Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение  
высшего образования «Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского»

Институт Информационных технологий, математики и механики

Отчет по лабораторной работе

**Вычисление арифметических выражений**

Выполнил:

студент гр. 381806 – 1

Пронькин Д.А.

Проверил:

ассистент каф. МОСТ, ИИТММ

Кустикова В.Д.

Нижний Новгород

2019 г.

Содержание

[Введение 3](#_Toc25922248)

[Постановка задачи 4](#_Toc25922249)

[Руководство пользователя 5](#_Toc25922250)

[Руководство программиста 7](#_Toc25922251)

[Описание структуры программы 7](#_Toc25922252)

[Описание структур данных 7](#_Toc25922253)

[Описание алгоритмов 9](#_Toc25922254)

[Алгоритмы структуры Stack 9](#_Toc25922255)

[Перевод в польскую нотацию 9](#_Toc25922256)

[Вычисление выражения записанного в польской нотации 10](#_Toc25922257)

[Заключение 12](#_Toc25922258)

[Литература 13](#_Toc25922259)

# Введение

Целью лабораторной работы является изучение и практическое применение структуры Стек. Темой данной работы является разработка программы, позволяющей вычислять значения произвольного арифметического выражения при помощи структуры Стек

# Постановка задачи

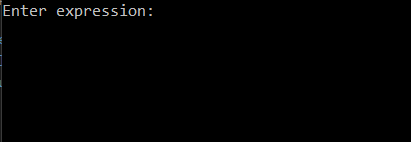
Задача: разработать программу, переводящую вводимое пользователем с клавиатуры арифметическое выражение, в польскую нотацию и вычисляющую его значение.

Входные данные: строка, содержащее арифметическое выражение, значение переменных содержащихся в выражение.

Выходные данные: строка, содержащая исходной выражение в польской нотации, значение выражения.

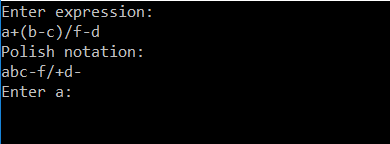
# Руководство пользователя

После запуска программы откроется консольный интерфейс с предложением ввести арифметическое выражение.



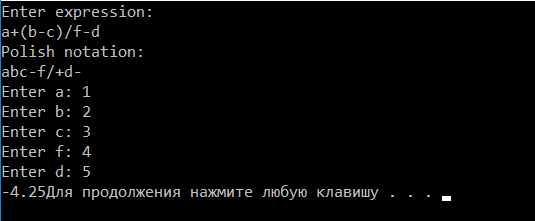
**Рис. 1 Стартовый экран**

После введения выражения нажмите клавишу Enter. Если ваше выражение корректно, то программа выведет польскую нотацию и предложить указать значение введенных переменных.



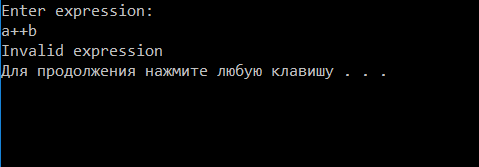
**Рис. 2 Программы вывела польскую нотацию и ожидает ввода переменной a**

Как только вы введете значения всех переменных, программа вычислит выражение и выведет результат

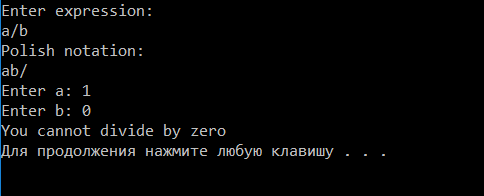


**Рис. 3Выведен результат**

Если вы введете некорректное выражение или значение переменной при которой выражение не вычисляется, то программа выведет сообщение об ошибке и завершит работу.



**Рис. 4Введено неверное выражение**



**Рис. 5Попытка деления на ноль**

# Руководство программиста

## Описание структуры программы

Stack.h – класс, реализующий функционал структуры данных Стек.

Converter.h, Converter.cpp–модуль со статическим классом, содержащий методы для работы с арифметическими выражениями.

## Описание структур данных

***Класс Stack***

Stack – класс, реализующий функционал структуры данных стек.

