***Week -9***

**Q1**

**AIM:** Write a menu driven program to implement Binary tree with the following operations.

1. Insertion
2. Preorder
3. Inorder
4. Postorder

**Description:**

1. START
2. Create a node class for having left and right attributes for each object of binary tree class
3. Now create a binary tree class and define the required methods as mentioned in the given problem.

**Insertion:**

1. START
2. Enter the value to be inserted in data variable.
3. if self.root is None:
4. self.root=Node(data)
5. else:
6. ptr=self.root
7. n=int(input("\n1. Left \t 2. Right\nEnter which side u want to insert: "))
8. while (ptr.left is not None) or (ptr.right is not None):
10. if n==1 and ptr.left is not None:
11. ptr=ptr.left
12. n=int(input("\n1. Left \t 2. Right\nEnter which side u want to insert: "))
13. elif n==2 and ptr.right is not None:
14. ptr=ptr.right
15. n=int(input("\n1. Left \t 2. Right\nEnter which side u want to insert: "))
16. elif (n==1 and ptr.left is None) or (n==2 and ptr.right is None):
17. break
18. else:
19. print("Wrong choice, try again")
20. newnode=Node(data)
21. if n==1:
22. ptr.left=newnode
23. elif n==2:
24. ptr.right=newnode
25. STOP

**Preorder Traversal**

1. def preorder(self,root):
2. if (root):
3. print(root.data,end=' ')
4. self.preorder(root.left)
5. self.preorder(root.right)

**Inorder Traversal**

1. def inorder(self,root):
2. if (root):
3. self.inorder(root.left)
4. print(root.data,end=' ')
5. self.inorder(root.right)

**Postorder Traversal**

1. def postorder(self,root):
2. if (root):
3. self.postorder(root.left)
4. self.postorder(root.right)
5. print(root.data,end=' ')
6. Now outside the class, create a binary tree object and do the given operations as required using a while loop
7. STOP

**Program:**

*class Node:*

*def \_\_init\_\_(self,data):*

*self.data=data*

*self.left=None*

*self.right=None*

*class Binary\_tree:*

*def \_\_init\_\_(self):*

*self.root=None*

*def insertion(self):*

*data=int(input("Enter the value: "))*

*if self.root is None:*

*self.root=Node(data)*

*print(self.root.data)*

*else:*

*ptr=self.root*

*n=int(input("\n1. Left \t 2. Right\nEnter which side u want to insert: "))*

*while (ptr.left is not None) or (ptr.right is not None):*

*if n==1 and ptr.left is not None:*

*ptr=ptr.left*

*n=int(input("\n1. Left \t 2. Right\nEnter which side u want to insert: "))*

*elif n==2 and ptr.right is not None:*

*ptr=ptr.right*

*n=int(input("\n1. Left \t 2. Right\nEnter which side u want to insert: "))*

*elif (n==1 and ptr.left is None) or (n==2 and ptr.right is None):*

*break*

*else:*

*print("Wrong choice, try again")*

*newnode=Node(data)*

*if n==1:*

*ptr.left=newnode*

*elif n==2:*

*ptr.right=newnode*

*def preorder(self,root):*

*if (root):*

*print(root.data,end=' ')*

*self.preorder(root.left)*

*self.preorder(root.right)*

*def inorder(self,root):*

*if (root):*

*self.inorder(root.left)*

*print(root.data,end=' ')*

*self.inorder(root.right)*

*def postorder(self,root):*

*if (root):*

*self.postorder(root.left)*

*self.postorder(root.right)*

*print(root.data,end=' ')*

*bt=Binary\_tree()*

*while True:*

*n=int(input("\n 1. Insertion\t2. Preorder\t3. Inorder\t4. Postorder\t5. Exit\nEnter your choice: "))*

*if n==1:*

*bt.insertion()*

*elif n==2:*

*print("The preorder traversal is: ")*

*bt.preorder(bt.root)*

*elif n==3:*

*print("The Inorder traversal is: ")*

*bt.inorder(bt.root)*

*elif n==4:*

*print("The postorder traversal is: ")*

*bt.postorder(bt.root)*

*elif n==5:*

*exit()*

*else:*

*print("Wrong choice, try again")*

**Output:**



**Time complexity:**

Insertion :O(log(n)) to base 2

**Conclusion:** The code is error free and it runs as expected