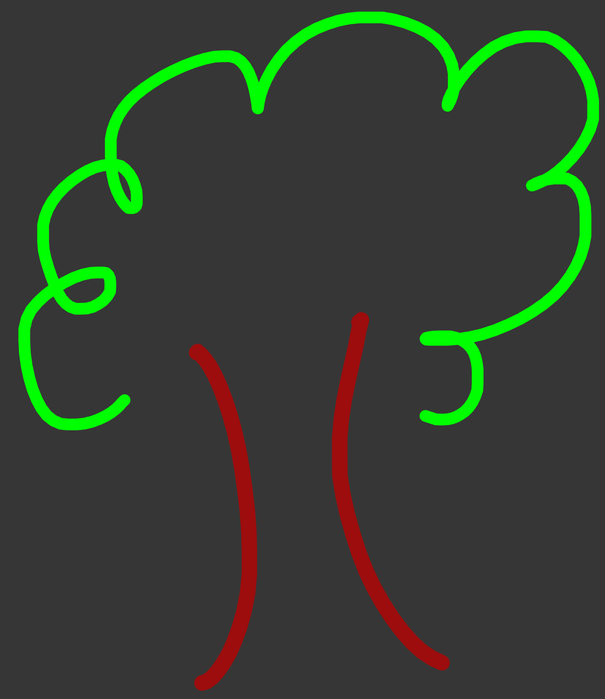
All the Best!!!

Binary Tree



Binary Tree Node Class

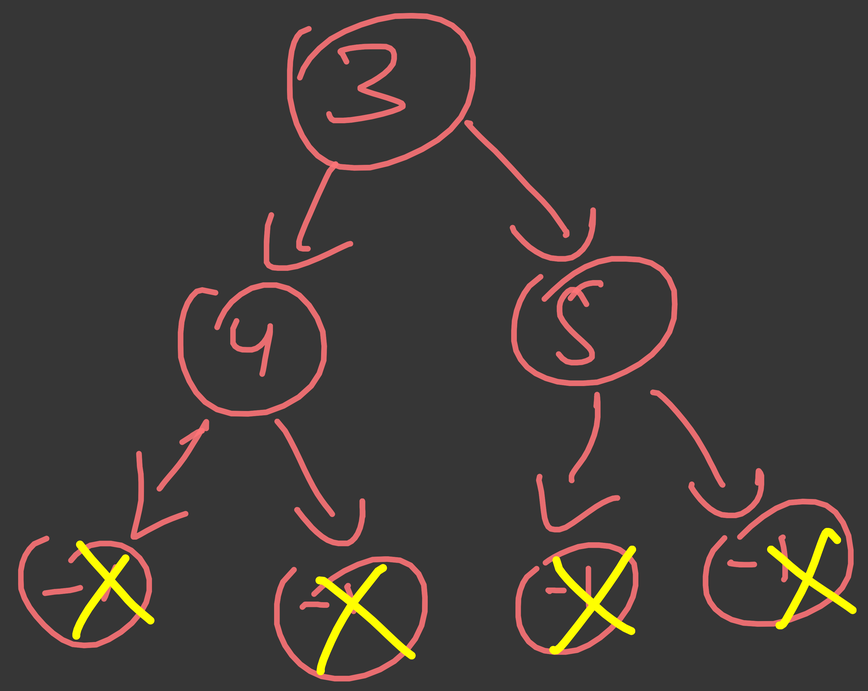
Tree Node

```
class Node {
public:
    Node *left;
    Node *right;
    int data;

Node(int d){
    data = d;
    left->next = NULL;
    right->next = NULL;
};
```

Build Tree :-

3 4 -1 -1 5 -1 -1




```
Node * buildTree ( ){
        int d;
        cin >> d;
        if (d == -1)
            return NULL;
        root = new Node(d);
        root -> left = buildTree ( )
        root -> right = buildTree( )
 return root;
}
```

Print Tree :- Pre-Order Traversal

```
void Print(Node *root){
    if(root==NULL) return;
    Cout<< root->data << endl;
    Print(root->left);
    Print(root->right);
}
```

