BINARY TREE TRAVERSAL

Exp. No.:
AIM:

ALGORITHM:

PROGRAM:

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

struct Node
{
   int key;
   struct Node *left, *right;
};
typedef struct Node node;

node *minRoot(node *root)
{
   node *current = root;
   while (current && current->left != NULL)
   {
      current = current->left;
   }
   return current;
}

node *insert(node *root, int data)
{
   node *temp;
   temp = (node *)malloc(sizeof(node));
   temp->key = data;
   temp->left = NULL;
   temp->right = NULL;
   if (root == NULL)
   {
      return temp;
   }
   if (root->key < data)
   {
      root->right = insert(root->right, data);
   }
```

```
    else if (root->key > data)
    {
        root->left = insert(root->left, data);
    }
    return root;
}

void inorder(node *root)
{
    if (root != NULL)
    {
        inorder(root->left);
        printf(" %d ", root->key);
        inorder(root->right);
    }
}

void preorder(node *root)
{
    if (root != NULL)
    {
        printf(" %d ", root->key);
        preorder(root->left);
        preorder(root->right);
    }
}

void postorder(node *root)
{
    if (root != NULL)
    {
        postorder(root->left);
        postorder(root->right);
        printf(" %d ", root->key);
    }
}

int main()
{
```

```
    node *root = NULL;
    int ch = 0;
    int temp;
    printf("Do You Want to Create Tree(1 to Start): ");
    scanf("%d", &ch);
    while (ch)
    {
       printf("Enter Node Data: ");
       scanf("%d", &temp);
       root = insert(root, temp);
       printf("Do You Want to Continue(0 to Exit): ");
       scanf("%d", &ch);
    }
    printf("\nTree Traversal:");
    printf("\nInOrder:");
    inorder(root);
    printf("\nPreOrder:");
    preorder(root);
    printf("\nPostOrder:");
    postorder(root);
    printf("\n");
    getch();
    clrscr();
    return 0;
}
```

OUTPUT:

```
C:\TURBOC3\BIN>TC
Do You Want to Create Tree(1 to Start): 1
Enter Node Data: 5
Do You Want to Continue(0 to Exit): 1
Enter Node Data: 2
Do You Want to Continue(0 to Exit): 1
Enter Node Data: 4
Do You Want to Continue(0 to Exit): 1
Enter Node Data: 3
Do You Want to Continue(0 to Exit): 1
Enter Node Data: 6
Do You Want to Continue(0 to Exit): 1
Enter Node Data: 87
Do You Want to Continue(0 to Exit): 1
Enter Node Data: 7
Do You Want to Continue(0 to Exit): 1
Enter Node Data: 9
Do You Want to Continue(0 to Exit): 0

Tree Traversal:
InOrder: 2  3  4  5  6  7  9  87
PreOrder: 5  2  4  3  6  87  7  9
PostOrder: 3  4  2  9  7  87  6  5
```

RESULT: