Лабораторная работа № 6 по курсу дискретного анализа: Длинная арифметика

Условие

Необходимо разработать программную библиотеку на языке C или C++, реализующую простейшие арифметические действия, проверку условий над целыми неотрицательными числами. На основании этой библиотеки, нужно составить программу, выполняющую вычисления над парами десятичных чисел и выводящую результат на стандартный файл вывода.

Метод решения

Для представления больших чисел будем использовать вектор цифр. Однако не будем ограничиваться десятичной системой счисления. Итак, представлять все числа будем массивом цифр, где каждая цифра будет в промежутке от 0 до 10^6.

Операцию сложения и вычитание будем производить в столбик. Только после этой операции нужно будет удалять ведущие нули, которые в общем случае могут появиться. Для этого будем хранить число как последовательность цифр, но в обратном порядке. Тогда за $O(1)$ можно будет удалять каждый ведущий ноль, и итоговая сложность операций будет $O(n)$.

Для умножения будем применять сразу несколько различных подходов:

- умножение в столбик $O(n^2)$
- умножение Карацубы $O(n^{log_2 3})$
- теоретико-числовое преобразование $O(n \cdot logn)$

Деление и взятие остатка можно сделать одной функцией тоже методом столбика. Где каждый раз подбирать очередную цифру бинарным поиском, тогда временная сложность этих двух функций можно оценить $O(n^2 logD)$, где D - это система счисления, в который мы представляем длинные числа (в моём случае D = 10^6).

Для возведения в степень просто воспользуемся бинарным возведением в степень и получим временную сложность порядка $O(n \cdot logn \cdot logP)$, где P - показатель степени.

Описание программы

К сожалению, тестирующая система не позволила использовать модульный подход к решению поставленной задачи, поэтому весь рабочий код размещён в единственном файле *main.cpp*.

Основным классом является *TBigInt*. Его конструктор переводит число из привычного нам представления в десятичной записи в виде строки в массив 10^6-тичных цифр.

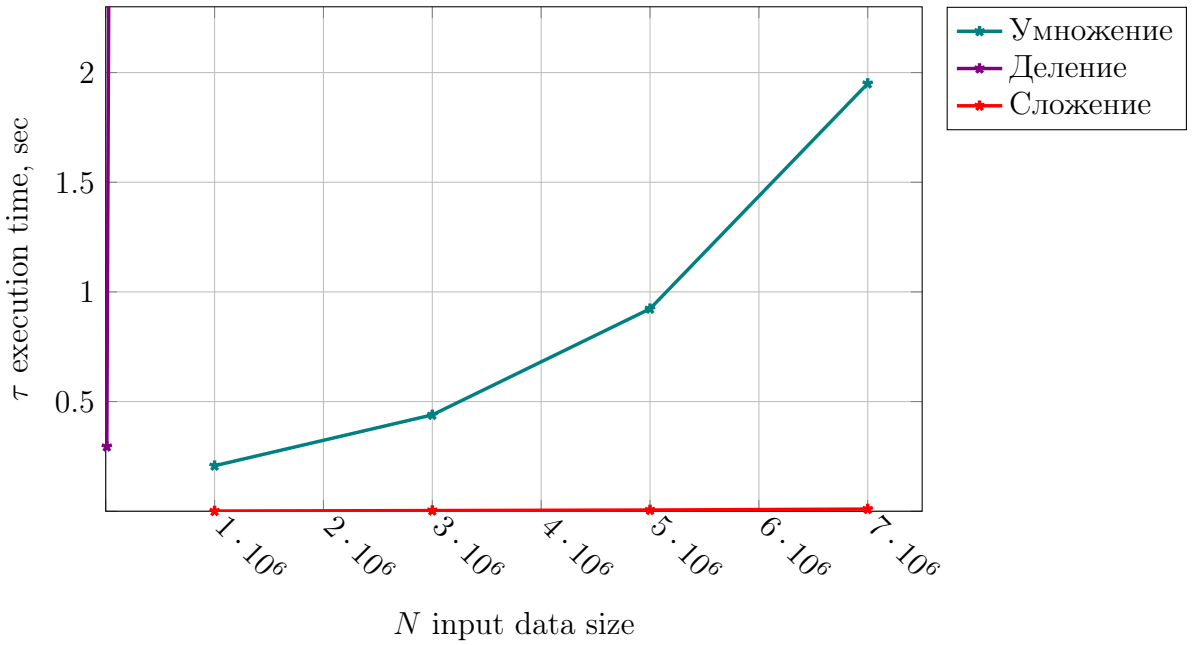
Сложение, вычитание, удаление ведущих нулей не вызывают затруднения. Что про умножение, так для начала нужно отметить тот факт, что сложность алгоритма $O(n^2)$ не есть что-то плохое, так например при умножение коротких чисел простое умножение в столбик обгоняет метод Карацубы и NTT. Поэтому определим граничные значения размеров чисел, где будем использовать тот или иной алгоритм умножения.