1 2 3 4 5 6

In-Order Traversal

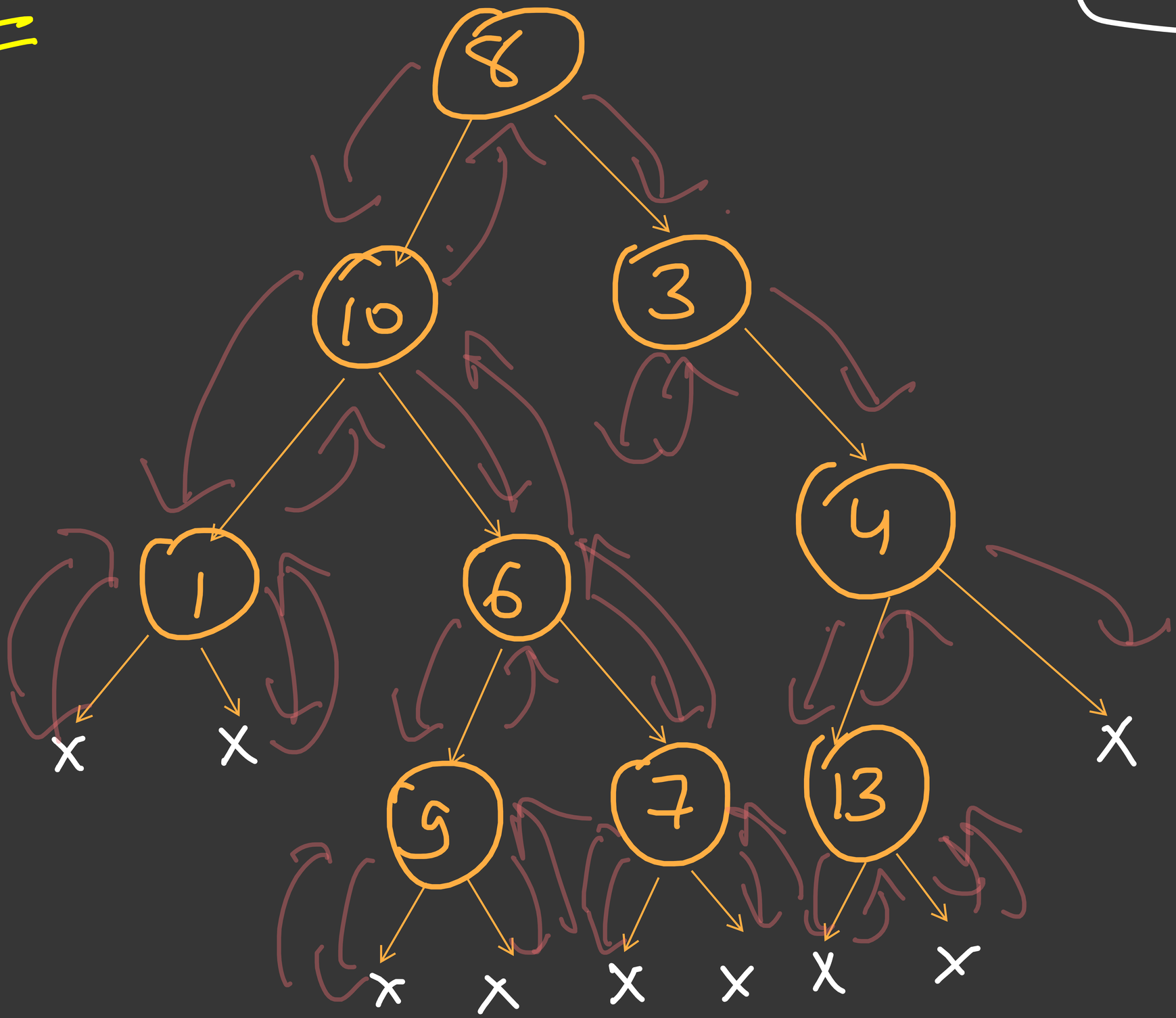
```
void Print(Node* root){
    if(root==NULL) return;

    Print(root->left);
    cout<< root->data << endl;
    Print(root->right);

}
```

