**ЛАБОРАТОРНАЯ РАБОТА №3-5**

***Цель*:** научиться реализовывать стековый калькулятор

**Отчет по работе** должен содержать **2 файла: 1) код** программы**; 2)** файл в формате **.pdf** со скринами результатов работы (входные данные и полученный результат).

**ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ**

**Стековый калькулятор** − это устройство (реальное или виртуальное), которое вычисляет значения арифметических выражений, записанных ***в постфиксной форме***. Стековым он называется по понятным причинам: если встречается число, то его нужно поместить его в стек; если знак операции –необходимо достать из стека столько чисел, сколько необходимо для выполнения данной операции.

*Именно так и работает т.н. математический сопроцессор (например, в INtel8086+AMD64) - часть ЦП, отвечающая за операции над числами с плавающей запятой.*

**Обратная польская нотация** (Revers Polish Notation) или, иначе, **постфиксная запись** арифметического выражения **позволяет избежать записи скобок** для обозначения приоритета арифметических операций.

Рассмотрим арифметическое выражение

(2−3)∗(12−10)+4/2

Его значение легко вычисляется и оказывается даже целым − это 0 (не забываем правила приоритета операций!). Это привычная для нас форма записи арифметических выражений, в которой если операция бинарная (т.е. требует 2 аргументов, например, сложение, деление), то один аргумент пишется перед знаком операции, а другой − после неё. Такая форма записи называется **инфиксной**.

**Нотация** (способ записи), в которой операнды пишутся перед знаком операции − называется **постфиксной** или **обратной польской** (**Revers Polish Notation RPN**).

**Нотация**, в которой операнды пишутся после знака операции − **прямой польской** или **префиксной**.

Например, уже рассмотренное выражение в **обратной польской записи** будет выглядеть как

2 3 – 12 10 − ∗ 4 2 / +

После пары 2 и 3 стоит знак вычитания. После пары 12 и 10 − тоже. Далее стоит знак умножения, потому что результаты этих двух операций надо умножить. Далее стоят 4 и 2 и после них знак деления. А после - знак сложения, показывающий, что результат предыдущего нужно сложить с результатом деления 4 на 2.

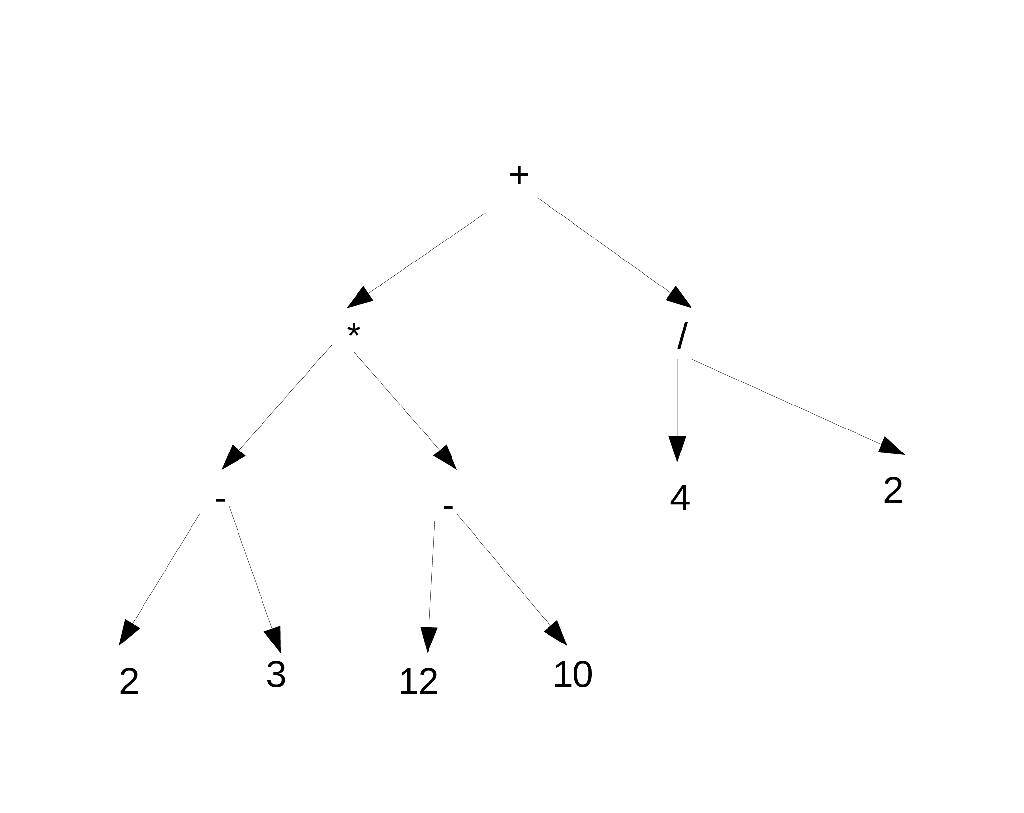
а в **прямой польской** как

+ ∗ − 2 3 – 12 10 / 4 2

Аналогичным образом, только теперь знак операции стоит перед операндами (или их описаниями в виде выражений в той же форме записи).

