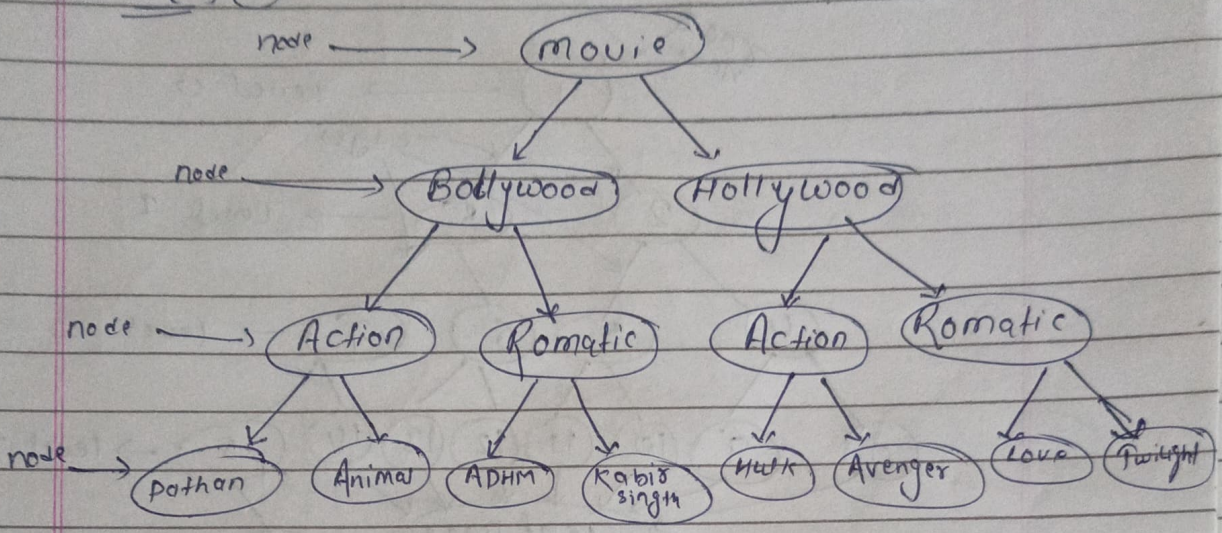
EAGAR
Page No.
Date 21/02/2024

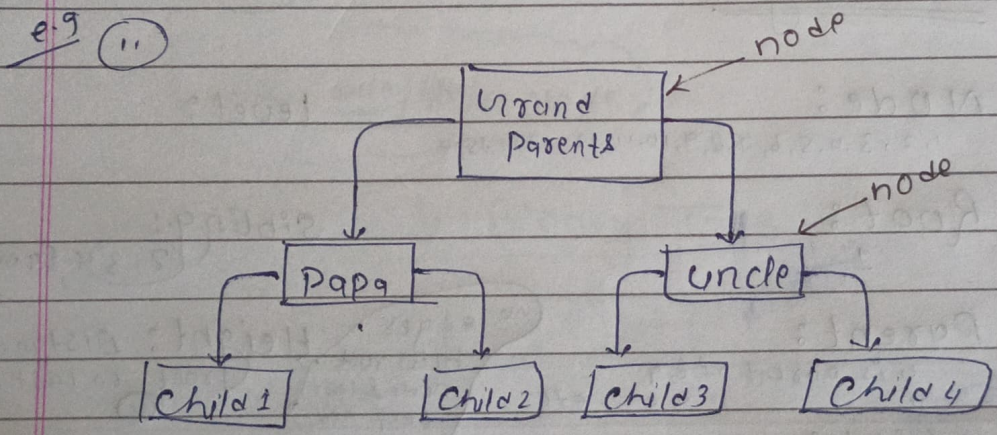
Tree

→ It is a type of data structure that represent a hierarchical relationship between data element called nodes.

