Неограниченный калькулятор

Документация к проекту

Оглавление

[Введение 3](#_Toc23949424)

[Как это работает? 3](#_Toc23949425)

[Схема проекта: 4](#_Toc23949426)

[LongNumber.h 5](#_Toc23949427)

[LongNumber.cpp 5](#_Toc23949428)

[Calculator.h 14](#_Toc23949429)

# Введение

Неограниченный калькулятор – это программа, которая получает на вход строку, содержащую математический пример. В примере присутствуют целые числа, скобки “()” , операции сложения “+”, вычитания ”-“,умножения ”\*”. На выходе программа содержит строку, содержащую ответ к примеру. Числа в программе могут быть очень длинными – до 4 294 967 295 знаков

Пример:

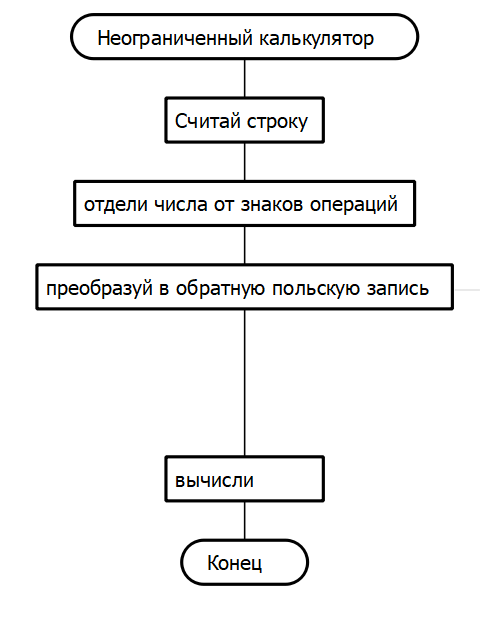
Входная строка: 30000\*20000 -400000 + 2\*(700000000 - 5000) – 8000

Выходная строка: 1999582000

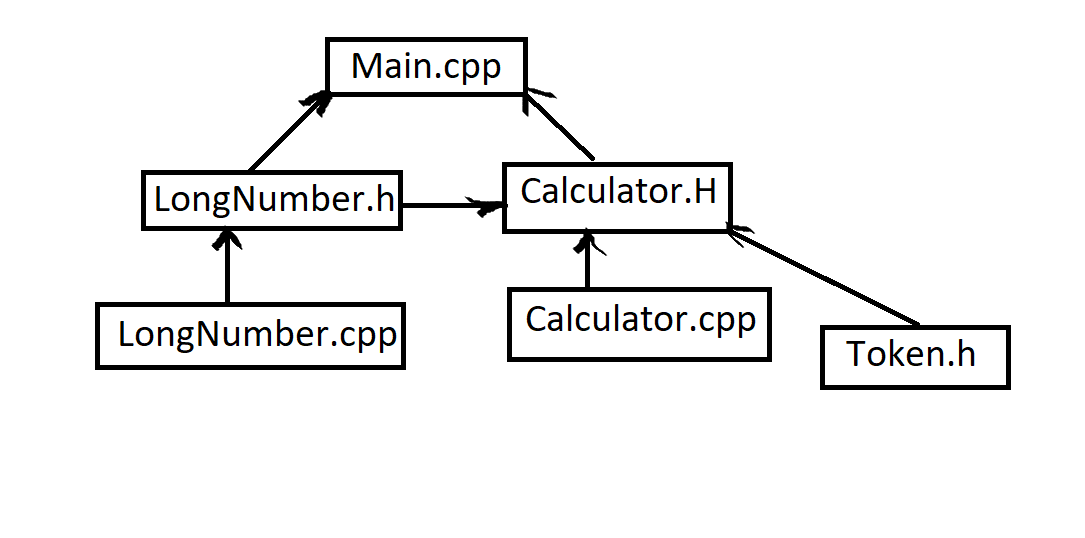
Примечание: при вводе некорректных данных программа выбрасывает исключение.

# Как это работает?

Работу алгоритма можно выразить по следующей схеме:



# Схема проекта:



# LongNumber.h

В этом файле описывается класс «длинное число».