В общем и целом, любое выражение можно представить в виде структуры, называемой **деревом** (синтаксическим деревом в данном случае, поскольку оно отражает структуру выражения).

Например, для разобранного выражения синтаксическое дерево будет выглядеть так:



Его конечные вершины, листья (из которых стрелки никуда не идут) − это операнды, а промежуточные (из которых идут стрелки) − операции. Прямая польская запись (префиксная) получается, если читать это дерево сверху вниз. Обратная (постфиксная) − если читать снизу вверх.

**Задание 1.**

Реализовать стековый калькулятор. Написать программу, которая читает выражение в обратной польской нотации и считает его значение или пишет, что выражение составлено некорректно (если оно некорректно).

**Формат ввода**

На первой строчке вводится целое число ***N*** – количество элементов в последовательности в записи ***RPN***.

На следующих ***N*** строках вводятся либо целые числа, либо один из арифметических символов: ***”+”, “-“, “\*”, “:”.***

**Формат вывода**

Одно целое число – значение арифметического выражения.

**Пример 1.**

**(1+2)\*3+10:2 - инфиксная нотация**

**1 2 + 3 \* 10 2 : + - обратная польская нотация, постфиксная**

|  |  |
| --- | --- |
| **Ввод** | **Вывод** |
| 9 | 14 |
| 1 |  |
| 2 |  |
| + |  |
| 3 |  |
| \* |  |
| 10 |  |
| 2 |  |
| : |  |
| + |  |

**ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ**

**Сортировочная станция Дейкстры**

Как нетрудно увидеть, выражения в обратной польской записи удобны для чтения компьютером, но неудобны для чтения и составления людьми.

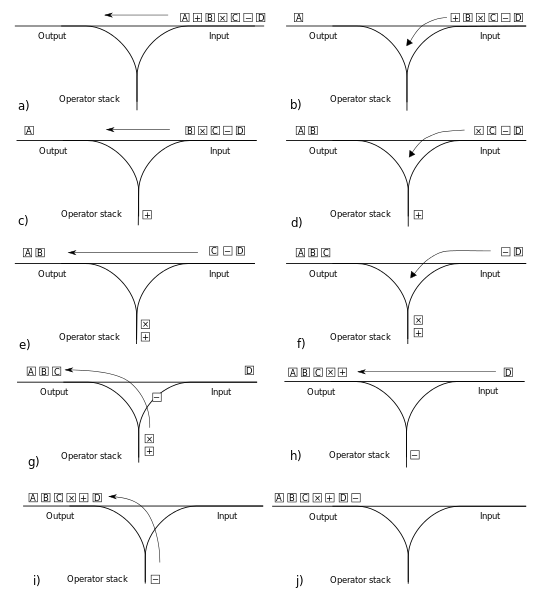
Поэтому хотелось бы доверить труд составления постфиксной формы выражений по их привычной инфиксной форме компьютеру.

Это можно сделать с помощью так называемого алгоритма сортировочной станции (Shunting Yard algorithm , придуман Э. Дейкстрой в 1961 году, см [https://en.wikipedia.org/wiki/Shunting-yard\_algorithm](https://en.wikipedia.org/wiki/Shunting-yard_algorithm%20) для трансляторов языка Algol60 <https://ir.cwi.nl/pub/9251>).

В следующем примере, взятом из википедии, разбирается преобразование выражения

A+B∗C−D

Есть сортировочная станция с 3 путями: 2 подъездных и 1 тупик. С правого подъездного пути едут выражения: в каждом «вагоне» или операнд или знак операции. Операнды свободно проезжают в левый путь (образуя очередь), а операторы заезжают в тупик.



By Salix alba - Own work, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=10960619>

Если приоритет входящего оператора ниже приоритета того, что на вершине стека (на входе в тупик), то из стека достаётся оператор с большим приоритетом (на шаге g это умножение). То же самое происходит, если приоритет приходящего оператора равен приоритету такового на вершине стека, но тот, что на вершине - ассоциативен.

**Задание 2\* (по желанию).**

Реализовать на python алгоритм сортировочной станции для преобразования произвольных арифметических выражений с 4 действиями (+,-,\*,/) из инфиксной записи в обратную польскую.