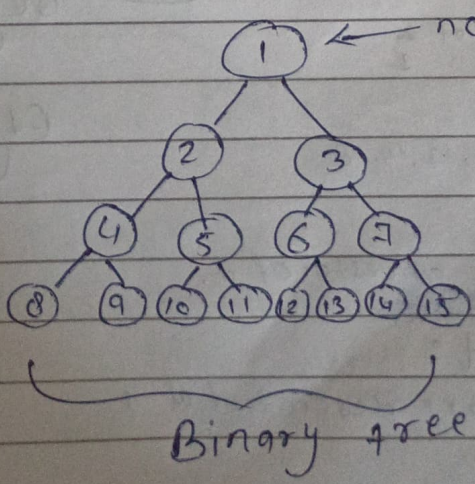
e.g (i)



e.g (ii)



e.g (iii)



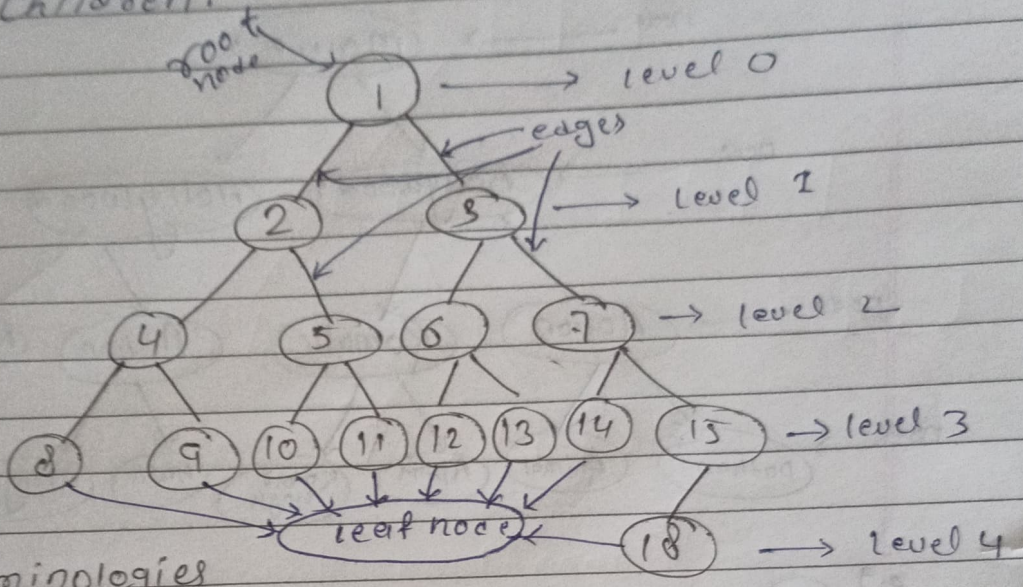
Binary tree

SAGAR

Page No.
Date 21/02/2024

Binary Tree

→ It is defined as a tree data structure where each node has atmost 2 children.



Terminologies

Node:
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 18

Root:
1

Parent:
4 is parent of 8, 9
2 " " " 4, 5 ... etc

Child:
4 is child of 2
2 " " " 1

Leaf:
8, 9, 10, 11, 12, 13, 14, 18

Ancestor:
1, 2, 4 are ancestor of 8.

Descendent:
2, 4, 8 are descendent of 1

Level:

Sibling:
2, 3 ← same parent

Height: Distance b/w root to last level.
(4)

no. edges b/w root node & last leaf node

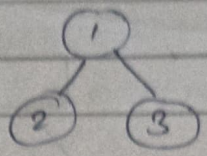
Degree: no. of child of that node.

Edges:

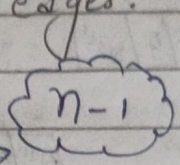
Binary tree

Atmost two child

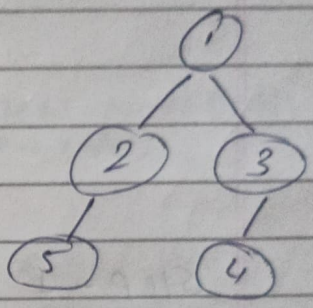
e.g.



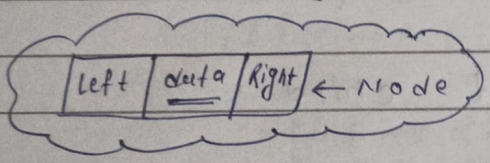
Given a binary tree with n-node find the no. of edges.



