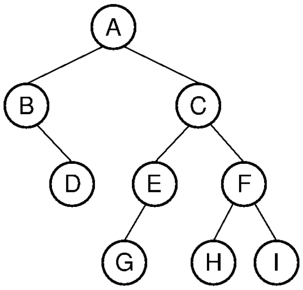
Topic: Tree ADT
Due: 9th Nov 2020

I hereby pledge that I have not copied from anyone of my classmates/ seniors or any other sources. (Put tick mark if you are taking the pledge) ✓

Roll Number : CED19I027

Due: 9th Nov 2020; '-4' marks for late submission

1. **For the below binary tree write**



a) **Preorder:** A B D C E G F H I

b) **Inorder:** B D A G E C H F I

c) **Postorder:** D B G E H I F C A

d) **Levelorder:** A B C D E F G H I

2. **Construct an expression tree for the following expression**

a) **A-(C/5*2)+(D*5%4)**

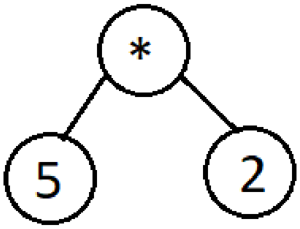
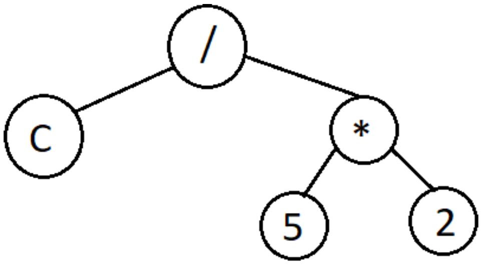
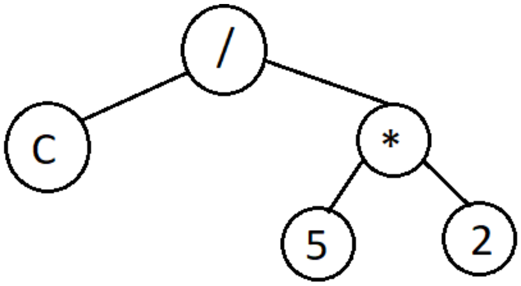
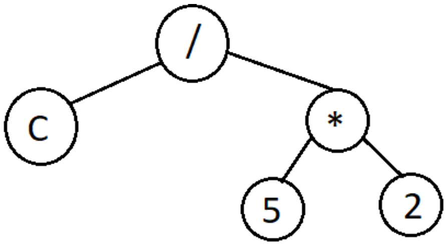
Let us first find the postfix expression for this infix expression

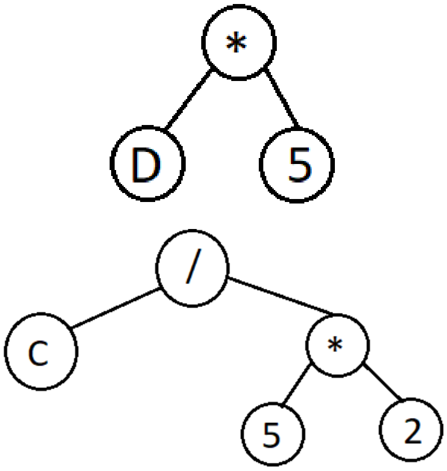
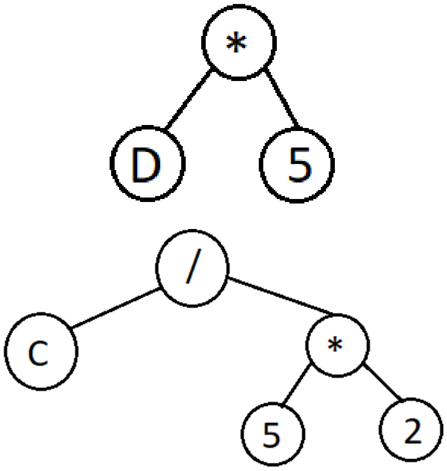
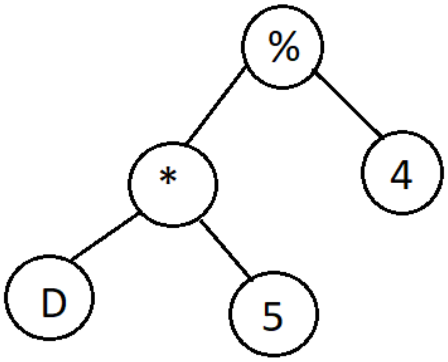
Infix	stack	Postfix
A		A
-	-	A
(	-(	A
C	-(	A C