Объявление:

template<class ValueType>

classTStack {

public:

ValueType\* elems;

int size;

int top;

public:

TStack(int size = 4);

TStack(constTStack&);

~TStack();

void Push(ValueType);

ValueTypePop();

boolIsEmpty() const;

boolIsFull() const;

ValueTypeTopElems() const;

};

**Описание методов класса**

TStack(int size = 4)

Назначение: конструктор, инициализация объекта

Входные параметры: размер стека

Возвращаемое значение: нет

TStack(constTStack&)

Назначение: конструктор копирования, создание копии текущего стека

Входные параметры: константная ссылка на копируемый объект

Возвращаемое значение: нет

~TStack()

Назначение: деструктор, высвобождение памяти и удаление объекта класс

Входные параметры: нет

Возвращаемое значение: нет

voidPush(ValueType)

Назначение: добавление значения в стек

Входные параметры: добавляемое значение

Возвращаемое значение: нет

ValueTypePop()

Назначение: извлечение элемента из стека

Входные параметры: нет

Возвращаемое значение: извлекаемое значение

boolIsEmpty() const

Назначение: проверка стека на пустоту

Входные параметры: нет

Возвращаемое значение: логическая переменная

boolIsFull() const

Назначение: проверка стека на полноту

Входные параметры: нет

Возвращаемое значение: логическая переменная

ValueTypeTopElems() const

Назначение: просмотр значения на вершине стека

Входные параметры: нет

Возвращаемое значение: значение на вершине стека

***Класс Converter***

Converter – класс, реализующий методы для работы с арифметическими выражениями

**Объявление**

static class Converter {

private:

staticbool Priority(char, char);

static double Calculator(double, double, char);

public:

static string CreatePostFixForm(const string&);

static double Calculate(const string&, double\*, int);

static double\* GetValueOperands(const string&, int\*);

};

**Описание методов класса**

static string CreatePostFixForm(const string&)

Назначение: создание польской нотации

Входные параметры: исходное выражение

Возвращаемое значение: преобразованное в польскую нотацию исходное выражение

static double Calculate(const string&, double\*, int)

Назначение: вычисление выражения при помощи польской нотации

Входные параметры: исходное выражение, записанное в польской нотации, массив значений переменных, количество переменных

Возвращаемое значение: результат вычисления выражения

static double\* GetValueOperands(const string&, int\*)

Назначение:получение значений переменных

Входные параметры: исходное выражение, записанное в польской нотации, ссылка на переменную которая содержит количество переменных

Возвращаемое значение: массив значений переменных

staticbool Priority(char, char)

Назначение: сравнение приоритета пришедшей операции и операции на вершине стека

Входные параметры: 2 значения операций

Возвращаемое значение: логическая переменная

staticdoubleCalculator(double, double, char)

Назначение: калькулятор

Входные параметры: значения переменных и арифметическое действие в типе char

Возвращаемое значение: результат вычисления

## Описание алгоритмов

### Алгоритмы структуры Stack

Стек ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) stack — стопка; читается стэк) — [абстрактный тип данных](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), представляющий собой [список элементов](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)), организованных по принципу [LIFO](https://ru.wikipedia.org/wiki/LIFO) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) lastin — firstout, «последним пришёл — первым вышел»).

Характеристики стека:

1. Массив данных
2. Индекс последнего занятого
3. Размер стека

Методы стека:

1. **Добавление элемента в стек:**если стек не полный, то увеличить значение индекса последнего не занятого на единицу и положить по этому индексу элемент в массив данных
2. **Получение данных на вершине стека:** если стек не пустой, то возвращается значение по индексу последнего занятого
3. **Удаление данных:** если стек не пустой, то возвращается значение по индексу последнего занятого и индекс последнего занятого уменьшается на единицу
4. **Проверка на пустоту:** если значение последнего занятого равно -1, то стек пуст
5. **Проверка на полноту:** если значение последнего занятого равно размеру стека – 1, то стек полон

### Перевод в польскую нотацию

На вход приходит строка содержащая выражение, на выходе получается строка содержащая польскую нотацию

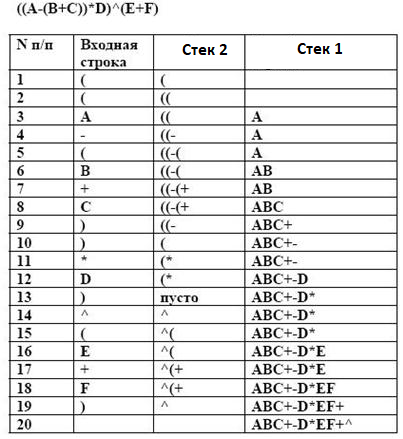
Перед выполнением алгоритма создаем два стека, в (1) стек будем записывать операнды, во (2) стек будем записывать операции

