```
//By Ritwik Chandra Pandey
//On 4th Sep 2021
//Creation of Exp Trees

#include<stdio.h>
#include<malloc.h>
struct tree {
	char data;
	struct tree *left;
	struct tree *right;
};
typedef struct tree * ENODE;
ENODE stack[30];
int top = -1;
ENODE newnode(char ch) {
	ENODE temp;
	temp = (ENODE)malloc(sizeof(struct tree));
	temp->data = ch;
	temp->left = NULL;
	temp->right = NULL;
	return(temp);
}
void push(ENODE temp) {
	stack[++top]=temp;
}
ENODE pop() {
	ENODE p;
	p=stack[top--];
	return(p);
}
void inorder(ENODE t) {
	if(t!=NULL) {
		inorder(t->left);
		printf("%c",t->data);
		inorder(t->right);
	}
}
void preorder(ENODE t) {
	if(t!=NULL) {
		printf("%c",t->data);
```

```
        preorder(t->left);
        inorder(t->right);
    }
}
void postorder(ENODE t) {
    if(t!=NULL) {
        postorder(t->left);
        postorder(t->right);
        printf("%c",t->data);
    }
}
void main() {
    char postfix_exp[20];
    ENODE temp;
    int j,i;
    printf("Enter a postfix expression : ");
    scanf("%s",postfix_exp);
    for(i=0;postfix_exp[i]!='\0';i++){
        if(postfix_exp[i]=='+' || postfix_exp[i]=='-' || postfix_exp[i]=='*' || postfix_exp[i]=='/'){
            temp = newnode(postfix_exp[i]);
            temp->right = pop();
         temp->left = pop();
         push(temp);
            }else{
                temp = newnode(postfix_exp[i]);
                push(temp);
            }
    }


    printf("Inorder Traversal of expression tree : ");
    inorder(temp);
    printf("\n");
    printf("Preorder Traversal of expression tree : ");
    preorder(temp);
    printf("\n");
    printf("Postorder Traversal of expression tree : ");
    postorder(temp);
    printf("\n");
}
```