InOrder

X -> NULL



1 10 9 6 7 8 3 13 4

Post Order Traversal :-

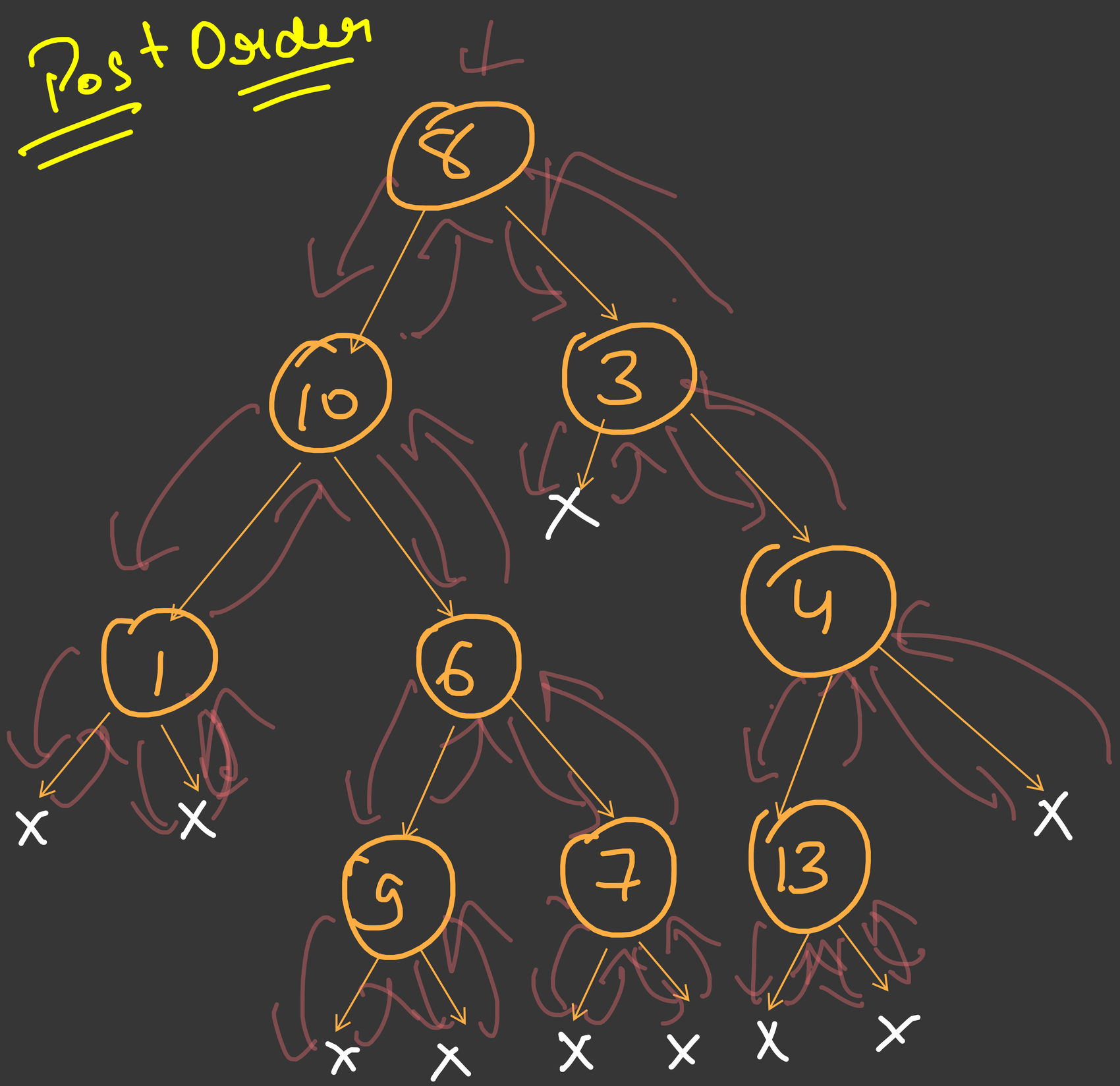
```
void Print(Node *root){
    if(root==NULL) return;

    Print(root->left);

    Print(root->right);

    cout<< root->data << endl;
}
```

