МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

Высшего образования

**«Нижегородский государственный университет им. Н.И. Лобачевского»**

**Национальный исследовательский университет**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

**Отчет по учебной практике**

**«Преобразование арифметических выражений в обратную польскую запись»**

**Выполнил:** студент группы 381706-1

Суслов Егор Игоревич

\_\_\_\_\_\_\_\_\_\_\_ Подпись

**Научный руководитель:**

ассистент каф. МОСТ ИИТММ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лебедев И.Г

Нижний Новгород

2018.

Содержание

[1. Введение 3](#_Toc533083472)

[2. Постановка задачи 4](#_Toc533083473)

[3. Руководство пользователя 5](#_Toc533083474)

[4. Руководство программиста 6](#_Toc533083475)

[4.1. Описание структуры программы 6](#_Toc533083476)

[4.2. Описание структур данных 6](#_Toc533083477)

[4.3. Описание алгоритмов 7](#_Toc533083478)

[5. Заключение 8](#_Toc533083479)

[6. Литература 9](#_Toc533083480)

# Введение

Обратная польская нотация была разработана [австралийским](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%81%D1%82%D1%80%D0%B0%D0%BB%D0%B8%D1%8F) [философом](https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%BB%D0%BE%D1%81%D0%BE%D1%84%D0%B8%D1%8F) и специалистом в области [теории вычислительных машин](https://ru.wikipedia.org/w/index.php?title=%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D1%85_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD&action=edit&redlink=1) [Чарльзом Хэмблином](https://ru.wikipedia.org/w/index.php?title=%D0%A5%D1%8D%D0%BC%D0%B1%D0%BB%D0%B8%D0%BD,_%D0%A7%D0%B0%D1%80%D0%BB%D1%8C%D0%B7&action=edit&redlink=1) в середине [1950-х](https://ru.wikipedia.org/wiki/1950-%D0%B5) на основе [польской нотации](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%8C%D1%81%D0%BA%D0%B0%D1%8F_%D0%BD%D0%BE%D1%82%D0%B0%D1%86%D0%B8%D1%8F), которая была предложена в [1920 году](https://ru.wikipedia.org/wiki/1920_%D0%B3%D0%BE%D0%B4) польским математиком [Яном Лукасевичем](https://ru.wikipedia.org/wiki/%D0%9B%D1%83%D0%BA%D0%B0%D1%81%D0%B5%D0%B2%D0%B8%D1%87,_%D0%AF%D0%BD). Работа Хэмблина была представлена на конференции в июне [1957](https://ru.wikipedia.org/wiki/1957), и издана в 1957 и [1962](https://ru.wikipedia.org/wiki/1962).

Первыми компьютерами, поддерживающими обратную польскую нотацию, были [KDF9](https://ru.wikipedia.org/w/index.php?title=KDF9&action=edit&redlink=1) от [English Electric Company](https://ru.wikipedia.org/w/index.php?title=English_Electric_Company&action=edit&redlink=1" \o "English Electric Company (страница отсутствует)), который был анонсирован в [1960](https://ru.wikipedia.org/wiki/1960) и выпущен (появился в продаже) в [1963](https://ru.wikipedia.org/wiki/1963), и [американский](https://ru.wikipedia.org/wiki/%D0%A1%D0%A8%D0%90" \o "США)[Burroughs B5000](https://ru.wikipedia.org/w/index.php?title=Burroughs_B5000&action=edit&redlink=1), анонсирован в [1961](https://ru.wikipedia.org/wiki/1961), выпущен в том же 1963. Один из проектировщиков B5000, Р. С. Бартон, позже написал, что разработал обратную польскую запись независимо от Хэмблина, примерно в 1958, в процессе чтения книги по [символьной логике](https://ru.wikipedia.org/w/index.php?title=%D0%A1%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BB%D0%BE%D0%B3%D0%B8%D0%BA%D0%B0&action=edit&redlink=1), и до того как познакомился с работой Хэмблина.