Класс имеет следующие свойства:

std::vector<unsigned char> m\_longNumber; // представление длинного числа

Sign m\_sign = Sign::PLUS; // знак числа

Класс имеет следующие методы:

LongNumber::LongNumber(const LongNumber& other); //конструктор копирования

LongNumber::LongNumber(const std::string& str); //конструктор объекта

void Read(const std::string& str); //чтение числа

std::string Print() const; //печать

static int compare(const LongNumber left, const LongNumber right); //сравнение двух чисел

static LongNumber multiplyByDigit(LongNumber& left, unsigned char right); //умножение длинного числа на цифру

void trim();//удаление незначащих нулей

friend LongNumber operator+(LongNumber left, LongNumber& right); //сложение

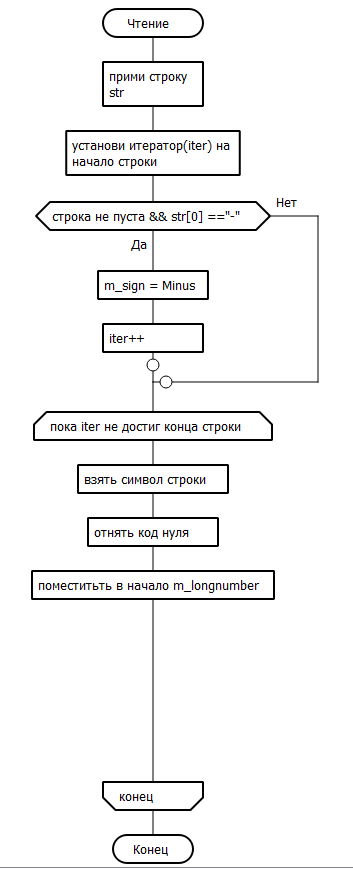
friend LongNumber operator-(LongNumber& left, LongNumber& right); //вычитание

friend LongNumber operator\*(LongNumber& left, LongNumber& right); //умножение

# LongNumber.cpp

Содержит реализацию методов класса

Read():



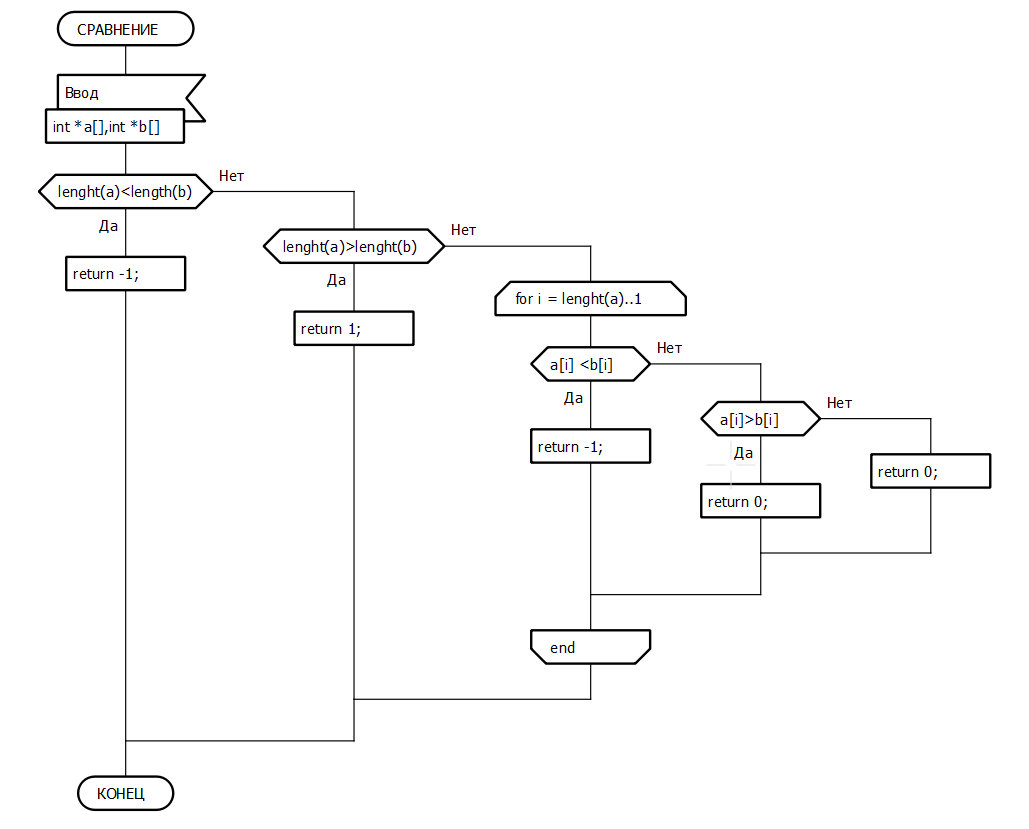
Числа в вектор записываются в обратном порядке

Зачем отнимать код нуля? Цифры 0… 9 в кодировке ASCII представлены в виде 48 + значение цифры. При преобразовании типа char к int, в целочисленную переменную передается код ASCII, значит для получения значения необходимо от кода вычесть 48, т.е код нуля.

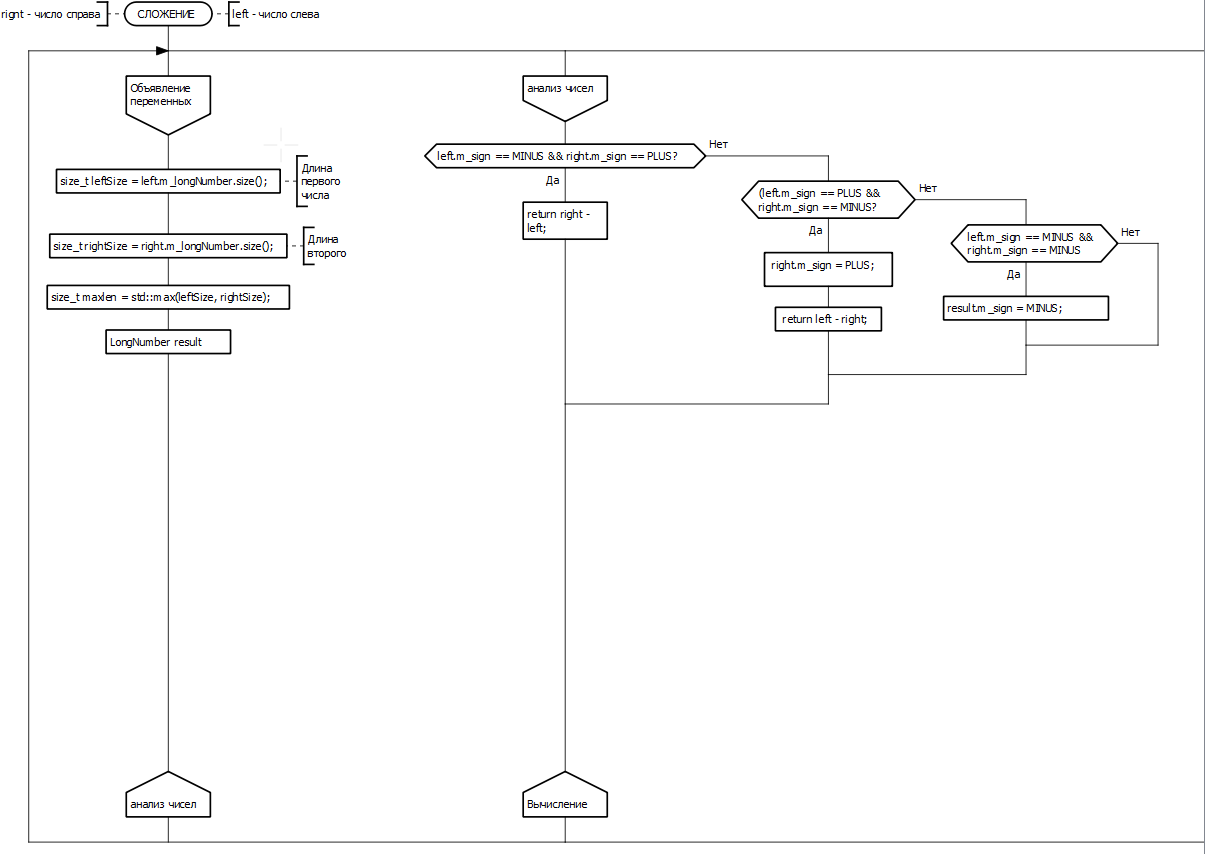
Print():

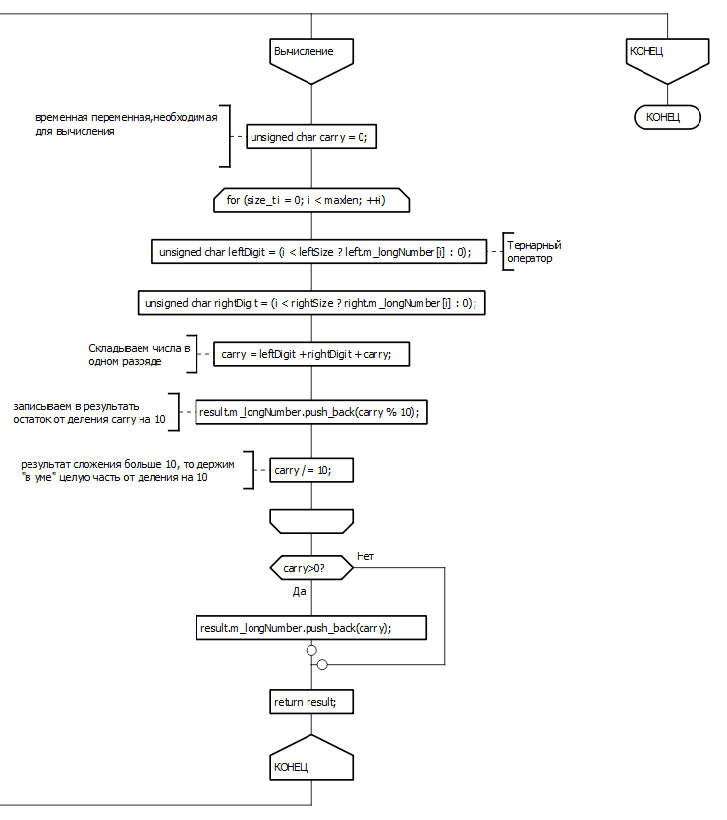


Сompare():

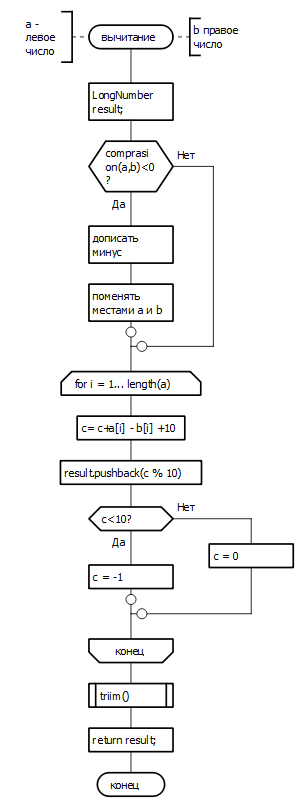


operator+:

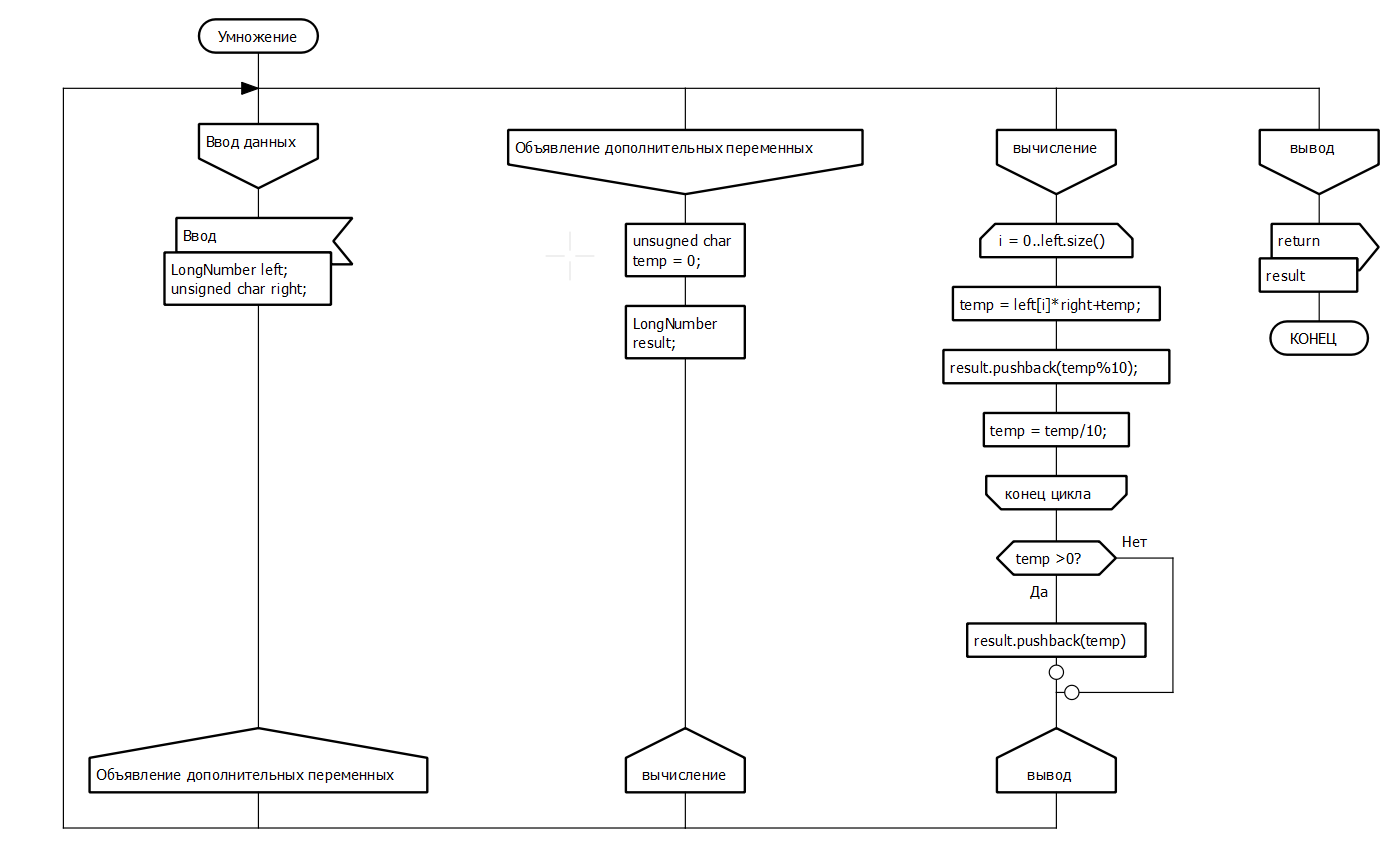




