**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Алгоритмы и структуры данных»**

Тема: **Стеки и очереди**

|  |  |  |
| --- | --- | --- |
| Студент гр. 7381 |  | Судакова П.С. |
| Преподаватель |  | Фирсов М.А. |

Санкт-Петербург

2018

**Задание.**

Вариант 11а-в

Рассматривается выражение следующего вида:

< выражение >::= < терм > | < терм > + < выражение > |

< терм > - < выражение >

< терм > ::= < множитель > | < множитель > \* < терм >

< множитель > ::= < число > | < переменная > | ( < выражение > ) |

< множитель > ^ < число >

< число > ::= < цифра >

< переменная > ::= < буква >

Такая форма записи выражения называется *инфиксной.*

Постфиксной (префиксной) формой записи выражения *aDb* называет-ся запись, в которой знак операции размещен за (перед) операндами: *abD (Dab).*

Примеры

***Инфиксная Постфиксная Префиксная***

a-b ab- -ab

a\*b+c ab\*c+ +\*abc

a\*(b+c) abc+\* \*a+bc

a+b^c^d\*e abc^d^e\*+ +a\*^b^cde.

Отметим, что постфиксная и префиксная формы записи выражений не со-держат скобок.

**Требуется:**

Вычислить как целое число значение выражения (без переменных), записанного в постфиксной форме в заданном текстовом файле *postfix*;

**Описание программы.**

Выражение просматривается слева направо. Если встречается операнд (число), то его значение (целое) заносится в стек, а если встречается знак операции , то из стека исключаются два последних элемента (это операнды данной операции) над ними выполняется операция и ее результат записывается в стек. В конце концов в стеке остается только одно число - значение всего выражения.

**Тестирование.**

Для проверки работоспособности программы был создан скрипт (см. ПРИЛОЖЕНИЕ В) для автоматического ввода и вывода тестовых данных. Результаты тестирования сохраняются в файл.

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
| (8+4/2)/(10/2-3\*2+6) -2 +9 | 8 4 2 / + 10 2 / 3 2 \* - 6 + / 2- 9 |
| (5\*6 +2)/(3-4 /9) \*9 | 112 |
| 2+(4 \*12+6/8\*3 )/5 | 12 |
| (4 \*8+8\*3 +1 )/5+ 9 | 20 |
| (5+7 +8\*4/8- 5)+7 | 18 |
| 2\*5 | 10 |

**Выводы.**

В процессе выполнения работы были изучены навыки по реализации такой структуры данных, как стек, были изучены различные способы записи математических выражений и методы перевода выражений из инфиксной в постфиксную нотацию.

Приложение А

Код программы

#include <cctype>

#include <map>

#include <sstream>

#include <string.h>

#include <iostream>

#include <stdlib.h>

#include <queue>

#include <string>

#include <cstring>

#include <stdio.h>

#include <cmath>

#include <stack>

#include <stdio.h>

using namespace std;

int summa(string s)

{

string srpn;

string::size\_type ind;

while ((ind = s.find(' ')) != string::npos)

{

s.erase(ind, 1);

}

map<char, size\_t> map;

map.insert(make\_pair('\*', 3));

map.insert(make\_pair('/', 3));

map.insert(make\_pair('+', 2));

map.insert(make\_pair('-', 2));

map.insert(make\_pair('(', 1));

stack<char> stack;

for (auto c : s)

{

if (!isdigit(c)) {

srpn += ' ';

if (')' == c) {

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

{

srpn += stack.top();

stack.pop();

srpn += ' ';

std::cout << srpn << std::endl;

}

stack.pop();

}

else if ('(' == c) {

stack.push(c);

}

else if (stack.empty() || (map[stack.top()] < map[c])) {

stack.push(c);

}

else

{

do

{

srpn += stack.top();

srpn += ' ';

stack.pop();

std::cout << srpn << std::endl;

} while (!(stack.empty() || (map[stack.top()] < map[c])));

stack.push(c);

}

}

else {

srpn += c;

std::cout << srpn << std::endl;

}

}

while (!stack.empty())

{

srpn += stack.top();

srpn += ' ';

stack.pop();

std::cout << srpn << std::endl;

}

std:: stack<double> dstack;

stringstream ss(srpn);

double d, d1;

char c;

while (ss.get(c))

{

if (isdigit(c))

{

ss.unget();

ss >> d;

dstack.push(d);

}

else if (!isspace(c))

{

d1 = dstack.top();

dstack.pop();

d = dstack.top();

dstack.pop();

switch (c)

{

case '+':

dstack.push(d + d1);

break;

case '-':

dstack.push(d - d1);

break;

case '\*':

dstack.push(d \* d1);

break;

case '/':

dstack.push(d / d1);

break;

}

}

}

int k;

if (!dstack.empty())

{

k = dstack.top();

dstack.pop();

}

return k;

}

int main() {

std::string buffer;

std::getline(std::cin, buffer);

std::cout << summa(buffer);

return 0;

}

Приложение B

Скрипт

g++ ./Source/main.cpp -std=c++11 -o Lab3

echo -e '\_\_\_\_\_\_\_\nTest 1:'

cat ./Tests/Test1.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab3 < ./Tests/Test1.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 2:'

cat ./Tests/Test2.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab3 < ./Tests/Test2.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 3:'

cat ./Tests/Test3.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab3 < ./Tests/Test3.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 4:'

cat ./Tests/Test4.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab3 < ./Tests/Test4.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 5:'

cat ./Tests/Test5.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab3 < ./Tests/Test5.txt

echo -e ''

echo -e '\_\_\_\_\_\_\_\nTest 6:'

cat ./Tests/Test6.txt

echo -e '\_\_\_\_\_\_\_\nTesting:\n'

./Lab3 < ./Tests/Test6.txt