Компания [Friden](https://ru.wikipedia.org/w/index.php?title=Friden&action=edit&redlink=1" \o "Friden (страница отсутствует)) перенесла ОПН в настольные [калькуляторы](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D0%BB%D1%8C%D0%BA%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80), выпустив в июне 1964 модель EC-130. А в 1968 инженеры [Hewlett-Packard](https://ru.wikipedia.org/wiki/Hewlett-Packard" \o "Hewlett-Packard) разработали настольный калькулятор [9100A](https://ru.wikipedia.org/w/index.php?title=Hewlett-Packard_9100A&action=edit&redlink=1) с поддержкой ОПН. Этот калькулятор сделал обратную польскую нотацию популярной среди учёных и инженеров, даже несмотря на то, что в ранней рекламе 9100A ОПН не упоминалась. В [1972](https://ru.wikipedia.org/wiki/1972) калькулятор [HP-35](https://ru.wikipedia.org/w/index.php?title=HP-35&action=edit&redlink=1) с поддержкой ОПН стал первым научным карманным калькулятором.

В 1971 году появился оригинальный [язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) [Forth](https://ru.wikipedia.org/wiki/Forth" \o "Forth), языковая машина которого имеет двухстековую структуру и где все вычисления проводятся на [стеке](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%BA). В этом языке ОПН является естественным способом записи любых операций с данными, хотя возможна, при желании, реализация и обычной ([инфиксной](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%B8%D0%BA%D1%81%D0%BD%D0%B0%D1%8F_%D0%BD%D0%BE%D1%82%D0%B0%D1%86%D0%B8%D1%8F)) записи арифметических операций.

ОПН применялась в советском инженерном калькуляторе Б3-19М (совместная разработка с ГДР), выпущенном в 1976 году. Все выпускаемые в [СССР](https://ru.wikipedia.org/wiki/%D0%A1%D0%A1%D0%A1%D0%A0) вплоть до конца 1980-х годов [программируемые микрокалькуляторы](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D0%BB%D1%8C%D0%BA%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80), за исключением «[Электроника МК-85](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%B8%D0%BA%D0%B0_%D0%9C%D0%9A-85)» и «[Электроника МК-90](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%B8%D0%BA%D0%B0_%D0%9C%D0%9A-90)», использовали ОПН — она проще реализовывалась и позволяла обойтись в программировании вычислений меньшим числом команд, по сравнению с обычной алгебраической нотацией, а количество программной памяти в этих моделях всегда было критическим ресурсом. ОПН используется в современных российских программируемых калькуляторах «[Электроника МК-152](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%B8%D0%BA%D0%B0_%D0%9C%D0%9A-152)» и «[ЭЛЕКТРОНИКА МК-161](https://ru.wikipedia.org/wiki/%D0%AD%D0%9B%D0%95%D0%9A%D0%A2%D0%A0%D0%9E%D0%9D%D0%98%D0%9A%D0%90_%D0%9C%D0%9A-161)», что обеспечивает их совместимость с программами, написанными для советских калькуляторов.

Отличительной особенностью обратной польской нотации является то, что все аргументы (или [операнды](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D0%BD%D0%B4)) расположены перед знаком операции. В общем виде запись выглядит следующим образом:

* Запись набора операций состоит из последовательности операндов и знаков операций. Операнды в выражении при письменной записи разделяются пробелами.
* Выражение читается слева направо. Когда в выражении встречается знак операции, выполняется соответствующая операция над двумя последними встретившимися перед ним операндами в порядке их записи. Результат операции заменяет в выражении последовательность её операндов и её знак, после чего выражение вычисляется дальше по тому же правилу.
* Результатом вычисления выражения становится результат последней вычисленной операции.

Например, рассмотрим вычисление выражения 7 2 3 \* − (эквивалентное выражение в инфиксной нотации: 7 − 2 \* 3).