**Алгоритм:**

1. Если пришел операнд, то записываем в стек (1)
2. Если пришла левая скобка, то записываем в стек (2)
3. Если пришла правая скобка, то перезаписываем все значения из стека (2) в стек (1), пока не встретится левая скобка, которую тоже удаляем
4. Если пришла операция, то сравниваем ее приоритет с приоритетом операции на вершине стека
   1. Если пришла операция с более высоким приоритетом чем на вершине стека, то записываем ее в стек (2)
   2. Если приоритет пришедшей операции ниже или равен приоритету операции на вершине стека, то перезаписываем все операции с большим или равным приоритетом из стека (2) в стек (1), пока не встретим левую скобку или не дойдем до конца стека или не встретим операцию с меньшим приоритетом

После того как дошли до конца строки перекладываем все значения из стека (2) в стек (1), а затем при помощи конкатенации строк формируем польскую нотацию.

Пример:

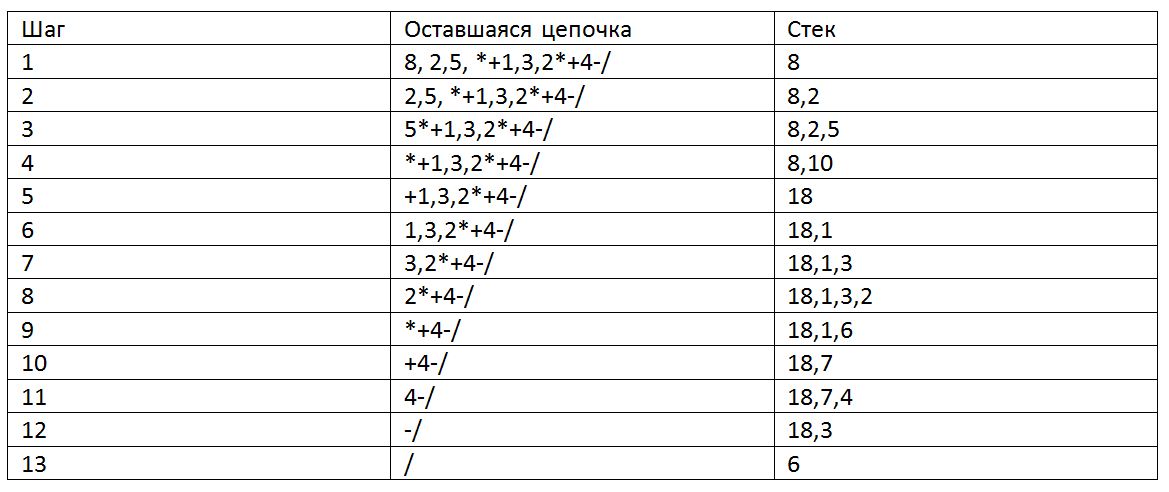


**Рис. 6Пример построения польской нотации**

### Вычисление выражения записанного в польской нотации

Обратная польская запись идеально подходит для вычисления формул на компьютере со стеком. Формула состоит из n символов, каждый из которых является либо операндом, либо оператором. Алгоритм для вычисления формулы в обратной польской записи с использованием стека прост. Нужно просто прочитать обратную польскую запись слева направо. Если встречается операнд, его нужно пометить в стек. Если встречается оператор, нужно выполнить заданную им операцию.  
В качестве примера рассмотрим вычисление следующего выражения: (8+2\*5)/(1+3\*2-4). Соответствующая формула в обратной польской записи выглядит так: 825\*+132\*+4-/  
Число на вершине стека – это правый операнд (а не левый). Это очень важно для операций деления, вычитания и возведения в степень, поскольку порядок следования операндов в данном случае имеет значение (в отличие от операций сложения и умножения). Другими словами, операция деления действует следующим образом: сначала в стек помещается числитель, потом знаменатель, и тогда операция даёт правильный результат. Отметим, что преобразовать обратную польскую запись в машинный код очень легко: нужно просто двигаться по формуле в обратной польской записи, записывая по одной команде для каждого символа. Если символ является константой или переменной, нужно вписывать команду помещения этой константы или переменной в стек, если символ является оператором, нужно вписывать команду выполнения это операции.

Пример:



**Рис. 7Пример вычисления по польской нотации**

# Заключение

В ходе выполнения лабораторной работы, была изучена структура данных стек, её принципы работы и возможный операции. Рассмотрен и реализован на практике алгоритм перевода выражения в постфиксную форму с последующим вычислением значения оного.

# Литература

Статья на habr “Обратная польская запись”

[https://habr.com/ru/post/100869/]