#include<iostream>

#include<stack>

#include <vector>

#include <cctype>

using namespace std;

struct node{

public :

char value;

node\* left;

node\* right;

node(char val) {

value = val;

left = NULL;

right = NULL;

}

};

class expressionTree{

private:

node\* root;

public:

expressionTree(){

root = NULL;

}

node\* constructTree(string expression){

stack<node\*> stack;

for (int i = expression.size()-1 ; i >=0 ; i--){

char c = expression[i];

if(isalnum(c)){ //abcd123 operand hoga koi

node\* node1 = new node(c);

stack.push(node1);

}

else{

node\* node1 = new node(c);

node1 -> left = stack.top();

stack.pop();

node1->right = stack.top();

stack.pop();

stack.push(node1);

}

}

return stack.top();

}

void postOrderTraversal(node\* root){

if(root == NULL)

return ;

postOrderTraversal(root -> left);

postOrderTraversal(root -> right);

cout << root -> value;

return;

}

void deleteTree(node\* node1){

if (!node1)

return ;

deleteTree(root->left);

deleteTree(root->right);

delete node1;

}

~expressionTree(){

deleteTree(root);

}

};

int main(){

expressionTree tree;

string expression;

cout << "Enter the prefix expression :: ";

cin >> expression ;

node\* root = tree.constructTree(expression);

cout << "POst order traversal :: ";

tree.postOrderTraversal(root);

}