1. Первый по порядку знак операции — «\*», поэтому первой выполняется операция умножения над операндами 2 и 3 (они стоят последними перед знаком). Выражение при этом преобразуется к виду 7 6 − (результат умножения — 6, — заменяет тройку «2 3 \*»).
2. Второй знак операции — «−». Выполняется операция вычитания над операндами 7 и 6.
3. Вычисление закончено. Результат последней операции равен 1, это и есть результат вычисления выражения.

Очевидное расширение обратной польской записи на унарные, тернарные и операции с любым другим количеством операндов: при использовании знаков таких операций в вычислении выражения операция применяется к соответствующему числу последних встретившихся операндов.

Особенности обратной польской записи следующие:

* Порядок выполнения операций однозначно задаётся порядком следования знаков операций в выражении, поэтому отпадает необходимость использования скобок и введения приоритетов и ассоциативности операций.
* В отличие от инфиксной записи, невозможно использовать одни и те же знаки для записи унарных и бинарных операций. Так, в инфиксной записи выражение 5 \* (−3 + 8) использует знак «минус» как символ унарной операции (изменение знака числа), а выражение (10 − 15) \* 3 применяет этот же знак для обозначения бинарной операции (вычитание).

Конкретная операция определяется тем, в какой позиции находится знак.

Обратная польская запись не позволяет этого: запись 5 3 − 8 + \*(условный аналог первого выражения) будет интерпретирована как ошибочная, поскольку невозможно определить, что «минус» после 5 и 3 обозначает не вычитание; в результате будет сделана попытка вычислить сначала 5 − 3, затем 2 + 8, после чего выяснится, что для операции умножения не хватает операндов.

Чтобы всё же записать это выражение, придётся либо переформулировать его (например, записав вместо выражения − 3 выражение 0 − 3), либо ввести для операции изменения знака отдельное обозначение, например, «±»: 5 3 ± 8 + \*.

* Так же, как и в инфиксной нотации, в ОПН одно и то же вычисление может быть записано в нескольких разных вариантах. Например, выражение (10 − 15) \* 3 в ОПН можно записать как 10 15 − 3 \*, а можно — как 3 10 15 − \*
* Из-за отсутствия скобок обратная польская запись короче инфиксной. За этот счёт при вычислениях на калькуляторах повышается скорость работы оператора (уменьшается количество нажимаемых клавиш), а в программируемых устройствах сокращается объём тех частей программы, которые описывают вычисления. Последнее может быть немаловажно для портативных и встроенных вычислительных устройств, имеющих жёсткие ограничения на объём памяти.

Автоматизация вычисления выражений в обратной польской нотации основана на использовании [стека](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%BA). Алгоритм вычисления для стековой машины элементарен:

1. Обработка входного символа
   * Если на вход подан операнд, он помещается на вершину стека.
   * Если на вход подан знак операции, то соответствующая операция выполняется над требуемым количеством значений, извлечённых из стека, взятых в порядке добавления. Результат выполненной операции кладётся на вершину стека.
2. Если входной набор символов обработан не полностью, перейти к шагу 1.
3. После полной обработки входного набора символов результат вычисления выражения лежит на вершине стека.

Реализация стековой машины, как программная, так и аппаратная, чрезвычайно проста и может быть очень эффективной. Обратная польская запись совершенно унифицирована — она принципиально одинаково записывает унарные, бинарные, тернарные и любые другие операции, а также обращения к функциям, что позволяет не усложнять конструкцию вычислительных устройств при расширении набора поддерживаемых операций. Это и послужило причиной использования обратной польской записи в некоторых научных и программируемых микрокалькуляторах.

# Постановка задачи

Выполнение работы предполагает решение следующих задач:

1. Разработка и реализация метода, обеспечивающего преобразование выражений из инфиксной в постфиксную нотацию.
2. Разработка и реализация метода вычисления конечного ответа выражений, записанных в польской нотации.
3. Разработка программы, демонстрирующей работу вышеперечисленных методов (получается из файла кода «Polish\_main.cpp»).
4. Набор автоматических тестов с использованием Google C++ Testing Framework.
5. Создание класса для обработки исключений – MyException

# Руководство пользователя

При запуске консольного приложения «Polish\_main.cpp» пользователя просят ввести произвольную последовательность операций над числами (все записывается в одну строку как в калькуляторе). Затем программа выводит эту же последовательность, но уже записанной по правилам обратной польской записи.

