**AIM:**

To write a c program to perform Tree traversal techniques.

(i) Inorder traversal

(ii) Preorder traversal

(iii) Postorder traversal

**ALGORITHM:**

**Step 1:** Start.

**Step 2:** Read number of nodes in the tree.

**Step 3:** Insert data into tree

**Step 4:** Read choice from the user.

**Step 5:** Perform Inorder, Preorder, Postorder traversal or exit as per the choice.

**Step 6:**  Stop.

**INORDER TRAVERSAL:**

**Step 1:** Traverse the left subtree in inorder.

**Step 2:** Visit the root node.

**Step 3:** Traverse the right subtree in inorder.

**PREORDER TRAVERSAL:**

**Step 1:** Visit the root node.

**Step 2:** Traverse the left subtree in preorder.

**Step 3:** Traverse the right subtree in preorder.

**POSTORDER TRAVERSAL:**

**Step 1:** Traverse the left subtree in postorder.

**Step 2:** Traverse the right subtree in postorder.

**Step 3:** Visit the root node.

**PROGRAM:**

#include<stdio.h>

#include<stdlib.h>

struct node {

struct node \*left;

int element;

struct node \*right;

};

typedef struct node Node;

Node \*Insert(Node \*Tree, int e);

void Inorder(Node \*Tree);

void Preorder(Node \*Tree);

void Postorder(Node \*Tree);

int main() {

Node \*Tree = NULL;

int n, i, e, ch;

printf("Enter number of nodes in the tree : ");

scanf("%d", &n);

printf("Enter the elements :\n");

for (i = 1; i <= n; i++) {

scanf("%d", &e);

Tree = Insert(Tree, e);

}

do {

printf("1. Inorder \n2. Preorder \n3. Postorder \n4. Exit\n");

printf("Enter your choice : ");

scanf("%d", &ch);

switch (ch) {

case 1:

Inorder(Tree);

printf("\n");

break;

case 2:

Preorder(Tree);

printf("\n");

break;

case 3:

Postorder(Tree);

printf("\n");

break;

}

} while (ch <= 3);

return 0;

}

Node \*Insert(Node \*Tree, int e) {

Node \*NewNode = malloc(sizeof(Node));

if (Tree == NULL) {

NewNode->element = e;

NewNode->left = NULL;

NewNode->right = NULL;

Tree = NewNode;

}

else if (e < Tree->element) {

Tree->left = Insert(Tree->left, e);

}

else if (e > Tree->element) {

Tree->right = Insert(Tree->right, e);

}

return Tree;

}

void Inorder(Node \*Tree) {

if (Tree != NULL) {

Inorder(Tree->left);

printf("%d\t", Tree->element);

Inorder(Tree->right);

}

}

void Preorder(Node \*Tree) {

if (Tree != NULL) {

printf("%d\t", Tree->element);

Preorder(Tree->left);

Preorder(Tree->right);

}

}

void Postorder(Node \*Tree) {

if (Tree != NULL) {

Postorder(Tree->left);

Postorder(Tree->right);

printf("%d\t", Tree->element);

}

}

**OUTPUT:**

Enter number of nodes in the tree : 9

Enter the elements :

43 10 79 90 12 54 11 9 50

1. Inorder

2. Preorder

3. Postorder

4. Exit

Enter your choice : 1

9 10 11 12 43 50 54 79 90

1. Inorder

2. Preorder

3. Postorder

4. Exit

Enter your choice : 2

43 10 9 12 11 79 54 50 90

1. Inorder

2. Preorder

3. Postorder

4. Exit

Enter your choice : 3

9 11 12 10 50 54 90 79 43

1. Inorder

2. Preorder

3. Postorder

4. Exit

Enter your choice : 4

**RESULT:**

Hence the program to perform Tree traversal techniques is implemented and performed successfully.