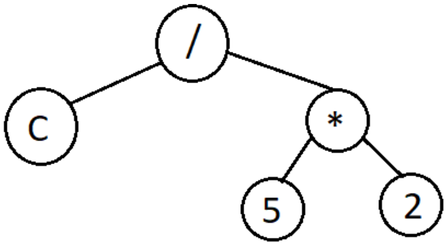
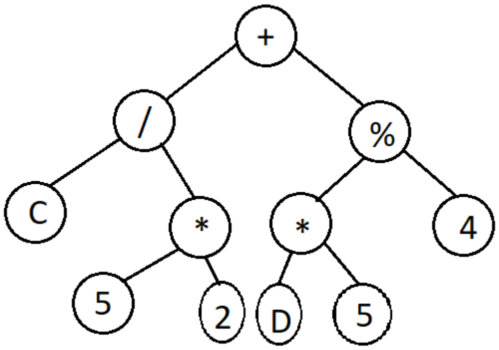
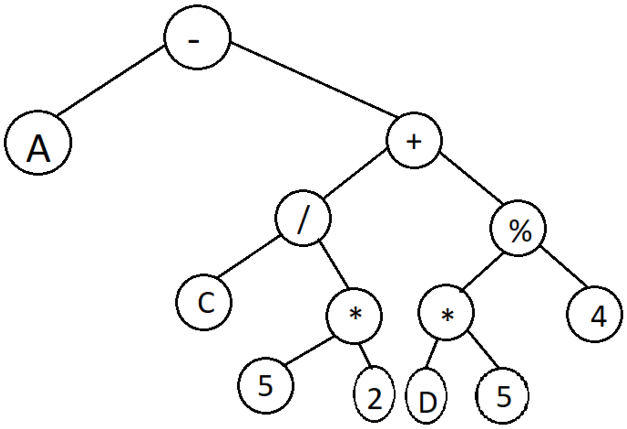
/	-(/	A C
5	-(/	A C 5
*	-(/*	A C 5
2	-(/*	A C 5 2
)	-	A C 5 2 * /
+	- +	A C 5 2 * /
(	- + (	A C 5 2 * /
D	- + (	A C 5 2 * / D
*	-+(*	A C 5 2 */ D
5	-+(*	A C 5 2 * / D 5
%	-+(%	A C 5 2 * / D 5 *
4	-+(%	A C 5 2 * / D 5 * 4
)	-+	A C 5 2 * / D 5 *4 %
		A C 5 2 * / D 5 * 4 % + -

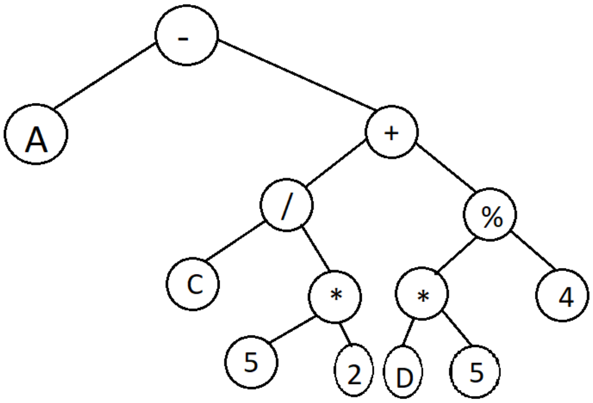
POSTFIX	STACK
A	A
C	C A
5	5 C A
2	2 5 C A