# Руководство программиста

## Описание структуры программы

Программа состоит из следующих модулей:

* Модуль Polish. Содержит пример использования методов для работы с ОПЗ. Реализация в файле «Polish \_main.cpp*».*
* Модуль PolishLib – статическая библиотека. Содержит файл «Polish.h», в котором описаны прототипы методов для работы с «Обратной польской записью» и «Polish.cpp», содержащий их реализацию. Также в модуле содержатся файлы «Stroka.h» и «Stroka.cpp» в которых описан интерфейс и реализация вспомогательного класса TString.
* Модуль PolishTest. Содержит 21 теста, описанные в файле «PolishTest.cpp*»,* разработанные с помощью использования Google C++ Testing Framework.
* Модуль ExceptionLib – библиотека, содержащая класс исключений.

## Описание структур данных

#### Класс TString – класс строк.

Поля со спецификатором доступа private:

* char\* s - указатель на область памяти для хранения строки.
* int length – размер строки.

Конструкторы и методы класса, объявленные со спецификатором public:

* TString()- конструктор по умолчанию.
* TString(TString &A)- конструктор копирования.
* ~TString()– деструктор.
* TString operator+(TString &A)- метод, позволяющий сложить две строки.
* TString& operator=(TString &A)- метод, позволяющий производить операцию присваивания.
* char& operator[](int i) – метод, позволяющий обращаться к элементам строки по индексу.
* friend std::istream& operator>>(std::istream &A, TString &B)-перегрузка ввода для строки.
* friend std::ostream& operator<<(std::ostream &A, const TString &B)- перегрузка вывода для строки.

## Описание алгоритмов

Рассмотрим алгоритм, который осуществляет предвычисление констант в выражении. Дано выражение в ОПЗ - ОПН (обратной польской записи – обратной польской нотации). Нам понадобится стек для хранения смешанных данных (чисел и операторов).

Алгоритм подобен тому, который применяется для вычисления выражений. Мы просматриваем выражение слева направо.

Пока есть символы для чтения:

* Читаем очередной символ.
* Если символ является числом, помещаем его в стек.
* Если символ является переменной, считая что переменная имеет значение ***null***, помещаем символ в стек.
* Если символ является оператором:
* 1) (если все аргументы оператора, лежащие в стеке, имеют значение, отличное от ***null***) выталкиваем аргументы оператора из стека и помещаем в стек результат операции;
* 2) (если хотя бы один из аргументов имеет значение ***null***) считая что результат операции ***null***, кладём символ оператора в стек.

После того, как всё выражение просмотрено, то, что осталось в стеке, является оптимизированым выражением (операторы выражения лежат в стеке в обратном порядке).

# Заключение

Все поставленные задачи были выполнены это подтверждается Google C++ Testing Framework (все 21 теста успешно выполняются) и работой с файлом кода «Polish\_main.cpp» использующего наш класс.

# Литература

* Книги:

1. A.O. Грудзинский. Методы программирования, Издательство Нижегородского госуниверситета, 2006.
2. Топп У., Форд У. Структуры данных в С++. - М. Бином, 1999.
3. Мейн М., Савитч У. Структуры данных и другие объекты в С++. - М.: Издательский дом “Вильямс”, 2003.
4. Гергель В.П. Методические материалы по курсу «Методы программирования 2», Нижний Новгород, 2015.

* Ссылки в Internet:

1. Учебно-методическое пособие из электронной библиотеки ННГУ: «ЛАБОРАТОРНЫЙ ПРАКТИКУМ» : [http://www.unn.ru/books/met\_files/Pract\_ADS.pdf].
2. Википедия [https://ru.wikipedia.org/wiki/Обратная\_польская\_запись]