Aim

Perform inorder, preorder, postorder traversals of a binary tree.

Algorithm

- Inorder: Left → Root → Right
- Preorder: Root → Left → Right
- Postorder: Left → Right → Root

C Code

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

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

struct Node* createNode(int data){
    struct Node* n=(struct Node*)malloc(sizeof(struct Node));
    n->data=data; n->left=n->right=NULL;
    return n;
}

void inorder(struct Node* root){
    if(!root) return;
    inorder(root->left);
    printf("%d ",root->data);
    inorder(root->right);
}

void preorder(struct Node* root){
    if(!root) return;
    printf("%d ",root->data);
    preorder(root->left);
```

```
    preorder(root->right);
}

void postorder(struct Node* root){
    if(!root) return;
    postorder(root->left);
    postorder(root->right);
    printf("%d ",root->data);
}

int main(){
    struct Node* root=createNode(1);
    root->left=createNode(2);
    root->right=createNode(3);
    root->left->left=createNode(4);
    root->left->right=createNode(5);

    printf("Inorder: "); inorder(root); printf("\n");
    printf("Preorder: "); preorder(root); printf("\n");
    printf("Postorder: "); postorder(root); printf("\n");
    return 0;
}
```

Input

Tree:

```
     1
    / \
   2   3
  / \
 4   5
```

Output

```
Inorder: 4 2 5 1 3
Preorder: 1 2 4 5 3
Postorder: 4 5 2 3 1
```

Result

All traversals performed correctly.