Topic: Tree ADT
Due: 9th Nov 2020

*	 C A
/	 A
D	D  A
5	5 D  A

*	 A
4	4  A
%	

	 A
+	 A
-	



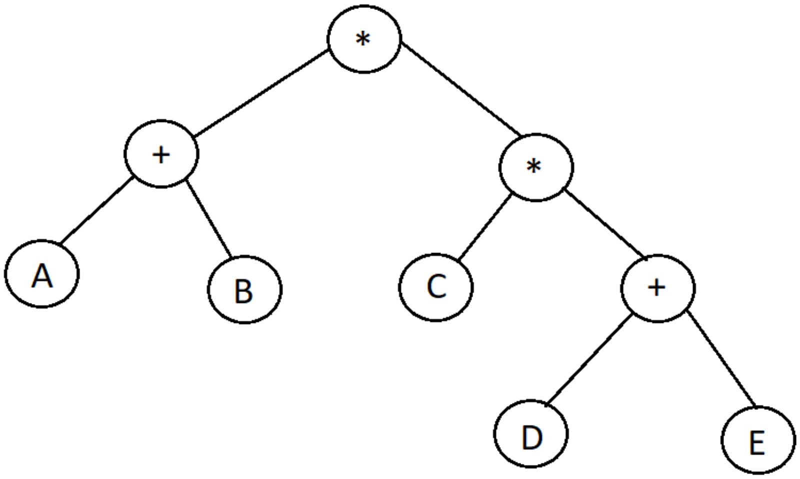
The final binary tree will look like this

3. **For the above constructed expression tree write the following**

    a) **Preorder:** - A + / C * 5 2 % * D 5 4
    b) **Inorder:** A - C / 5 * 2 + D * 5 % 4
    c) **Postorder:** A C 5 2 * / D 5 * 4 % + -
    d) **Levelorder:** - A + / % C * * 4 5 2 D 5

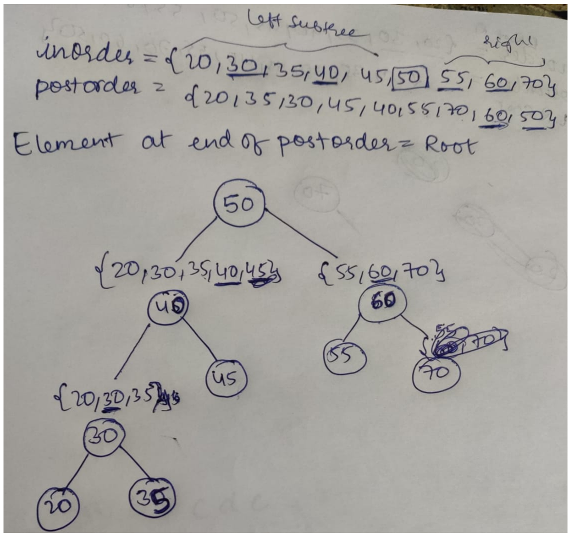
4. **Convert the following postfix expression to expression tree**

    a) **a b + c d e + * ***

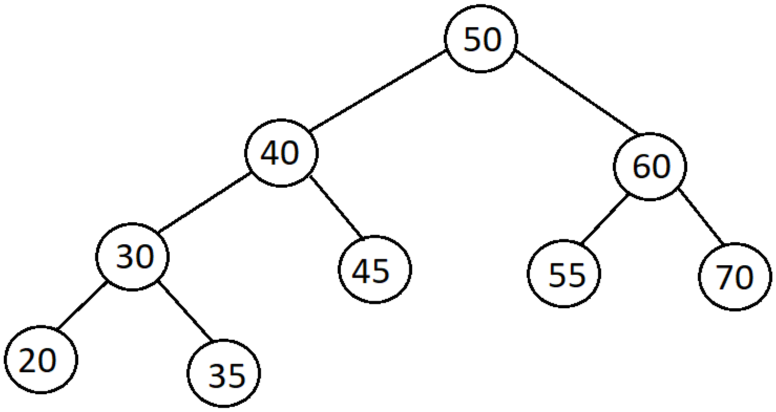


5. **Construct binary tree using the order given below**

inOrder = {20, 30, 35, 40, 45, 50, 55, 60, 70};
postOrder = {20, 35, 30, 45, 40, 55, 70, 60, 50};


Left subtree
right
inorder = {20, 30, 35, 40, 45, 50 55, 60, 70}
postorder = {20, 35, 30, 45, 40, 55, 70, 60, 50}
Element at end of postorder = Root

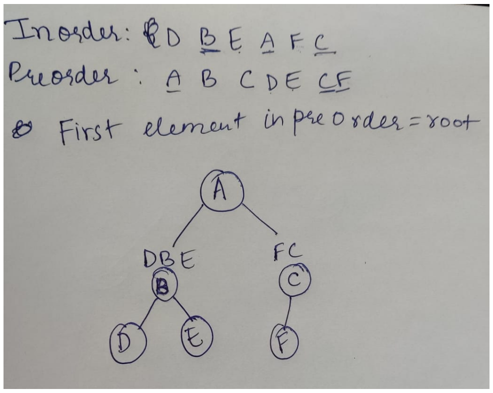
final Binary tree Structure will look like this:



It is also an ordered Binary Search tree.

6. Construct binary tree using the order given below

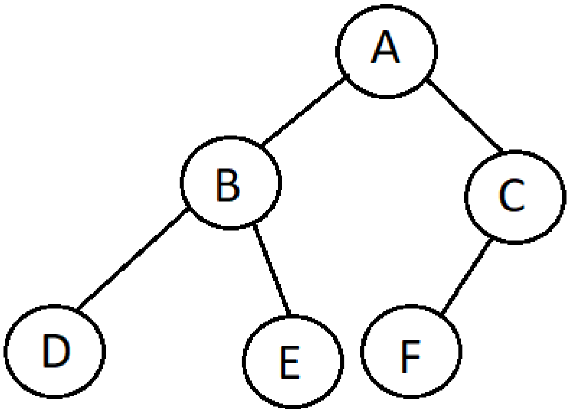
Inorder sequence: D B E A F C
Preorder sequence: A B D E C F



Inorder: ~~B~~D B E A F C
Preorder : A B C DE CF
- First element in pre Order = root

A
DBE
B
FC
C
D
E
F

Final binary tree will look as follows:

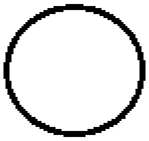


Topic: Tree ADT
Due: 9th Nov 2020

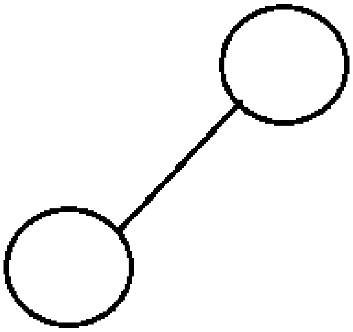
7. **How many binary trees are possible with 'n' nodes in general? Draw them for the following 'n' values.**
a) n=1 b) n=2 c) n=3 d) n=4

ans)

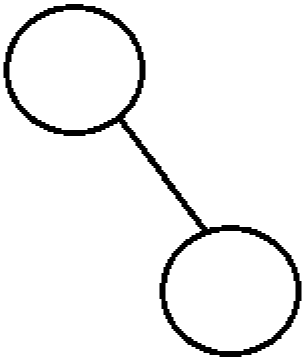
(a) for n=1, only one binary tree is possible