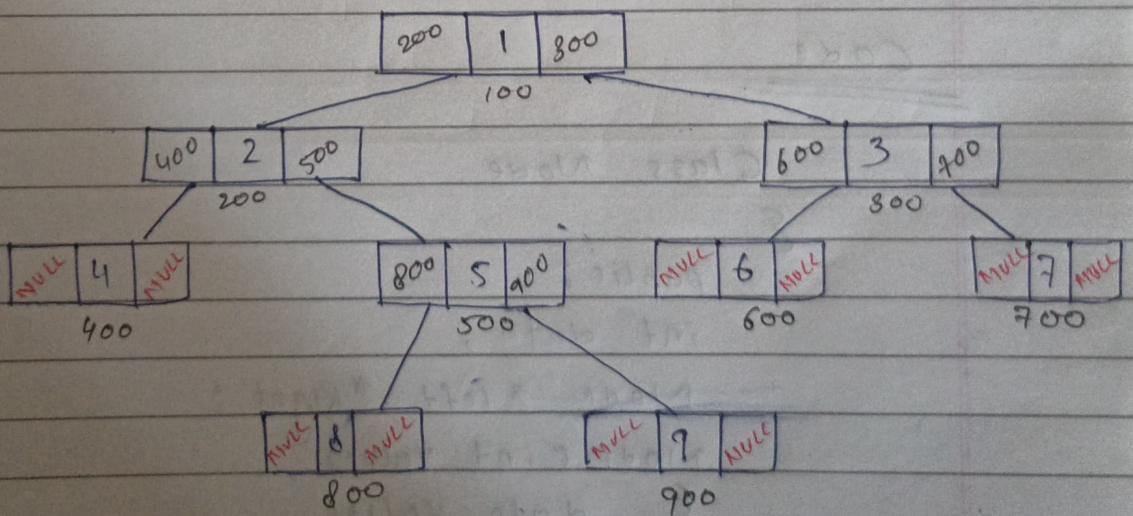
Total no. of edges.



5 node
4 edge.

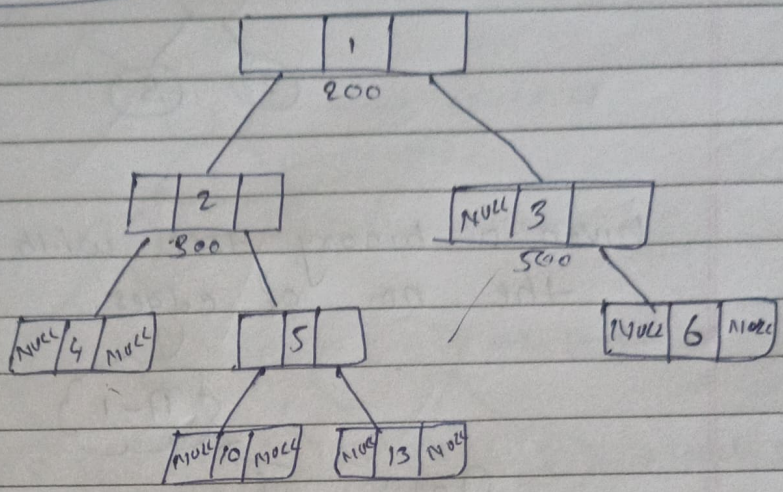


How to Create Binary Tree



i/p : 1 2 3 4 5 -1 6 -1 -1 10 13 -1 -1
-1 -1 -1 -1

-1 means NULL



Step-1 Create Root node

step-2 Left child

step-3 Right child

Code

```
Class Node
{
    public:
    int data;
    Node *left *Right;
    Node (int value)
    {   data : value
        left = Right = NULL;
};
```

```
int main()
{   int x, first, second
    cin>>x;
    queue <Node *> q;
    Node *root = new Node(x);
       q.push(root);
    while( !q.empty())
    {
        Node *temp = q.front();
              q.pop();
        cin>>first; // left child
        if(first!=-1)
        {  temp->left = new Node(first);
           q.push(temp->left);
        }

        cin>>second; // Right child
        if(second!=-1)
        {
          temp->Right = new Node(second);
           q.push(temp->Right);
    }
}
```