Post Order

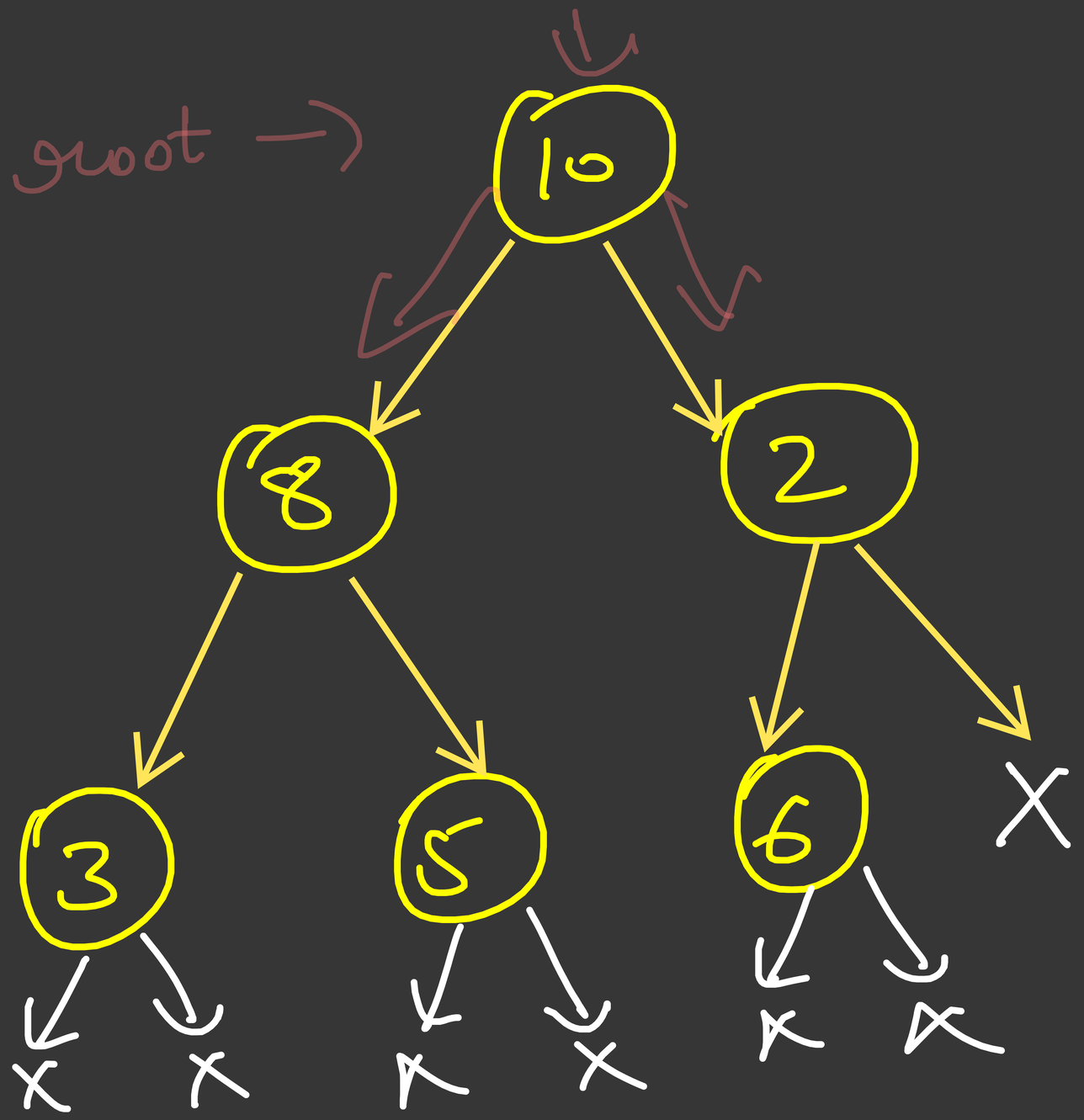


1
9
7
6
10
13
4
3
8

Pre-Order Iterative :-

Algo :-

1. Push root in Stack
2. While (Stack is not empty)
   - i) Pop and Store top node from Stack
   - ii) print it
   - iii) push right if exist
   - iv) push left if exist



O/P:- 10 8 3 5 2 6

Stack