

## Binary Tree Construction using linked list

```
void BinaryTree(node *root){  
    node *l, *r; char ans;
```

```
    if (root != NULL) {
```

```
        printf("\n Do you want to create Left child (Y/N) := ");  
        ans = toupper(getche());
```

```
        if (ans == 'Y') {
```

```
            l = (node*) malloc(sizeof(node));
```

```
printf("\n Enter data := ");
```

```
printf("Enter left child of %.c", root->data);
```

```
root->left = l;
```

```
l->data = getche();
```

```
l->left = NULL;
```

```
l->right = NULL;
```

```
root->left = l;
```

```
    BinaryTree(l);
```

```
}
```

```
printf("\n Do you want to create right child (Y/N) := ");
```

```
ans = toupper(getche());
```

```
if (ans == 'Y') {
```

```
    r = (node*) malloc(sizeof(node));
```

```
printf("Enter right child of %.c", root->data);
```

```
r->data = getche();
```

```
r->left = NULL;
```

```
r->right = NULL;
```

```
root->right = r;
```

```
    BinaryTree(r);
```

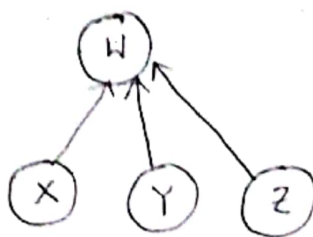
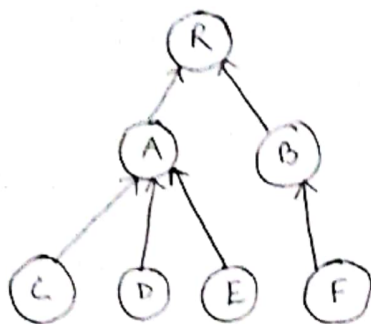
```
}
```

```
}
```

②



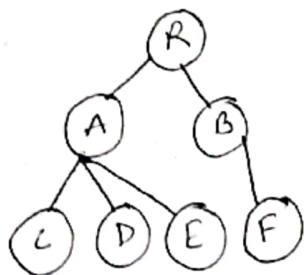
## Parent Pointer Implementation



Parent's index		0	0	1	1	1	2		7	7	7
Label	R	A	B	C	D	E	F	W	X	Y	Z
Node index	0	1	2	3	4	5	6	7	8	9	10

- Easy to find ancestors
- Difficult to find children
- Minimum space

## General Tree Implementations

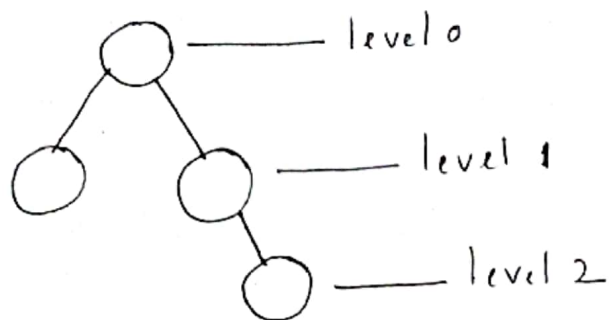


Index	Val	Par
0	R	•
1	A	0
2	B	0
3	C	1
4	D	1
5	E	1
6	F	2
•		
•		
•		

Diagram illustrating the general tree implementation using an array structure. The array has columns for Index, Value (Val), and Parent (Par). Arrows show the mapping from child nodes to their parents:

- Node 1 (A) points to Node 0 (R)
- Node 2 (B) points to Node 0 (R)
- Node 3 (C) points to Node 1 (A)
- Node 4 (D) points to Node 1 (A)
- Node 5 (E) points to Node 1 (A)
- Node 6 (F) points to Node 2 (B)

## Height of a binary search Tree:



### Method-1

$$\text{height} = \max(\text{level}_i) = 2$$

### Method-2

$$\begin{aligned}\text{height} &= \lceil \log_2 n \rceil = \lceil \log_2 4 \rceil = \lceil \log_2 2^2 \rceil \\ &= \lceil 2 \log_2 2 \rceil \\ &= \lceil 2 * 1 \rceil \\ &= 2\end{aligned}$$

