AIM

Design and implement binary tree and demonstrate its working.

ALGORTHIM

For in order

Step 1:repeat step 2 to 4 while root not equal to NULL

Step 2:in order(root->left)

Step 3:write->data

Step 4:write root->data

Step 5:end

For post order

Step 1:repet strp 2 to 4 while root not equal to NULL

Step 2:post order(root->left)

Step 3:post order(root->right)

Step 4:write root->data

Step 5:end

For pre order

Step 1:repeat step 2 to 4 while root not equal to NULL

Step 2:write root->data

Step 3:pre order(root->left)

Step 4:pre order(root->right)

Step 4:end

Program:

#include<stdio.h>

#include<stdlib.h>

struct node

{

int element;

struct node \*left;

struct node \*right;

};

struct node \*createNode(int val)

{

struct node \*Node=(struct node \*)malloc(sizeof(struct node));

Node->element=val;

Node->left=NULL;

Node->right=NULL;

return(Node);

}

void traversepreorder(struct node \*root)

{

if(root==NULL)

return;

traverseinorder(root->left);

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

traverseinorder(root->right);

}

int main()

{

struct node \*root=createNode(36);

root->left=createNode(26);

root->right=createNode(46);

root->left->left=createNode(21);

root->left->right=createNode(31);

root->left->left->left=createNode(11);

root->left->left->right=createNode(24);

root->right->left=createNode(41);

root->right->right=createNode(56);

root->right->right->left=createNode(51);

root->right->right->right=createNode(66);

printf("Preorder Traversal\n");

traversepreorder(root);

printf("\nPostorder Traversal\n");

traversepostorder(root);

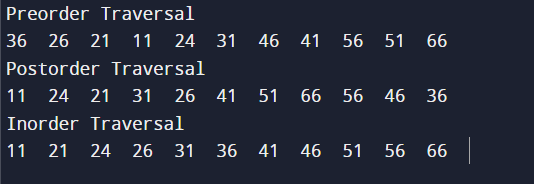
printf("\nInorder Traversal\n");

traverseinorder(root);

return 0;

}

Output:

****

Link of the github: