# Сортировочная станция

Алгоритм сортировочной станции — алгоритм разбора математических выражений, представленных в инфиксной записи. Выход — очередь токенов, из которой можно получить: выражение в **постфиксной нотации [обратной польской нотации (RPN)], абстрактное синтаксическое дерево выражения (AST).**

Инфиксная запись — как привыкли люди. «2 \* (1 + 3) / 4»

Постфиксная запись — как привыкли машины. «2 1 3 + \* 4 /»

Очевидный плюс — отсутствие скобок.

**Абстрактное синтаксическое дерево** — представление выражение в виде дерева

**Токен** — логически неделимая последовательность символов в выражении.

**Токенизация** (лексический анализ) — получение из входной последовательности токенов. В выражении «2.5 \* (1 + 3) / 4 + sin(10)»:

• 2.5, 1, 3, 4, 10 — токены, тип «число»;

• (, ), (, ) — токены, тип «скобка»;

• \*, +, /, + — токены, тип «операция»;

• sin — токен, тип «функция».

В ходе работы алгоритм сортировочной станции использует следующие конструкции:

• стек операторов;

• результирующая очередь (очередь вывода).

**Для использования алгоритма нужно определить некоторые свойства операторов:**

1) Приоритет оператора

Свойство, которое определяет порядок применения оператора.

Чем выше приоритет, тем раньше применяется оператор (к примеру, сложение, вычитание — 1, умножение, деление — 2, возведение в степень — 3).

2) Ассоциативность оператора Свойство, определяющее правила применения оператора к операндам (вот и линал пригодился).

Ассоциативность: 2 + 3 + 4 = (2 + 3) + 4 = 2 + (3 + 4)

Левая ассоциативность: 2 – 4 – 6 = (2 – 4) – 6 != 2 – (4 – 6)

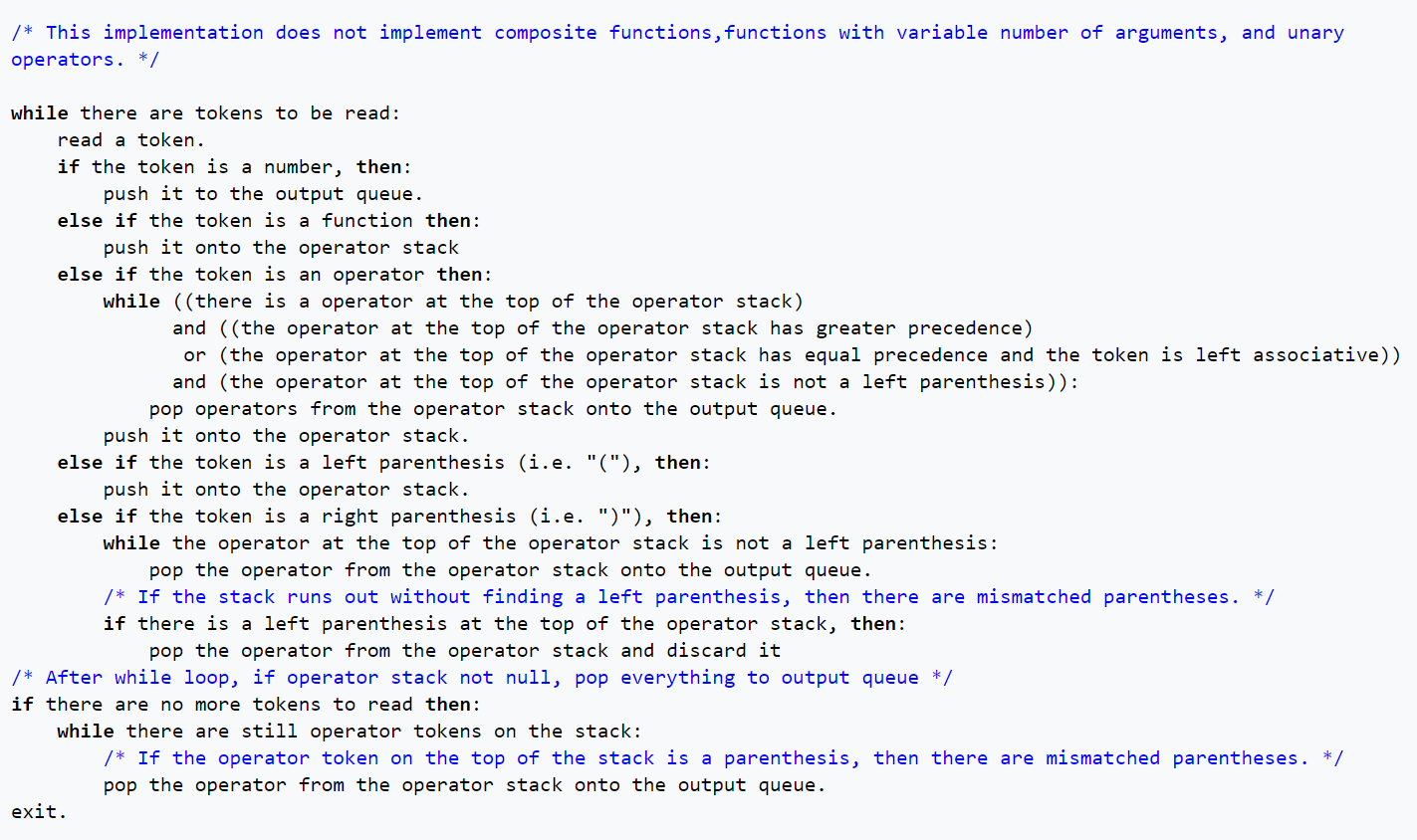
Правая ассоциативность: 2 ^ 3 ^ 4 = 2 ^ (3 ^ 4) != (2 ^ 3) ^ 4

3) Количество операндов

Сколько операндов нужно оператору (также относится и к функциям: количество аргументов функции).

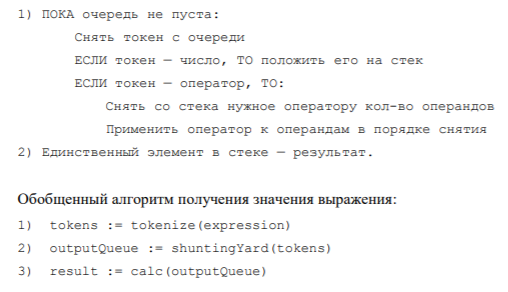
Бинарные операторы: 2 – 3, 2 + 3, 2 \* 3 … Унарные операторы: -1, ~5, …

**Непосредственно алгоритм разбора: Алгоритм принимает на вход последовательность токенов.**



**Алгоритм подсчета значения выражения:**

Вход алгоритма — очередь токенов, полученная после разбора выражения. Алгоритм использует стек значений как вспомогательную конструкцию.



**Что делать с унарным минусом?** Если минус стоит у числа, то на этапе токенизации можно сразу считать отрицательное число. Если минус стоит у функции, то можно преобразовать подстроку –f(x) к виду –1 \* f(x).