### Method-3

~~height(T) = max(height(T<sub>L</sub>), height(T<sub>R</sub>)) + 1~~

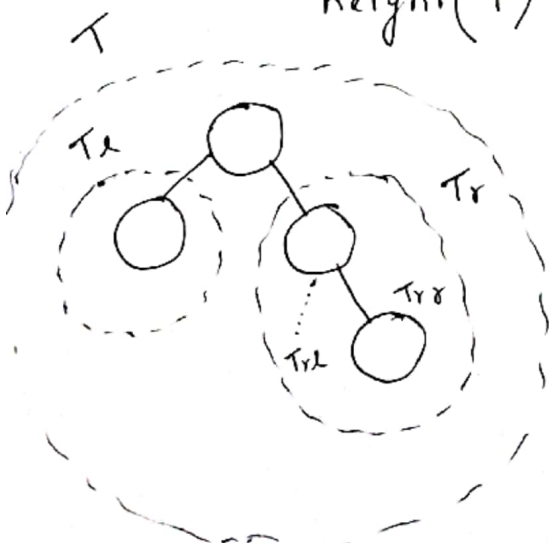
$$\text{height}(T) = \max(\text{height}(T_L), \text{height}(T_R)) + 1$$

$$= \max(0, \max(\text{height}(T_{LL}), \text{height}(T_{LR})) + 1) + 1$$

$$= \max(0, \max(-1, 0) + 1) + 1$$

$$= \max(0, 0 + 1) + 1$$

$$= \max(0, 1) + 1 = 1 + 1 = 2$$



Tree  
NULL

○  
otherwise

height  
-1

Zero

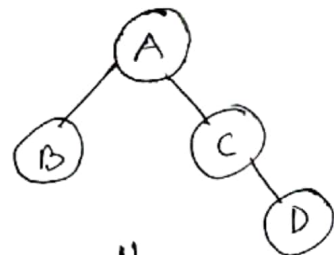
$\max(\text{height}(T_L), \text{height}(T_R)) + 1$

# Algorithm of height calculation (Method)

(5)

```
int height(node *root) {  
    if (root == NULL)  
        return -1;  
    else if ((root->left == NULL) && (root->right == NULL))  
        return 0;  
    else  
        return max(height(root->left), height(root->right))  
            + 1;  
}
```

```
int max(int x, int y) {  
    if (x > y) return x;  
    else return y;  
}
```



## Recursion Tree

height(

root
A

) = 2

return max( height(

root
B

), height(

root
C

) ) + 1 = 2

0                      1

return 0                      1

return max( height(

root
D

) ) + 1 = 1

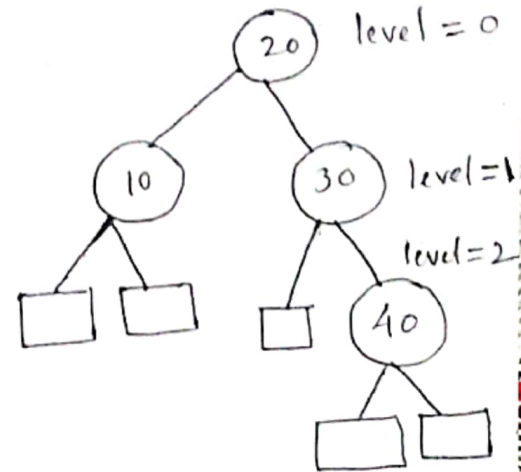
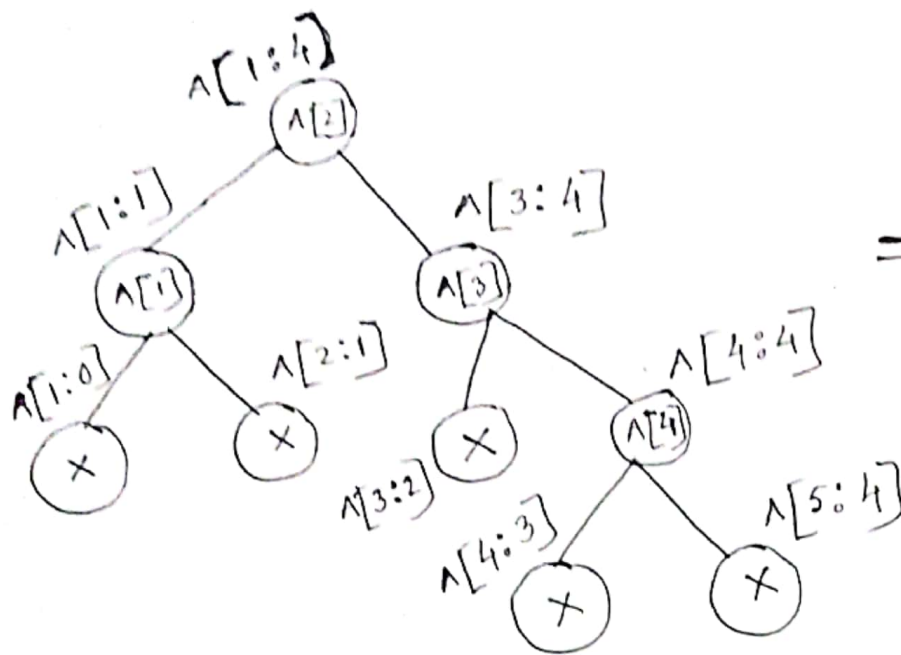
return -1                      return 0



Average Number of Successful and Unsuccessful comparison:

(6)

A	10	20	30	40
	1	2	3	4



Average Number of Successful Comparisons:

For	20	element	Comparison	=	1
For	10	"	"	=	2
For	30	"	"	=	2
For	40	"	"	=	3

[Use internal node ○ level]

$$\text{Average Comparison} = \frac{1+2+2+3}{4} = \frac{8}{4} = 2$$

Using Formula:

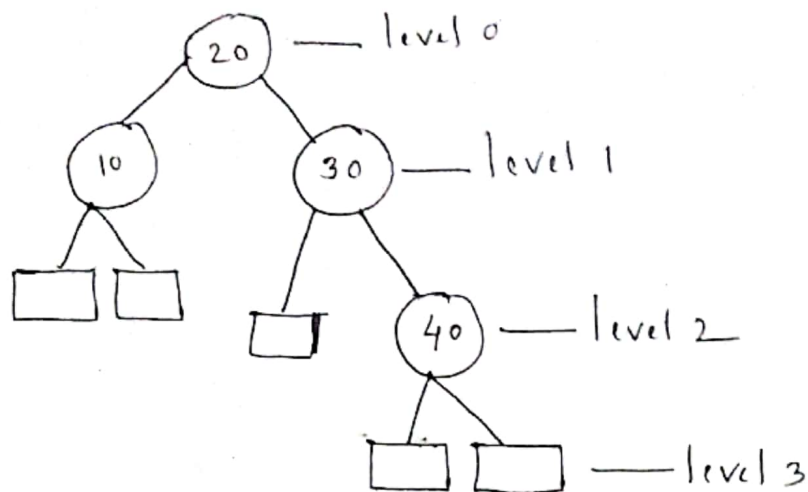
$$\text{Average Comparison} = \frac{\sum (l_i + 1)}{n}$$

$$= \frac{(0+1) + (1+1) + (1+1) + (2+1)}{4}$$

$$= 2$$

# Average Number of Unsuccessful Comparison

7



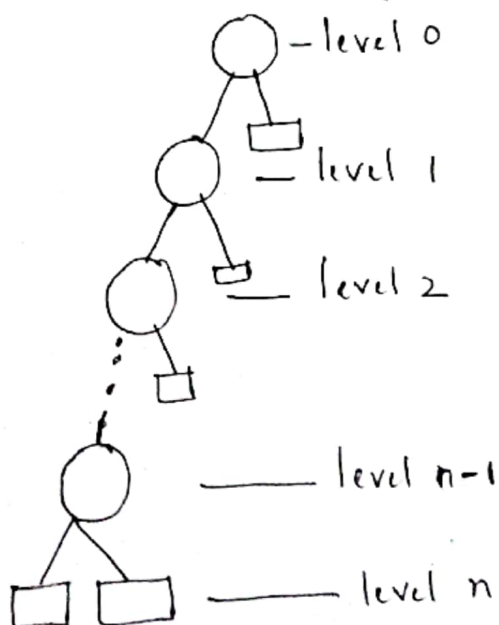
Use external node level



$$\begin{aligned} \text{Average comparison} &= \frac{\sum l_i}{h+1} \\ &= \frac{2+2+2+3+3}{5} \\ &= \frac{12}{5} \end{aligned}$$

$$= 2.4$$

Internal and External Path length calculation for one sided binary Search Tree:



$I(T)$  = Internal Path length

$$= 0+1+2+3+\dots+n-1$$

$$= \frac{(n-1)n}{2} \dots \text{eq (1)}$$

$$\left[ 1+2+3+\dots+n = \frac{n(n+1)}{2} \right]$$

$E(T)$  = External path length

$$= 1+2+3+\dots+n+n$$

$$= \frac{n(n+1)}{2} + n = n \left\{ \frac{n+1}{2} + 1 \right\} = \frac{n(n+3)}{2}$$

$$\dots \text{eq (2)}$$

$$E(T) - I(T)$$

$$= \frac{n(n+3)}{2} - \frac{(n-1)n}{2}$$

$$= \frac{n}{2} \{ (n+3) - (n-1) \} = \frac{n}{2} \times 4 = 2n$$

$$\boxed{E(T) - I(T) = 2n}$$