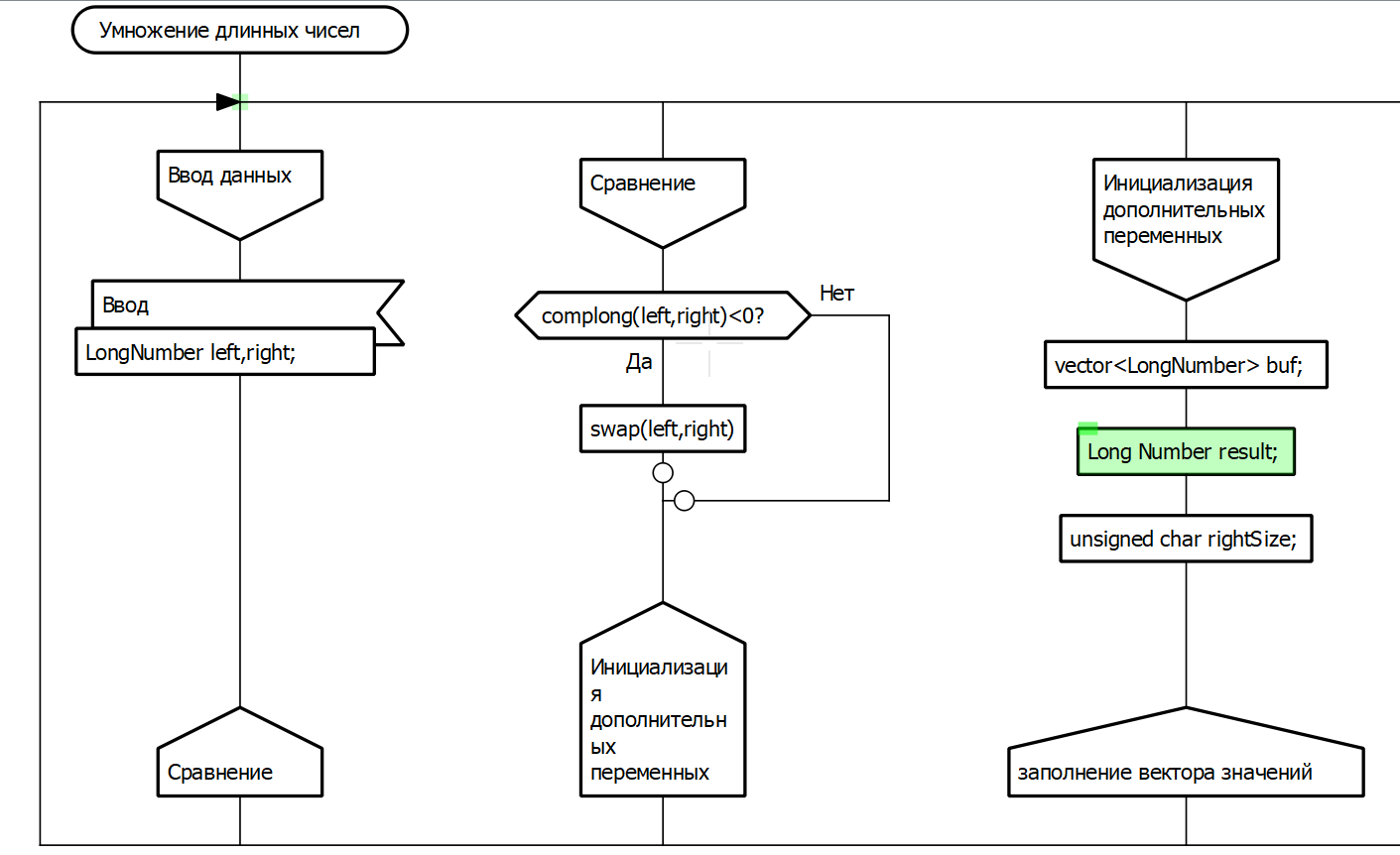
operator-:

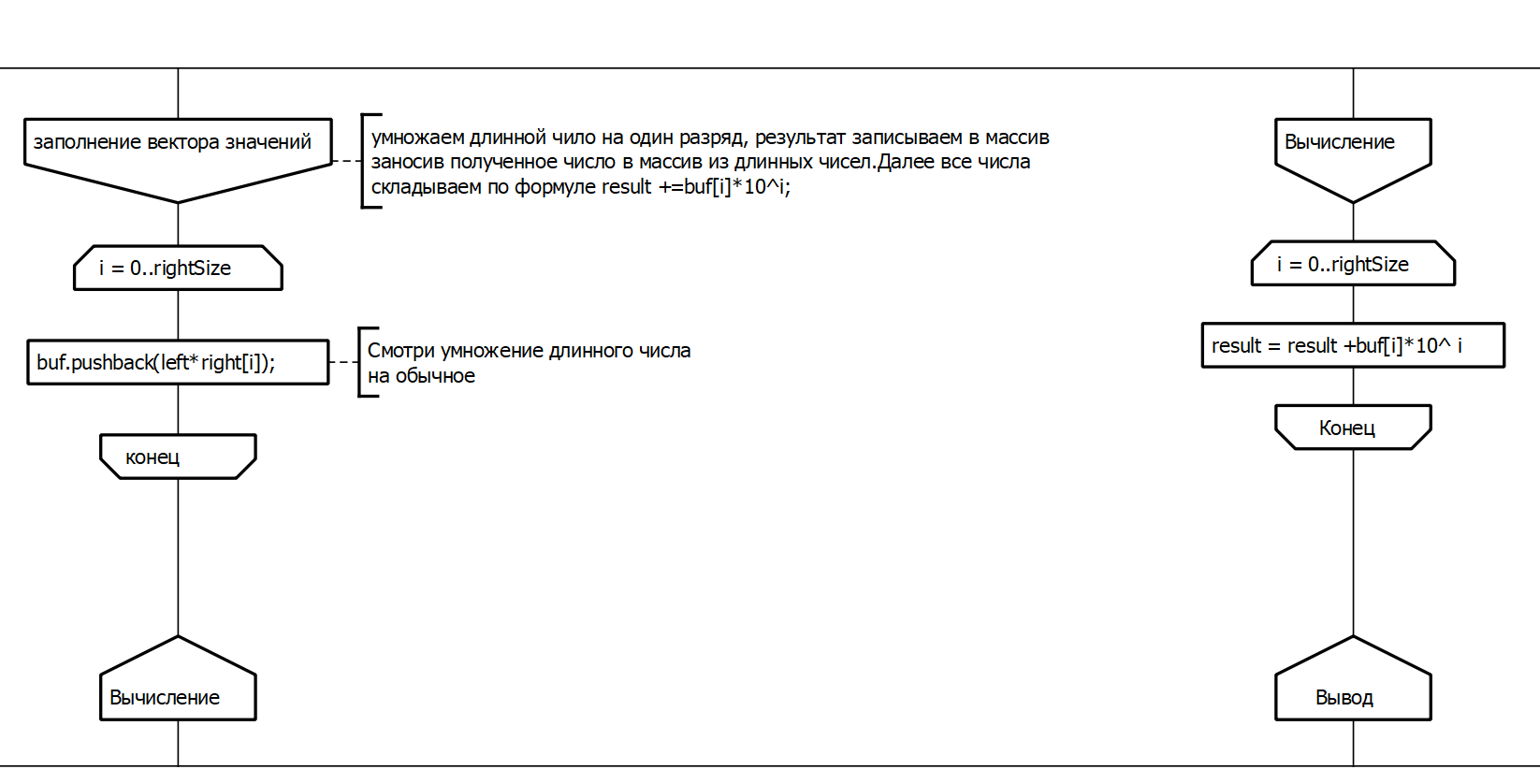


multiplyByDigit():



Operator\*





# Calculator.h

Этот файл описывает класс, который обеспечивает итоговое вычисление примера

Методы класса:

static LongNumber evaluate(const std::string&);// итоговое вычисление

static std::vector<Token> tokenize(const std::string&);// разбиение строки на лексемы

static std::vector<Token> infix2rpn(const std::vector<Token>&);// преобразование в обратную польскую нотацию

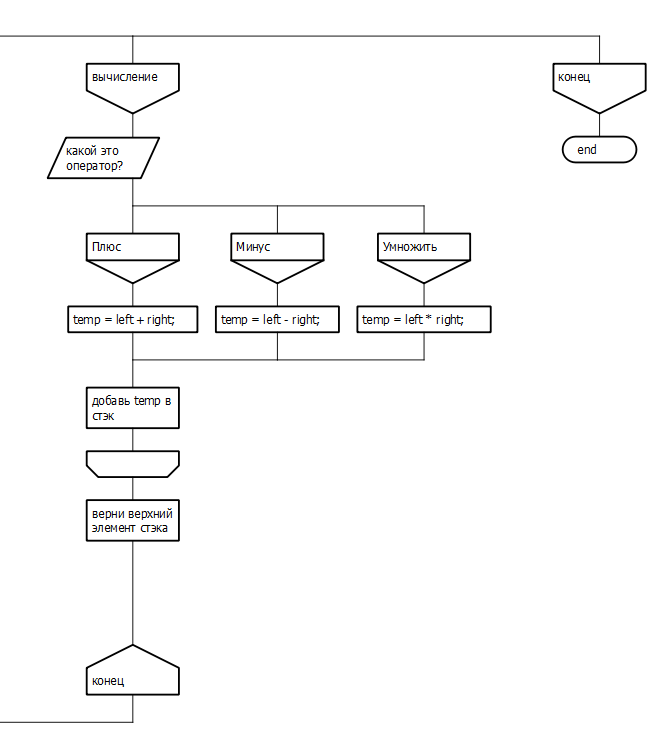
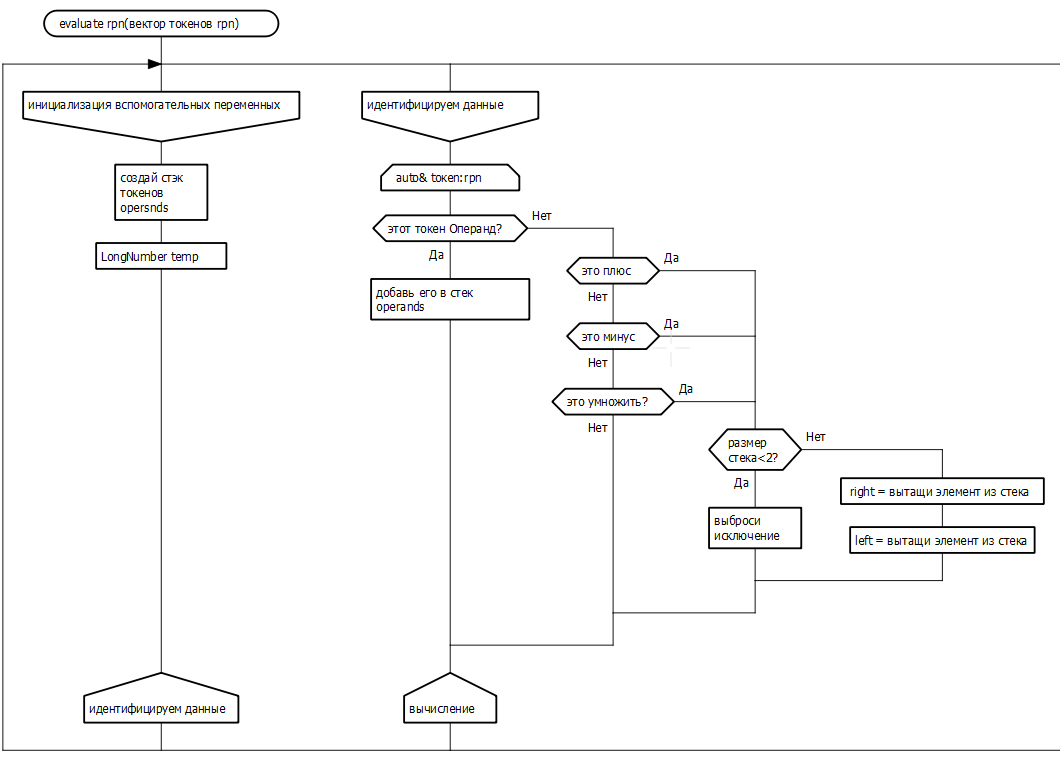
static LongNumber evaluateRpn(const std::vector<Token>&); //вычисление примера в обратное польской нотации

static int getPriority(const Token&); //назначение приоритета операциям

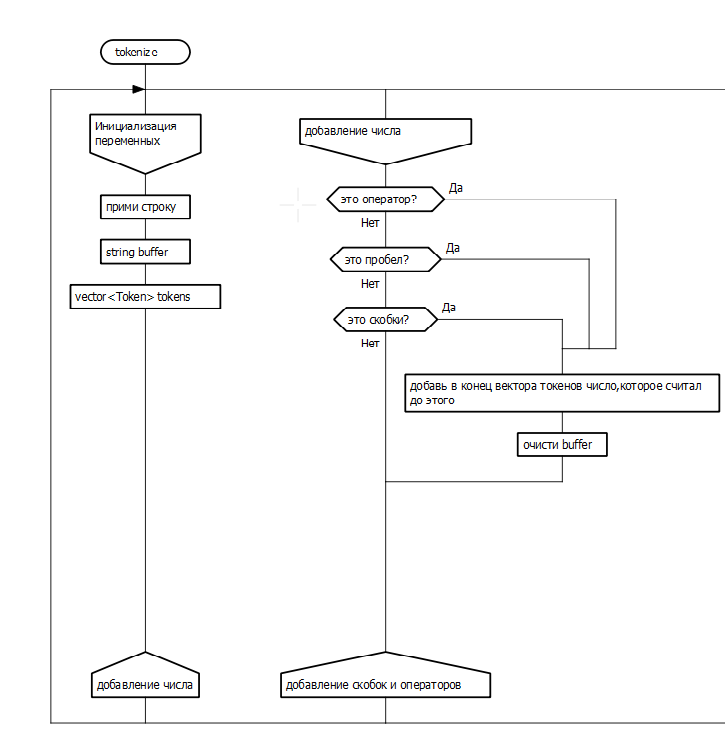
## Calculator.cpp

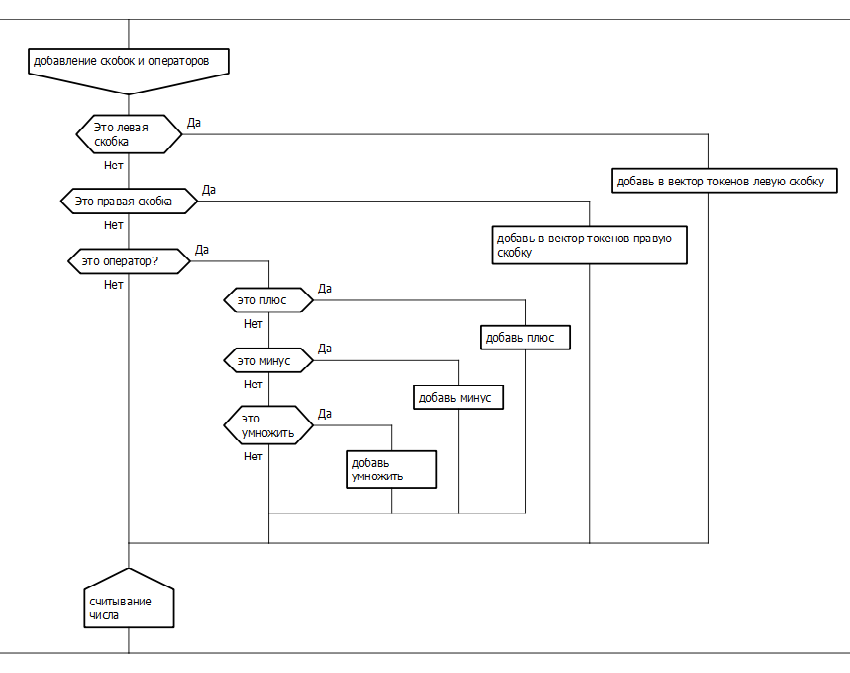
Файл содержит реализацию методов класса Calculator

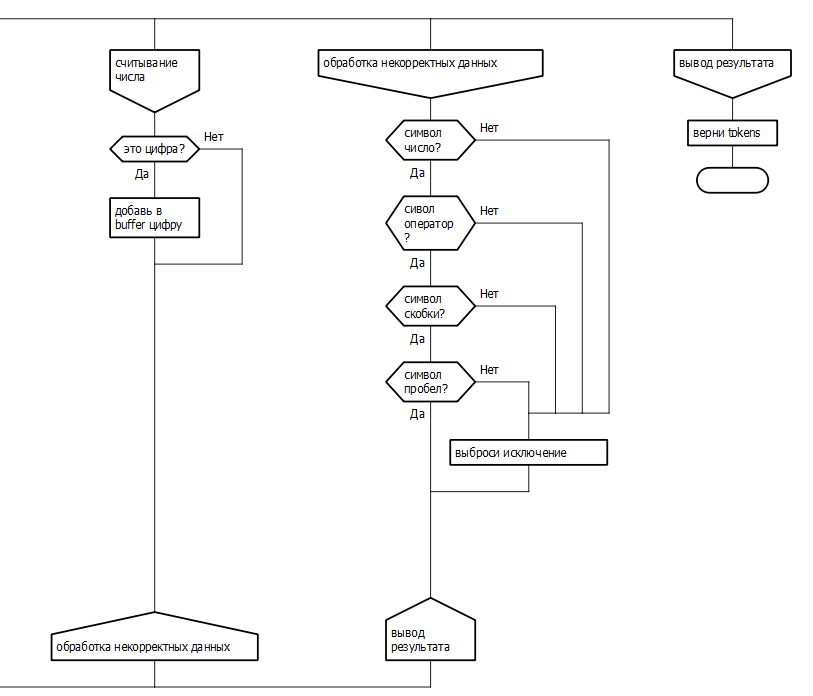
evaluateRpn():



Tokenize():







infix2rpn():

cм.файл : Неограниченный калькулятор\Документация\ДРАКОН-схемы\Операции с длинными числами