#include<iostream>

#include<stack>

using namespace std;

struct Node

{

char value;

Node\* left;

Node\* right;

};

Node\* create(char value){

Node\* newNode=new Node;

newNode->value=value;

newNode->left=newNode->right=NULL;

return newNode;

}

Node\* createTree(const string& prefix){

stack<Node\*>st;

int length=prefix.length();

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

char c=prefix[i];

if(isalnum(c)){

Node\* node=create(c);

st.push(node);

}

else{

Node\* node=create(c);

node->left=st.top();

st.pop();

node->right=st.top();

st.pop();

}

}

return st.top();

}

void postorderTraversal(Node\* root){

if(root==NULL)

return;

stack<Node\*>st;

st.push(root);

stack<char>result;

while(!st.empty()){

Node\* curr=st.top();

st.pop();

result.push(curr->value);

if(curr->left)

st.push(curr->left);

if(curr->right)

st.push(curr->right);

}

if(!result.empty()){

cout<<result.top();

result.pop();

}

}

void deleteTree(Node\* root){

if(root==NULL)

return;

deleteTree(root->left);

deleteTree(root->right);

delete(root);

}

int main(){

string prefix="+--a\*bc/def";

Node\* root=createTree(prefix);

cout<<"Postorder traversal"<<"\n";

postorderTraversal(root);

cout<<endl;

deleteTree(root);

}