#include<iostream>

#include<stack>

#include<string>

#include<cstring>

#include<algorithm>

using namespace std;

class tree\_node

{

public:

char var;

tree\_node\* left, \* right;

};

class exp\_tree

{

char pre[30];

tree\_node\* root;

public:

exp\_tree()

{

root == NULL;

}

void get\_exp()

{

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

cin >> pre;

}

tree\_node\* get\_root()

{

return root;

}

void exp\_to\_tree();

void infix(tree\_node\*);

void postfix(tree\_node\*);

};

void exp\_tree::exp\_to\_tree()

{

int top = -1, z;

tree\_node\* stack[20];

\_strrev(pre);

for (int i = 0; pre[i] != 0; i++)

{

stack[++top] = new tree\_node;

if (pre[i] >= 97 && pre[i] <= 122)

{

stack[top]->var = pre[i];

stack[top]->left = stack[top]->right = NULL;

}

else

{

stack[top]->var = pre[i];

stack[top]->left = stack[top - 1];

stack[top]->right = stack[top - 2];

top = top - 2;

root = stack[top] = stack[top + 2];

}

}

}

void exp\_tree::infix(tree\_node\* temp)

{

if (temp == NULL)

{

return;

}

infix(temp->left);

cout << temp->var;

infix(temp->right);

}

void exp\_tree::postfix(tree\_node\* temp)

{

if (temp == NULL)

return;

postfix(temp->left);

postfix(temp->right);

cout << temp->var;

}

int main()

{

exp\_tree exp;

exp.get\_exp();

exp.exp\_to\_tree();

cout << "Expression tree Successfully created" << endl;

cout << "Infix Expression is : " << endl;

exp.infix(exp.get\_root());

cout << endl << "Postfix Expression is : ";

exp.postfix(exp.get\_root());

return 0;

}