(b) for n=2 , two binary trees are possible

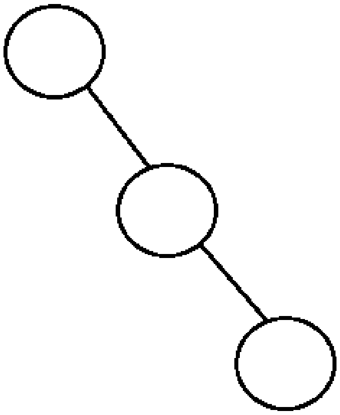


1

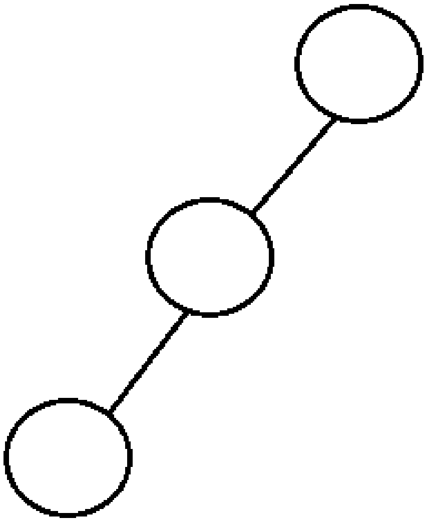


2

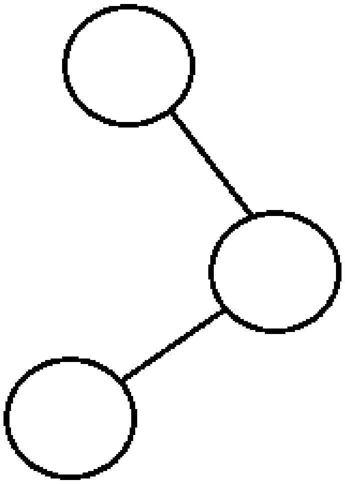
(c) for n=3 , five binary trees are possible



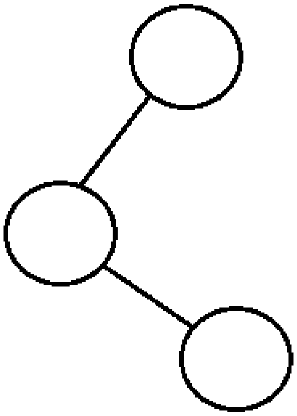
1



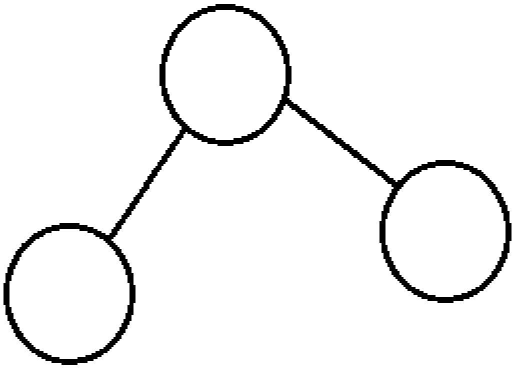
2



3

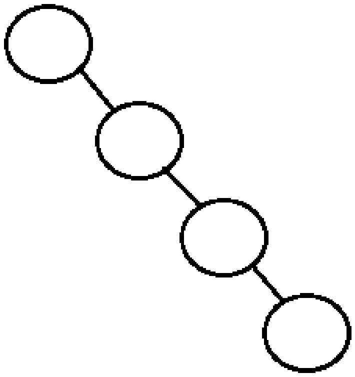


4

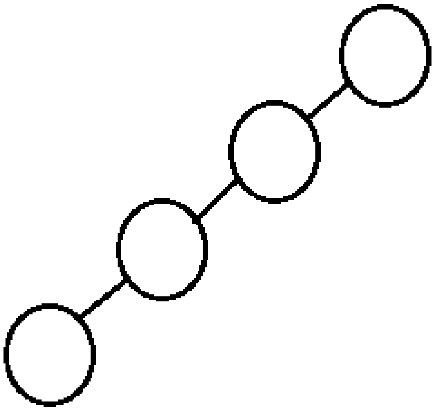


5

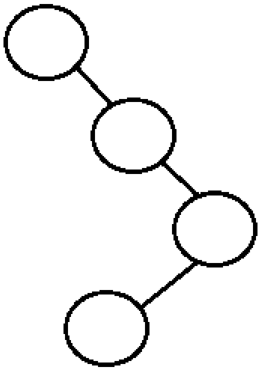
(d) for n= 4 , 14 binary trees are possible



