LAB - (8) : - Binary Search Tree

15/1/24

```
#include <stdio.h>
#include <stdlib.h>

struct node
{
    int data;
    struct node *left;
    struct node *right;
};

struct node *newNode(int data)
{
    struct node *node = (struct node*) malloc
                        (sizeof(struct node));

    node->data = data;
    node->left = node->right = NULL;
    return node;
}
```

15/2/2024

```
struct node *insert(struct node *root, int data)
{
    if (root == NULL)
        return newNode(data);
    if (data <= root->data)
        root->left = insert(root->left, data);
    else
        root->right = insert(root->right, data);
    return root;
}
```

```
void inorder (struct node *temp)
{
    if (temp == NULL)
        return;
    inorder (temp -> left);
    printf ("%d", temp -> data);
    inorder (temp -> right);
}

void preorder (struct node * temp)
{
    if (temp == NULL)
        return;
    printf ("%d", temp -> data);
    preorder (temp -> left);
    preorder (temp -> right);
}

void postorder (struct node * temp)
{
    if (temp == NULL)
        return;
    postorder (temp -> left);
    postorder (temp -> right);
    printf ("%d", temp -> data);
}
```

```
int main()
{
  struct node * root = NULL;
  int data, choice;

  do
  {
    printf("Enter your choice: \n 1. Insert \n 2. Print
    Inorder \n 3. Print Preorder \n 4. Print Postorder \n
    5. Exit \n");
    scanf("%d", &choice);
    switch (choice)
    {
      case 1: printf("Enter value to be inserted: ");
              scanf("%d", &data);
              root = insert (root, data);
              break;
      case 2: printf("Inorder traversal of binary tree
                      is \n");
              inorder (root);
              printf("\n");
              break;
```

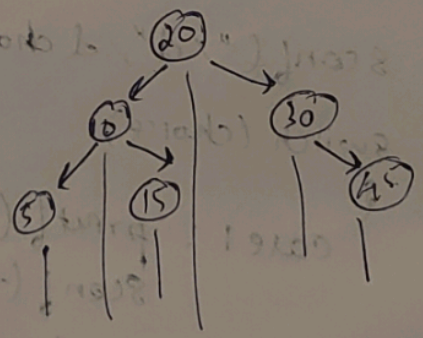
```
case 3: printf("Preorder traversal of binary tree is \n");
        preorder(root);
        printf("\n");
        break;
case 4: printf("Postorder traversal of binary tree is \n");
        postorder(root);
        printf("\n");
        break;

case 5  : printf("Exiting...");
          break;

default: printf("Invalid choice");
}
} while(choice != 5);
return 0;
}
```

Output:

```
Enter your choice:
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit
1
Enter value to be inserted : 20
Enter your choice:
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit
```



1
Enter value to be inserted : 10
Enter your choice :
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit

1
Enter value to be inserted : 30
Enter your choice :
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit

1
Enter value to be inserted : 5
Enter your choice :
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit

1
Enter value to be inserted : 15
Enter your choice :
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit

1
Enter value to be inserted : 45

Enter your choice:
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit

2
Inorder traversal of binary tree is:
5 10 15 20 30 45

Enter your choice:
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit
3
Preorder traversal of binary tree is:
20 10 5 15 30 45

Enter your choice:
1. Insert
2. Print Inorder
3. Print Preorder
4. Print Postorder
5. Exit
4
Postorder traversal of binary tree is:
5 15 10 45 30 20

Leetcode → Rotate List

```
struct ListNode* rotateRight(struct ListNode *head, int k)
{
    if(head == NULL || head->next == NULL || k==0)
        return head;

    int len = 1;
    struct ListNode *tail = head;
    while (tail->next != NULL)
    {
        tail = tail->next;
        len++;
    }
    k = k % len;
    if(k == 0)
        return head;
    struct ListNode *p = head;
    for(int i=0; i < len - k - 1; i++)
    {
        p = p->next;
    }
    tail->next = head;
    head = p->next;
    p->next = NULL;
    return head;
}
```