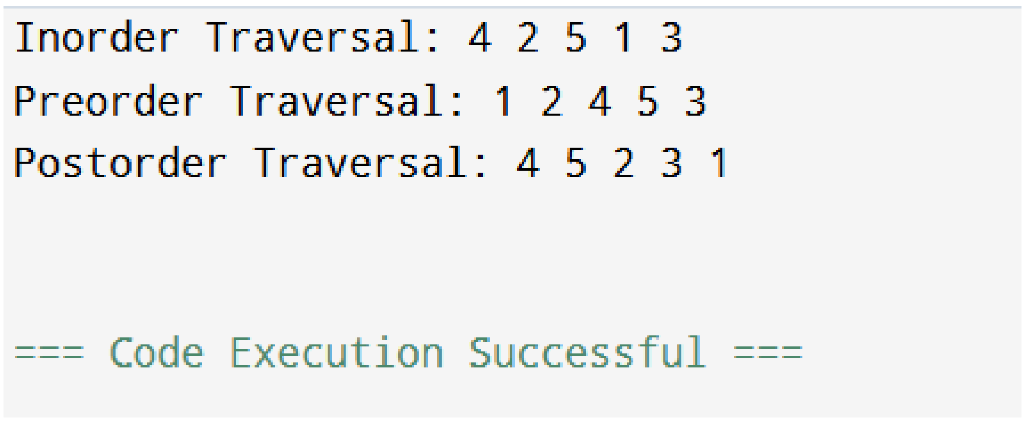
Experiment 19: Binary Tree Traversing

Code:

```
#include  <stdio.h>

#include <stdlib.h>

struct node {

   int data;

   struct node *left, *right;

};

struct node* createNode(int value) {

   struct node* newNode = (struct node*)malloc(sizeof(struct node));

   newNode->data = value;

   newNode->left = newNode->right = NULL;

   return newNode;

}

void inorder(struct node* root) {

   if (root == NULL) return;

   inorder(root->left);

   printf("%d ", root->data);

   inorder(root->right);

}

void preorder(struct node* root) {

   if (root == NULL) return;

   printf("%d ", root->data);

   preorder(root->left);

   preorder(root->right);

}

void postorder(struct node* root) {
   if (root == NULL) 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 Traversal: ");
   inorder(root);
   printf("\n");
   printf("Preorder Traversal: ");
   preorder(root);
   printf("\n");
   printf("Postorder Traversal: ");
   postorder(root);
   printf("\n");
   return 0;
}
```

Output:



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


=== Code Execution Successful ===
```