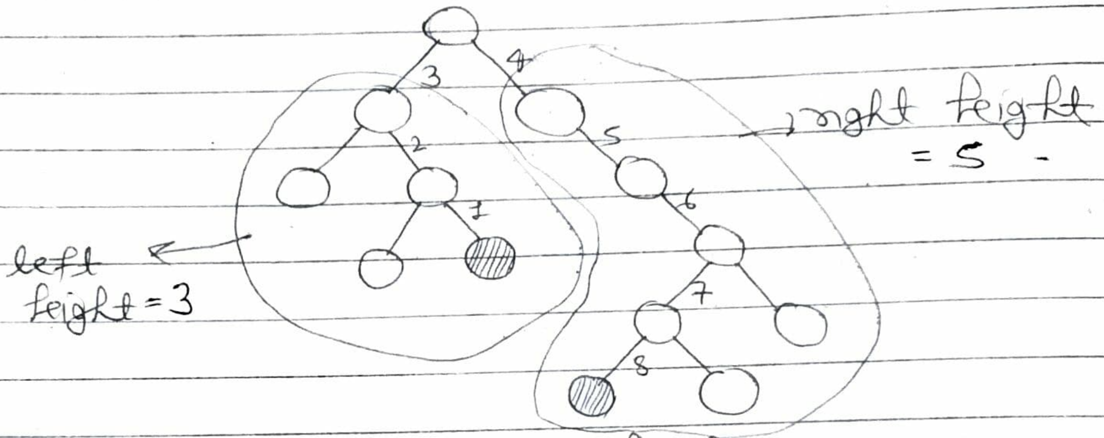
— DP On Trees —

- → General Syntax
- → How DP can be applied to Trees (Identification)
- → Diameter of a Binary Tree
- → Maximum Path sum from Any node to any.
- → Maximum Path sum from leaf to leafs
- → Diameter of N-ary Tree.

→ Identification -



Find longest path b/w 2 leaves which are shaded?

=> 8

we have to find left height & right height for all nodes which take for n node - $O(n) \times n$

$= O(n^2)$

We can use dynamic programming to reduce complexity.

→ General Syntax

```
int function_name (I/P →)
{
    Base Condition

    HYPOTHESIS

    INDUCTION
}
```

Code-

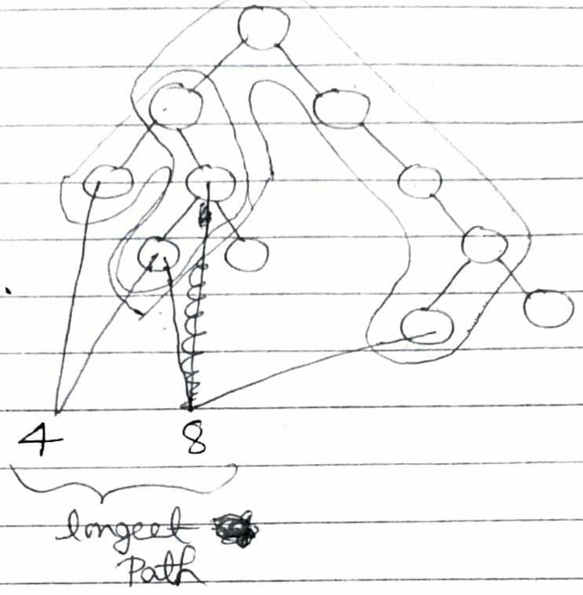
```
int solve (Node * root, int * res)
{
    if (root == NULLPTR)
        return 0;

    int l = solve (root -> left, res);
    int r = solve (root -> right, res);

    int temp = calculate temp answer
    int ans = max (temp, relation)
                           ↳ f+l+r
        res = max (res, ans)
    return temp;
}
```

* Diameter of Tree (Binary Tree) -

We have to select two leafs & find the no of nodes (including selected leaves) b/w the leafs. such that

We have to return no of nodes along the longest path b/w 2 leafs.

Ex-

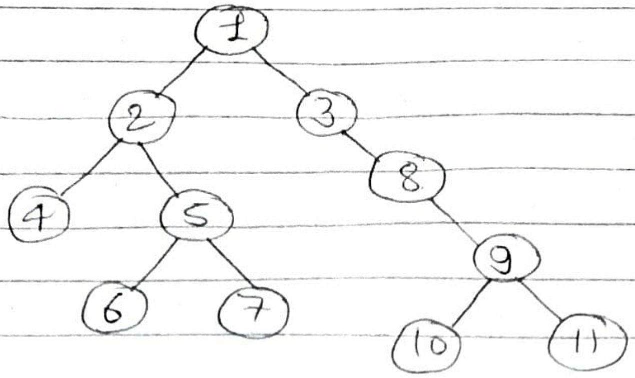


Code-

```
int solve(Node* root, int* res)
{ if(root == nullptr)
    return 0;
  int l = solve(root->left, res);
  int r = solve(root->right, res);

  int temp = max(l,r)+1;
  int ans = max(temp, 1+l+r);
  res = max(res, ans);
  return temp;
}
```

} always same

* Maximum Path sum from Any node to Any -



Code-

```
int solve(node * root, int * res)
{
  if(root == NULL)
  return 0;

  int l = solve(root->left, res)
  int r = solve(root->right, res)

  ~~int temp = max (solve (l,~~
  int temp = max(max(l,r) + root->value,
                          root->value);
  int ans = max(temp, l+r+root->value);
   res = max (res, ans);
  return temp;
}
```

* Maximum Path Sum from leaf to leaf -

```
temp = max (l,r) + root -> value;
   if (root -> left && root -> right == NULL)
                  ==
                 NULL
     temp = max (temp, root -> value);

int ans = max( temp, l+r+root -> value)
res = max (res, ans)
return temp;
```