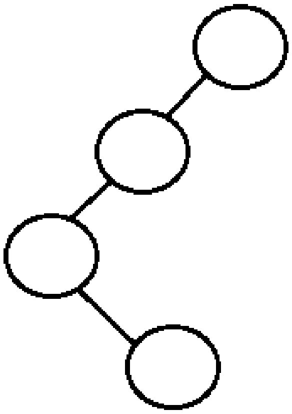
1



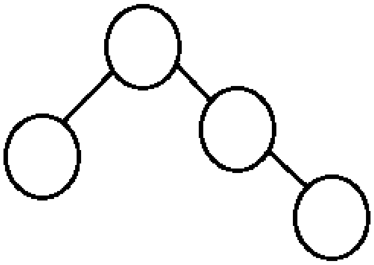
2



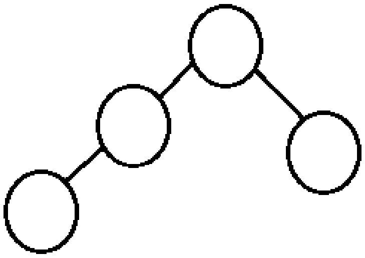
3



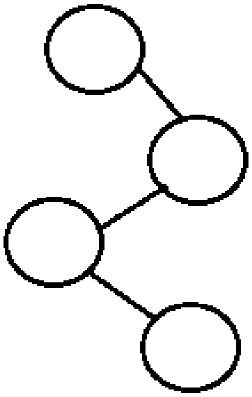
4



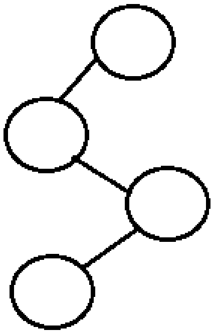
5



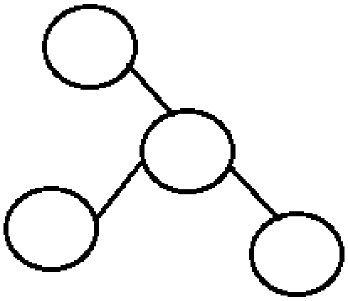
6



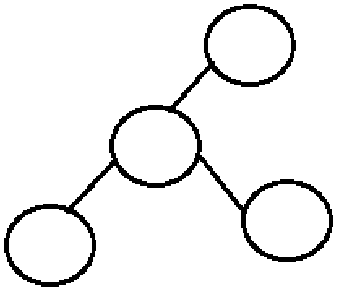
7



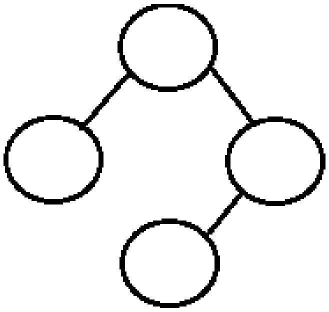
8



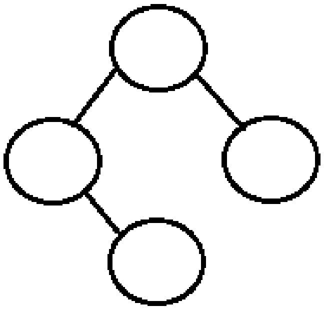
9



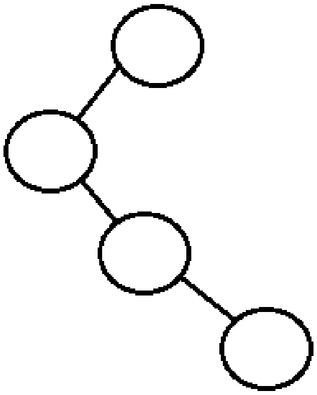
10



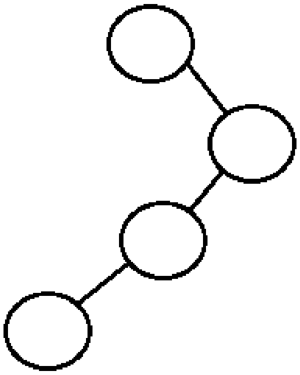
11



12



13



14

now for n nodes in general,

Let N(x) = number of binary trees for x nodes in general

then, N(0)= 1 (one empty tree)

N(1) = 1

N(2) = 2

N(3) = 5

N(3) can also be written as

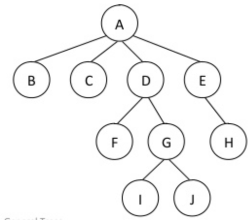
N(3) = N(2)*N(0) + N(1)*N(1) + N(0)*N(2)

in general ,

$$\mathbf{N(n) = \Sigma^{n}_{i=1} N(i-1)*N(n-i)}$$

Topic: Tree ADT
Due: 9th Nov 2020

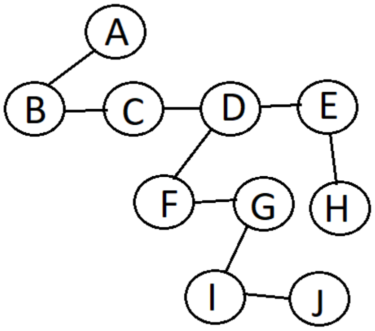
8. Convert a tree to a binary tree



SOL)

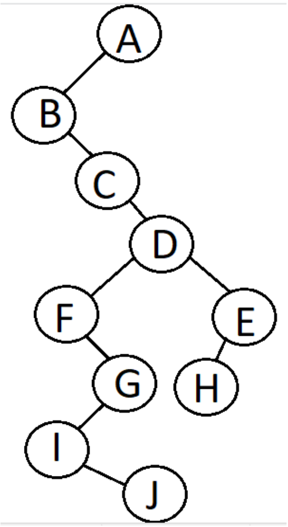
STEP 1:

Remove all links except left most links and join all sibling nodes in the same level



STEP 2:

Redraw



