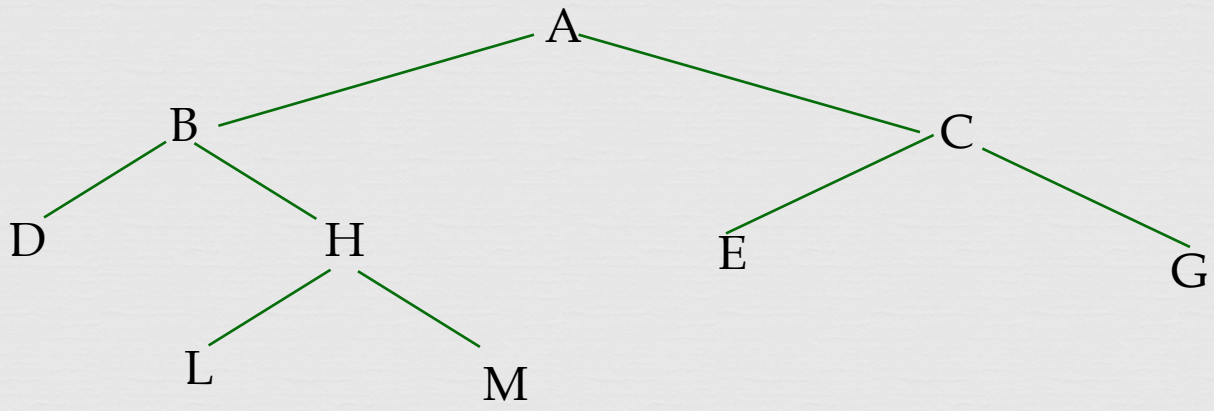
Normal Tree



Extended Tree

Trees

- Question: Simulate the postorder traversal algorithm for the following tree.



Trees

- Ans: Consider the tree as T. We simulate the postorder traversal algorithm with T, showing the contents of STACK at each step.

1.	Initially push NULL onto STACK. Then set PTR:= A.	STACK: 0
2.	Proceed down the left-most path rooted at PTR=A, pushing A, B, D onto stack. Push –C onto stack after A and before B and push –H after B and before D.	STACK: 0, A, -C, B, -H, D
3.	(i) Pop and process D.	STACK: 0, A, -C, B, -H
	(ii) Pop -H and set PTR:=H.	STACK: 0, A, -C, B
4.	Proceed down the left-most path rooted at PTR=H, pushing H,L onto stack. Push –M onto stack after H and before L.	STACK: 0, A, -C, B, H, -M, L
5.	(i) Pop and process L.	STACK: 0, A, -C, B, H, -M
	(ii) Pop –M and set PTR:=M.	STACK: 0, A, -C, B, H
6.	Proceed down the left-most path rooted at PTR=M, pushing M onto stack.	STACK: 0, A, -C, B, H, M
7.	(i) Pop and process M.	STACK: 0,A, -C, B, H
	(ii) Pop and process H.	STACK: 0,A, -C, B
	(iii) Pop and process B.	STACK: 0,A, -C
	(iv) Pop –C and set PTR:=C.	STACK: 0,A
8.	Proceed down the left-most path rooted at PTR=C, pushing C, E onto stack. Push –G onto stack after C and before E.	STACK: 0, A, C, -G, E
9.	(i) Pop and process E.	STACK: 0, A, C, -G
	(ii) Pop –G and set PTR:=G.	STACK: 0, A,.C
10.	Proceed down the left-most path rooted at PTR=G, pushing G onto stack.	STACK: 0, A, C, G
11.	(i) Pop and process G.	STACK: 0,A, C
	(ii) Pop and process C.	STACK: 0,A
	(iii) Pop and process A.	STACK: 0
	(iv) Pop 0 and set PTR:=0.	

Since PTR=NULL, the algorithm is completed.

Graph Terminology

- Question: Define the following terms: Connected graph, Path, Weighted graph.
- Ans:
- Connected Graph: graph G is said to be connected if there is a path between any to of its nodes.
- Path: A path P of length n from a node u to a node v is defined as a sequence of n+1 nodes.
- Weighted graph: A graph G is said to be weighted if each edge e in G is assigned a nonnegative numerical value. Fig X is a weighted graph.

Trees

- Question: Write the steps of preorder and postorder traversal of a binary tree.
- Ans:
- Preorder:

a) Proceed down the left most path rooted at PTR, processing each node N on the path and pushing each right child R(N), if any, onto STACK. The traversing ends after a node N with no left child L(N) is processed.

b) Pop and assign to PTR the top element on STACK. If PTR≠NULL, return to step (a); otherwise Exit.

- Postorder:

a) Proceed down the left most path rooted at PTR. At each node N of the path, push N onto STACK and, if N has a right child R(N), push –R(N) onto STACK.

b) Pop and process positive nodes on STACK. If NULL is popped, then Exit. If a negative node is popped, that is if PTR=-N for some node N, set PTR=N and to step (a).

Binary Search Tree

- Question: Mention the advantages of a binary search tree.
- Ans: Advantages of Binary Search Tree:

1. Easy to access any elements.
2. Save time to search any element.
3. Easy to insert and delete any element.

Thank You

Manik Hosen