```
/// Author: Prathamesh Patil - Exp 1: Expression Tree
#include<iostream>
#include<ctype.h>
using namespace std;
class TreeNode{
public:
char data;
TreeNode *left;
TreeNode *right;
TreeNode(char ch){
data = ch;
left = NULL;
right = NULL;
}
};
class Node{
public:
TreeNode *data;
Node *prev;
};
class Stack{
private:
Node *top;
public:
Stack(){
top = NULL;
}
void push(TreeNode *treeNode){
Node *newNode = new Node;
newNode->data = treeNode;
newNode->prev = top;
top = newNode;

}
TreeNode* pop(){
if(top == NULL){
cout<<"\nSTACK UNDERFLOW !";
return NULL;
}
Node *temp = top;
TreeNode* data = temp->data;
top = top->prev;
delete temp;
return data;
}
bool isEmpty(){
if(top == NULL){
return true;
}
return false;
}
TreeNode* peek(){
return top->data;
}
~Stack(){
Node *temp;
temp = top;
while(temp!=NULL){
delete temp;
temp = temp->prev;
}
delete top;
}
};
void inorderTraversal(TreeNode* root){
Stack stk;
TreeNode *current = root;
while(current!=NULL || !stk.isEmpty()){
while(current!=NULL){
stk.push(current);
current = current->left;
}

current = stk.pop();
cout<<current->data<< " ";
current=current->right;
}
}
int main(){
string postfix;
Stack stk;
cout<<"Enter the postfix expression:"<<endl;
cin>>postfix;
for(int i =0;i<postfix.length();i++){
TreeNode *newNode = new TreeNode(postfix[i]);
if(isalpha(postfix[i])){
stk.push(newNode);
}
else{
newNode->right = stk.pop();
newNode->left = stk.pop();
stk.push(newNode);
}
}
cout<<"The Expression Tree is:"<<endl;
inorderTraversal(stk.pop());
return 0;}
```

OUTPUT:

```
/tmp/3QEQnOXXmt.o
Enter the postfix expression:
ab+c-
The Expression Tree is:
a + b - c
```

```
/tmp/3QEQnOXXmt.o
Enter the postfix expression:
ab-c*d+
The Expression Tree is:
a - b * c + d s
```