**Code:-**

**Stack.h Header File:-**

#include<stdio.h>

#include<stdlib.h>

typedef struct tree{

char val;

struct tree \*lchild;

struct tree \*rchild;

}tree;

typedef struct node

{

tree \*data;

struct node \*next;

}stack;

stack\* push(stack \*top,tree \*data)

{

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

NewNode->data=data;

NewNode->next=NULL;

if(top==NULL)

{

top=NewNode;

}

else

{

NewNode->next=top;

top=NewNode;

}

return top;

}

stack\* pop(stack \*top)

{

stack \*temp=top;

if(top==NULL)

{

printf("\nStack is Empty");

}

else

{

top=top->next;

free(temp);

}

return top;

}

tree\* ReadTop(stack \*top)

{

return top->data;

}

**Expression.c File:-**

#include <stdio.h>

#include "stack.h"

#include <stdlib.h>

#include<string.h>

tree\* postfix\_ex(tree \*root){

stack \*top=NULL;

char str[30];

printf("Enter the Postfix Expression : ");

scanf("%s",&str);

for(int i=0;str[i]!='\0';i++){

tree \*tree\_node=(tree\*)malloc(sizeof(tree));

tree\_node->val=str[i];

tree\_node->lchild=tree\_node->rchild=NULL;

if(isalpha(str[i])){

top=push(top,tree\_node);

}

else{

tree\_node->rchild=ReadTop(top);

top=pop(top);

tree\_node->lchild=ReadTop(top);

top=pop(top);

top=push(top,tree\_node);

}

}

root=ReadTop(top);

top=pop(top);

return root;

}

tree\* prefix\_ex(tree \*root){

stack \*top=NULL;

char str[30];

printf("Enter the Prefix Expression : ");

scanf("%s",&str);

int length=strlen(str);

for(int i=length-1;i>=0;i--){

tree \*tree\_node=(tree\*)malloc(sizeof(tree));

tree\_node->val=str[i];

tree\_node->lchild=tree\_node->rchild=NULL;

if(isalpha(str[i])){

top=push(top,tree\_node);

}

else{

tree\_node->lchild=ReadTop(top);

top=pop(top);

tree\_node->rchild=ReadTop(top);

top=pop(top);

top=push(top,tree\_node);

}

}

root=ReadTop(top);

top=pop(top);

return root;

}

void pre\_order(tree \*root){

if(root!=NULL){

printf("%c",root->val);

pre\_order(root->lchild);

pre\_order(root->rchild);

}

}

void post\_order(tree \*root){

if(root!=NULL){

post\_order(root->lchild);

post\_order(root->rchild);

printf("%c",root->val);

}

}

void in\_order(tree \*root){

if(root!=NULL){

in\_order(root->lchild);

printf("%c",root->val);

in\_order(root->rchild);

}

}

void main(){

int ch;

tree \*root=NULL;

printf("Enter Prefix or Post Expression '1/2'");

scanf("%d",&ch);

if(ch==1){

root=prefix\_ex(root);

}

else{

root=postfix\_ex(root);

}

printf("\nPre Order: ");

pre\_order(root);

printf("\nPost Order: ");

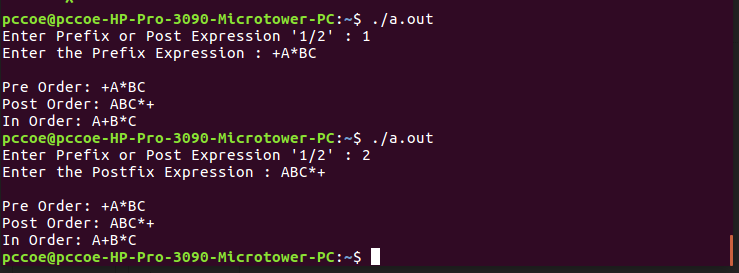
post\_order(root);

printf("\nIn Order: ");

in\_order(root);

}

**Output:-**

****