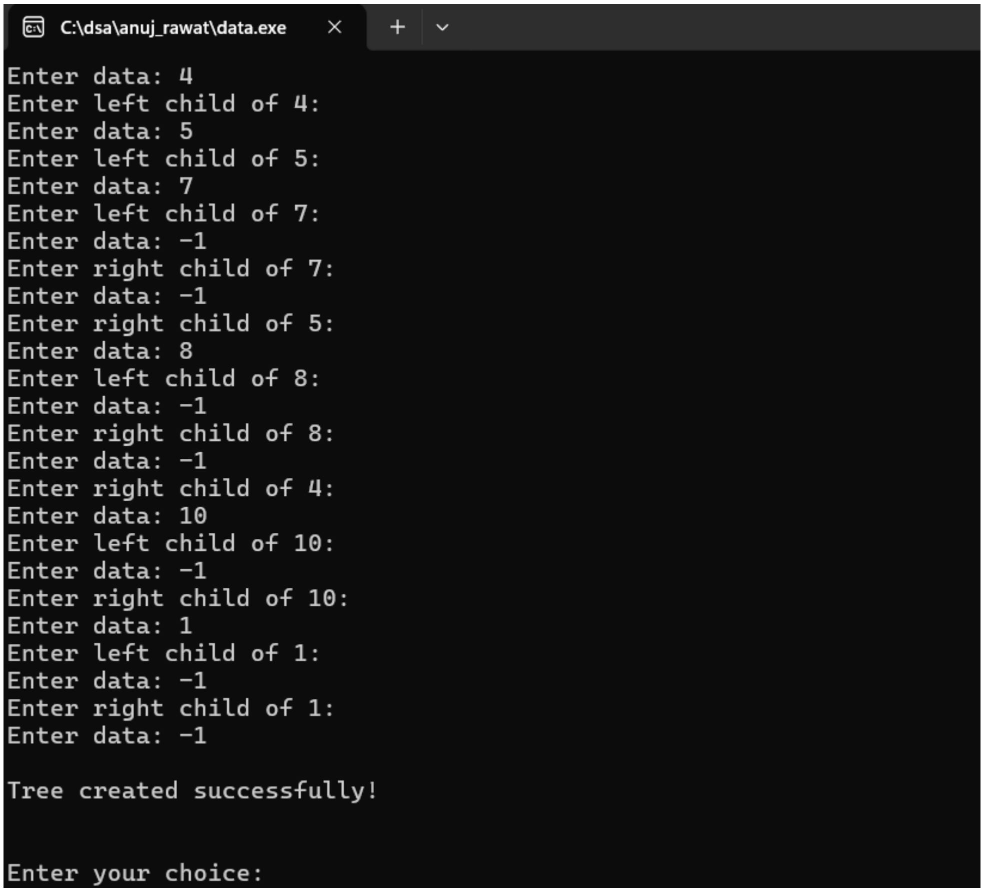
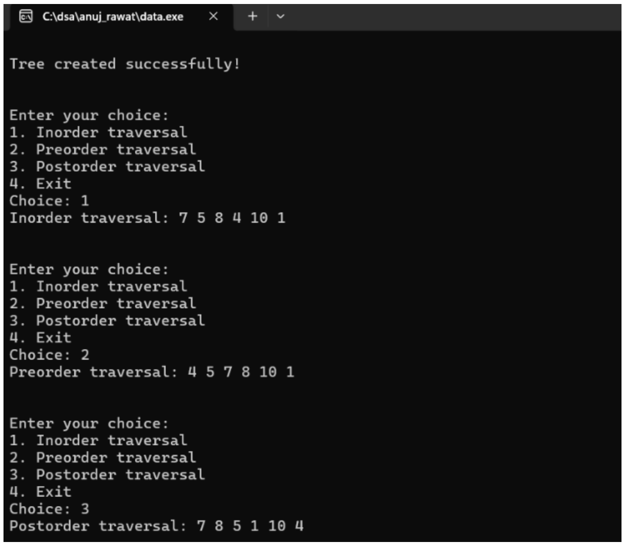
```
#include <stdio.h>
#include <stdlib.h>

struct node {
    int data;
    struct node *left, *right;
};

struct node* create() {
    int x;
    struct node* newnode;
    newnode = (struct node*)malloc(sizeof(struct node));

    printf("Enter data: ");
    scanf("%d", &x);

    if (x == -1) {
        return 0;
    }

    newnode->data = x;
    printf("Enter left child of %d:\n", x);
    newnode->left = create();
    printf("Enter right child of %d:\n", x);
    newnode->right = create();
    return newnode;
}

void preorder(struct node* root) {
    if (root == 0)
        return;
    printf(" %d", root->data);
    preorder(root->left);
    preorder(root->right);
}

void inorder(struct node* root) {
    if (root == 0)
        return;
    inorder(root->left);
    printf(" %d", root->data);
    inorder(root->right);
}

void postorder(struct node* root) {
    if (root == 0)
        return;
    postorder(root->left);
    postorder(root->right);
    printf(" %d", root->data);
}
```

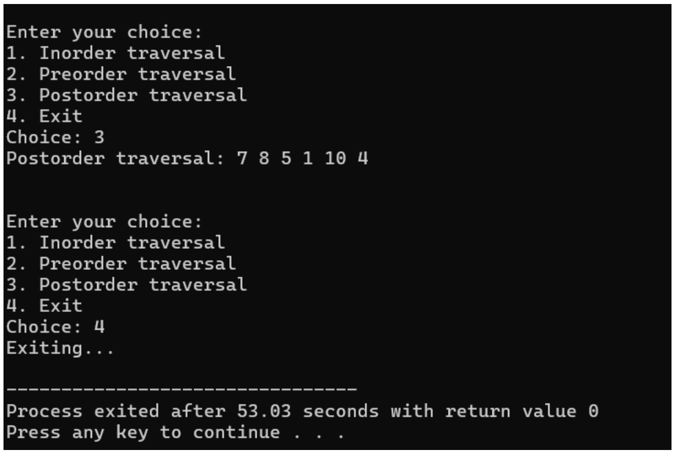
```
int main() {
    struct node* root;
    root = create();
    printf("\nTree created successfully!\n");

    while (1) {
        int ch;
        printf("\n\nEnter your choice:\n");
        printf("1. Inorder traversal\n");
        printf("2. Preorder traversal\n");
        printf("3. Postorder traversal\n");
        printf("4. Exit\n");
        printf("Choice: ");
        scanf("%d", &ch);

        switch (ch) {
            case 1:
                printf("Inorder traversal:");
                inorder(root);
                printf("\n");
                break;
            case 2:
                printf("Preorder traversal:");
                preorder(root);
                printf("\n");
                break;
            case 3:
                printf("Postorder traversal:");
                postorder(root);
                printf("\n");
                break;
            case 4:
                printf("Exiting...\n");
                exit(0);
            default:
                printf("Invalid choice!\n");
        }
    }

    return 0;
}
```



```
Enter data: 4
Enter left child of 4:
Enter data: 5
Enter left child of 5:
Enter data: 7
Enter left child of 7:
Enter data: -1
Enter right child of 7:
Enter data: -1
Enter right child of 5:
Enter data: 8
Enter left child of 8:
Enter data: -1
Enter right child of 8:
Enter data: -1
Enter right child of 4:
Enter data: 10
Enter left child of 10:
Enter data: -1
Enter right child of 10:
Enter data: 1
Enter left child of 1:
Enter data: -1
Enter right child of 1:
Enter data: -1

Tree created successfully!


Enter your choice:
```



```
Tree created successfully!


Enter your choice:
1. Inorder traversal
2. Preorder traversal
3. Postorder traversal
4. Exit
Choice: 1
Inorder traversal: 7 5 8 4 10 1


Enter your choice:
1. Inorder traversal
2. Preorder traversal
3. Postorder traversal
4. Exit
Choice: 2
Preorder traversal: 4 5 7 8 10 1


Enter your choice:
1. Inorder traversal
2. Preorder traversal
3. Postorder traversal
4. Exit
Choice: 3
Postorder traversal: 7 8 5 1 10 4
```



```
Enter your choice:
1. Inorder traversal
2. Preorder traversal
3. Postorder traversal
4. Exit
Choice: 3
Postorder traversal: 7 8 5 1 10 4


Enter your choice:
1. Inorder traversal
2. Preorder traversal
3. Postorder traversal
4. Exit
Choice: 4
Exiting...

--------------------------------
Process exited after 53.03 seconds with return value 0
Press any key to continue . . .
```