/\* C++ implementation to convert

infix expression to postfix\*/

#include<bits/stdc++.h>

using namespace std;

//Function to return precedence of operators

int prec(char c) {

if(c == '^')

return 3;

else if(c == '/' || c=='\*')

return 2;

else if(c == '+' || c == '-')

return 1;

else

return -1;

}

// The main function to convert infix expression

//to postfix expression

void infixToPostfix(string s) {

stack<char> st; //For stack operations, we are using C++ built in stack

string result;

for(int i = 0; i < s.length(); i++) {

char c = s[i];

// If the scanned character is

// an operand, add it to output string.

if((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') || (c >= '0' && c <= '9'))

result += c;

// If the scanned character is an

// ‘(‘, push it to the stack.

else if(c == '(')

st.push('(');

// If the scanned character is an ‘)’,

// pop and to output string from the stack

// until an ‘(‘ is encountered.

else if(c == ')') {

while(st.top() != '(')

{

result += st.top();

st.pop();

}

st.pop();

}

//If an operator is scanned

else {

while(!st.empty() && prec(s[i]) <= prec(st.top())) {

result += st.top();

st.pop();

}

st.push(c);

}

}

// Pop all the remaining elements from the stack

while(!st.empty()) {

result += st.top();

st.pop();

}

cout << result << endl;

}

//Driver program to test above functions

int main() {

string exp = "(a+b+c)(a^2+b^2+c^2-ab-bc-ca)";

infixToPostfix(exp);

return 0;

}