Assignment . 7

#include <iostream>

#include <stack>

#include <cctype>

#include <string>

using namespace std;

int precedence(char op) {

if (op == '+' || op == '-') return 1;

if (op == '\*' || op == '/') return 2;

return 0;

}

bool isOperator(char c) {

return (c == '+' || c == '-' || c == '\*' || c == '/');

}

string infixToPostfix(string infix) {

stack<char> s;

string postfix;

for (char c : infix) {

if (isalnum(c)) {

postfix += c;

} else if (c == '(') {

s.push(c);

} else if (c == ')') {

while (!s.empty() && s.top() != '(') {

postfix += s.top();

s.pop();

}

s.pop();

} else if (isOperator(c)) {

while (!s.empty() && precedence(s.top()) >= precedence(c)) {

postfix += s.top();

s.pop();

}

s.push(c);

}

}

while (!s.empty()) {

postfix += s.top();

s.pop();

}

return postfix;

}

int evaluatePostfix(string postfix) {

stack<int> s;

for (char c : postfix) {

if (isdigit(c)) {

s.push(c - '0');

} else if (isOperator(c)) {

int b = s.top(); s.pop();

int a = s.top(); s.pop();

switch (c) {

case '+': s.push(a + b); break;

case '-': s.push(a - b); break;

case '\*': s.push(a \* b); break;

case '/': s.push(a / b); break;

}

}

}

return s.top();

}

int main() {

string infix;

cout << "Enter infix expression: ";

cin >> infix;

string postfix = infixToPostfix(infix);

cout << "Postfix expression: " << postfix << endl;

int result = evaluatePostfix(postfix);

cout << "Evaluation result: " << result << endl;

return 0;

}

Enter infix expression: 2\*(3+4)-5

Postfix expression: 234+\*5-

Evaluation result: 9