Memory representation of Binary tree:

A binary tree data structure is represented using two methods:

1. Array Representation
2. Linked List Representation

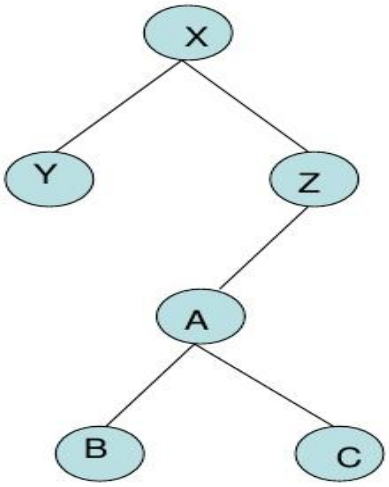
1. Array representation of Binary tree

This representation uses a single linear array ***tree*** following the below rules:

(i) The root of the tree is stored in ***tree[0].***

(ii) If a node occupies ***tree[i],*** then its left child is stored in ***tree[2*i+1],*** and its right child is stored in ***tree[2*i+2].***

Example:



	0	1	2	3	4	5	6	7	8	9	10	11	12
Tree:	X	Y	Z			A						B	C

Root is node X stored in tree [0], i.e. i=0
its left child will be stored in tree[2*i+1]=tree[2*0+1]=tree[1].
Therefore, the left child, i.e. node Y is stored in tree[1] location in the array.

The right child of node X will be stored in tree[2*i+2] =tree[2*0+2]=tree[2]
Therefore, the right child, i.e. node Z is stored in tree[2] location in the array.

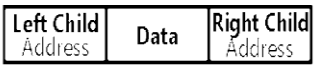
Left child of node Z at tree[2] location will be stored in tree[2*i+1]=tree[2*2+1]=tree[5] i.e. node A will be stored in tree [5]
And so on.

2. Linked list representation:

A node is divided into three fields:

- Data
- left pointer filed used to store address of left child,
- right pointer filed used to store address of right child

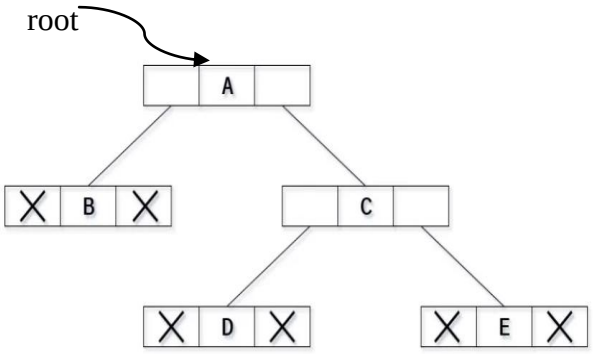
Example of a node:



Structure of a node:

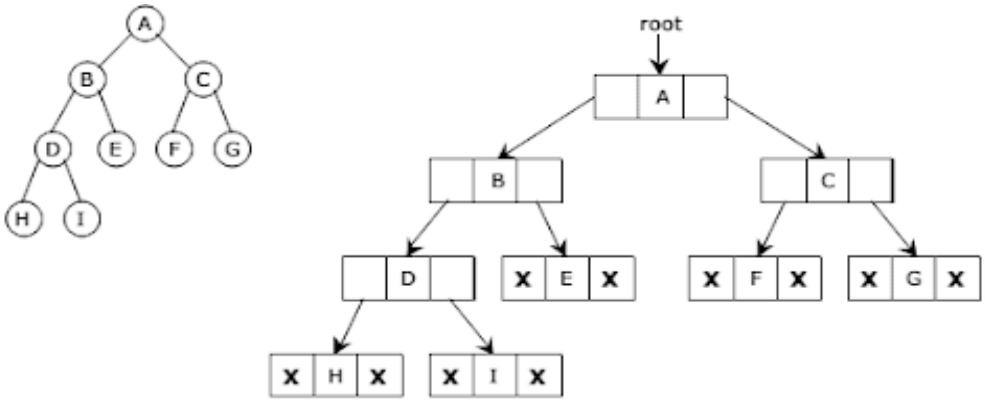
```
struct node
     {
     int data;
     struct node *left, *right;
     }*root=NULL;
```

Exmple-1:



If any of left or right child is empty then the pointers contain NULL value

Example-2 :



[Linked list representation of a binary tree]

// Cross symbol indicates NULL