Белорусский государственный университет информатики и радиоэлектроники

Факультет компьютерных технологий

ОТЧЕТ

по лабораторной работе №5

на тему: **Обратная польская запись**

|  |  |
| --- | --- |
| Выполнил  студент  группы №980161  Алейчик И.Д. | Проверил  доцент  Казакевич Ю.Д. |

Минск 2020

**Лабораторная работа №1**

**Вариант 1**

**Тема:**

Обратная польская запись

**Цель работы:**

Изучить правила формирования постфиксной записи ариф-метических выраженийс использованием стека.

**Краткие теоретические сведения:**

Одной из задач при разработке трансляторов является задача расшифров-ки арифметических выражений.Выражениеa+bзаписано винфикснойформе,+ab–впрефиксной, ab+–в постфиксной. В наиболее распространенной инфиксной форме для указания последовательности выполнения операций необходимо расставлять скобки. Польский математик Я. Лукашевич использовал тот факт, что при записи пост-фиксной формы скобки не нужны, а последовательность операндов и операций удобна для расшифровки. Поэтому постфиксная запись выражений получила название обратной польской записи (ОПЗ).

Рассмотрим алгоритм получения ОПЗ из исходной строки символов, в ко-торой записано выражение в инфиксной форме.

1. Символы-операнды переписываются в выходную строку, в которой формируется постфиксная форма выражения.

2. Открывающая скобка записывается в стек.

3. Очередная операция выталкивает в выходную строку все операции из стека с большим или равным приоритетом.

4. Закрывающая скобка выталкивает все операции из стека до ближайшей открывающей скобки в выходную строку, открывающая скобка удаляется из стека, а закрывающая –игнорируется.

5. Если после просмотра последнего символа исходной строки в стеке остались операции, то все они выталкиваются в выходную строку

**Задание:**



**Ход выполнения работы:**

int priority(char sy) {

switch (sy) {

case '\*':

case '/':

return 3;

break;

case '+':

case '-':

return 2;

break;

case '(':

return 1;

break;

case ')':

return -1;

break;

default:

return 0;

break;

}

}

string converToReversePolishNat(string str) {

string obl = "";

stack<char> SumbolStack;

int preorotet;

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

preorotet = priority(str[i]);

if (preorotet == 0)obl += str[i];

if (preorotet == 1)SumbolStack.push(str[i]);

if (preorotet > 1) {

while (!SumbolStack.empty()) {

if (priority(SumbolStack.top()) >= preorotet) {

obl += SumbolStack.top();

SumbolStack.pop();

}

else break;

}

SumbolStack.push(str[i]);

}

if (preorotet == -1) {

while (priority(SumbolStack.top()) != 1) {

obl += SumbolStack.top();

SumbolStack.pop();

}

SumbolStack.pop();

}

}

while (!SumbolStack.empty()) {

obl += SumbolStack.top();

SumbolStack.pop();

}

return obl;

}

double createAnswer(string obl) {

stack<double> valueStack;

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

if ((int)obl[i] >= 97 && (int)obl[i] <= 122) {

double localValue;

cout << "Input value of: " << obl[i] << endl;

cin >> localValue;

valueStack.push(localValue);

}

else {

double a = 0, b = 0;

b = valueStack.top();

valueStack.pop();

a = valueStack.top();

valueStack.pop();

switch (obl[i]) {

case'\*':

valueStack.push(a \* b);

break;

case'/':

valueStack.push(a / b);

break;

case'+':

valueStack.push(a + b);

break;

case'-':

valueStack.push(a - b);

break;

default:

break;

}

}

}

return valueStack.top();

}

void polskaRecords() {

string s = "(a+(b/c-d))\*e";

cout << "Input equation: ";// << endl;

cin >> s;

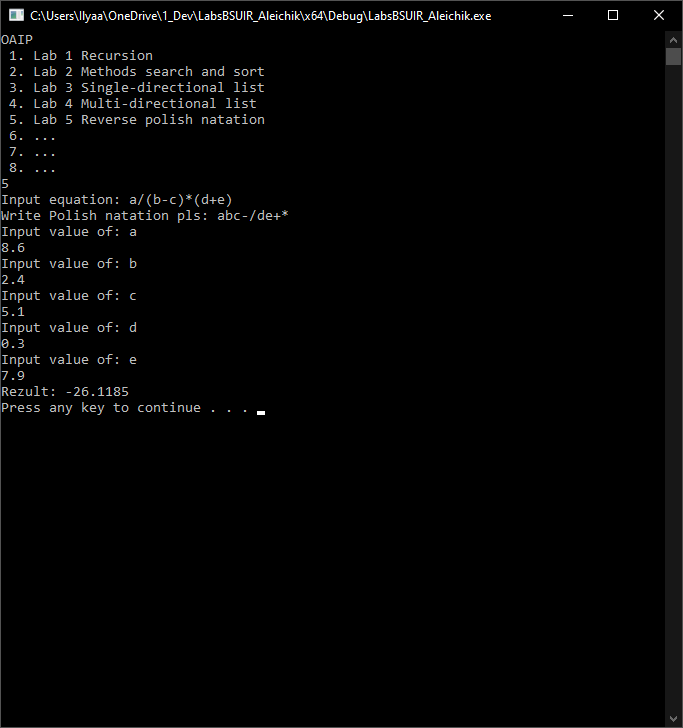
cout << "Write Polish natation pls: " << converToReversePolishNat(s) << endl;

cout << "Rezult: " << createAnswer(converToReversePolishNat(s)) << endl;

system("pause");

}

**Результат работы программы:**



**Вывод:**

Изучены правила формирования постфиксной записи ариф-метических выраженийс использованием стека.