Practical-5

Aim: Create a Binary Tree and perform Tree traversals (Preorder, Postorder, Inorder) using the concept of recursion.

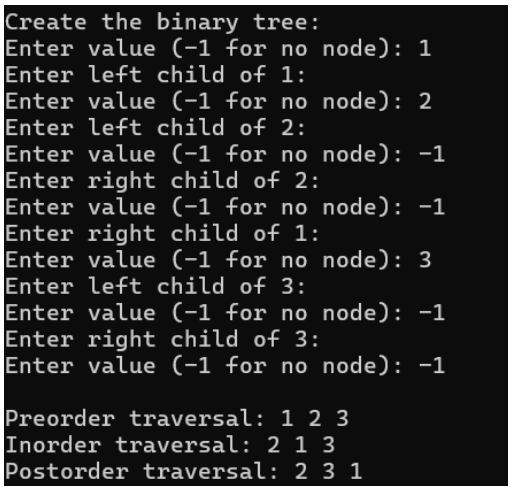
Code:

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

// Define structure for a tree node
struct Node
{
   int data;
   struct Node* left;
   struct Node* right;
};
// function to create a new node
struct Node* createNode(int value)
{
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode->data = value;
   newNode->left = NULL;
   newNode->right = NULL;
   return newNode;
}
// function to insert nodes in a binary tree manually
struct Node* insertNode()
{
   int value;
   printf("Enter value (-1 for no node): ");
   scanf("%d", &value);
   if (value == -1)
   {
      return NULL;  // No node is created if input is -1
   }
   struct Node* node = createNode(value);
   // recursive insertion for left child
   printf("Enter left child of %d:\n", value);
   node->left = insertNode();
   // recursive insertion for right child
   printf("Enter right child of %d:\n", value);
   node->right = insertNode();
   return node;
}
// Preorder traversal (Root -> Left -> Right)
void preorderTraversal(struct Node* node)
{
   if (node == NULL)
   {
      return;
   }
   printf("%d ", node->data);       // Visit root
   preorderTraversal(node->left);   // Traverse left subtree
```

```
    preorderTraversal(node->right);  // Traverse right subtree
}
// Inorder traversal (Left -> Root -> Right)
void inorderTraversal(struct Node* node)
{
    if (node == NULL)
    {
        return;
    }
    inorderTraversal(node->left);    // Traverse left subtree
    printf("%d ", node->data);       // Visit root
    inorderTraversal(node->right);   // Traverse right subtree
}
// Postorder traversal (Left -> Right -> Root)
void postorderTraversal(struct Node* node)
{
    if (node == NULL)
    {
        return;
    }
    postorderTraversal(node->left);  // Traverse left subtree
    postorderTraversal(node->right); // Traverse right subtree
    printf("%d ", node->data);       // Visit root
}
int main()
{
    struct Node* root = NULL;
    // Creating the binary tree from user input
    printf("Create the binary tree:\n");
    root = insertNode();
    // Perform tree traversals
    printf("\nPreorder traversal: ");
    preorderTraversal(root);
    printf("\nInorder traversal: ");
    inorderTraversal(root);
    printf("\nPostorder traversal: ");
    postorderTraversal(root);
    return 0;
}
```

Output:



```
Create the binary tree:
Enter value (-1 for no node): 1
Enter left child of 1:
Enter value (-1 for no node): 2
Enter left child of 2:
Enter value (-1 for no node): -1
Enter right child of 2:
Enter value (-1 for no node): -1
Enter right child of 1:
Enter value (-1 for no node): 3
Enter left child of 3:
Enter value (-1 for no node): -1
Enter right child of 3:
Enter value (-1 for no node): -1

Preorder traversal: 1 2 3
Inorder traversal: 2 1 3
Postorder traversal: 2 3 1
```