9. A binary tree T has ‘n’ leaf nodes. The number of nodes of degree ‘2’ in T is

Sol) In a binary tree ,

number of leaves = ‘n’

let the number of nodes with one child = N_1

let the number of nodes with two children = N_2

let e be the number of edges in the binary tree .

now we know that e = (total number of nodes) -1

$e = n + N_1 + N_2 - 1$

also, number of edges = (sum of degrees+1’s of all nodes)/2

$e = [(n*1) + (N_1*2) + (N_2*3)]/2$

$e = (n + 2N_1 + 3N_2)/2$

Combining both equations of e , we get

$(n + 2N_1 + 3N_2)/2 = n + N_1 + N_2 - 1$

$\mathbf{n = N_2 + 1}$

Number of leaf nodes = (Number of nodes with 2 children) + 1

Number of nodes with 2 children = n-1

10. Match the following (Applications of different data structures)

a) Doubly Linked list (F)Time sharing problem/ round robin scheduling algorithm
b) Stack (E)Digital Image
c) Queue (B)Recursive calls and function calls
d) Priority queue (A)The cache in the browser that allows you to hit the BACK button
e) 2D-Array (D)No longer FIFO
f) Circular linked list (G)Computer file system
g) Tree (C)CPU scheduling and Disk Scheduling

**************THE END ************