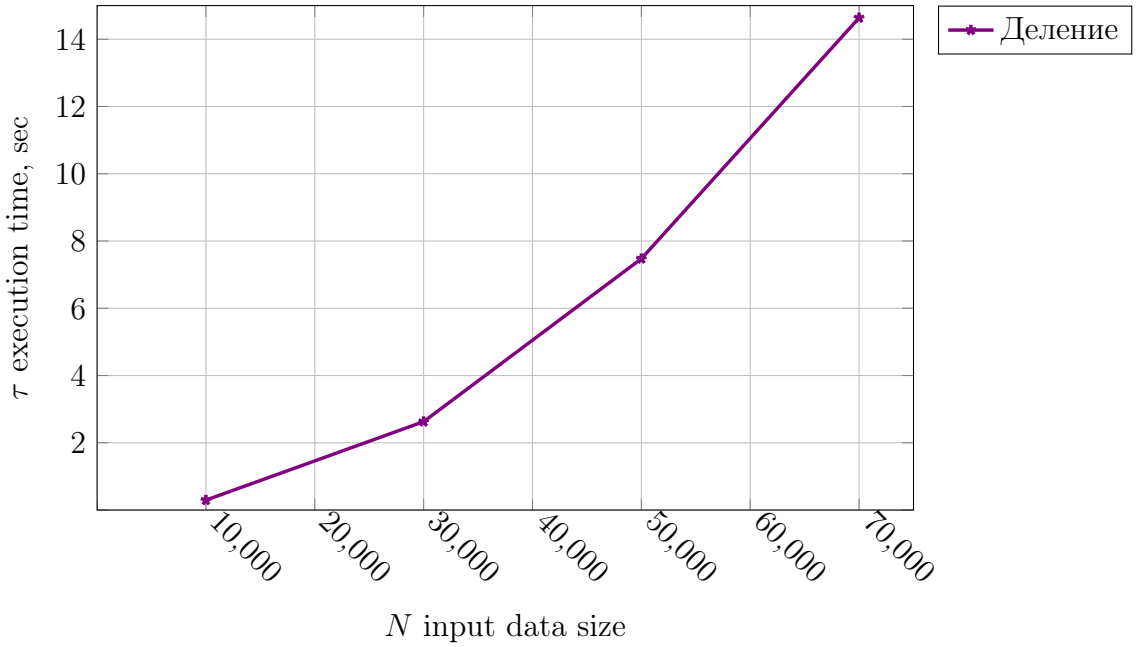
Для реализации NTT нам потребуется найти простой модуль и его примитивный корень. Чтобы оценить на сколько большой нам нужен простой модуль: он должен быть больше максимального значения в одной ячейке вектора во время выполнения умножения, а это $n*D^2$, где n длина числа в цифрах, а D мы выбрали 10^6. Учитывая, что в память влезут будут числа порядка $n = 10^5$, то модуль M нужно брать в районе 10^{18}. К счастью это можно сделать[1]. Примитивный корень (первообразный корень) — это образующий элемент мультипликативной группы кольца вычетов по модулю M. Далее имея на руках обе эти константы мы уже можем реализовать задуманный алгоритм. Правда для перемножения таких больших чисел по такому модулю M, нам придётся использовать *int128*, это затратно, но такова цена нашего большого D. Сам алгоритм рассматривает перемножение как перемножение многочленов, далее на каждом шаге вычленяются четные и нечётные степени аргумента, и рекурсивно решается подзадача, вот откуда появляется логарифм в асимптотике.

[1] https://bigprimes.org/

Тест производительности

Операция переменяется над двумя числами длины n





Выводы

В ходе выполнения лабораторной работы реализовал длинную арифметику в системе счисления 10^6. Сложнее всего было реализовать операции умножения и деления длинных чисел. Дополнительно я изучил алгоритм Карацубы, быстрое преобразование Фурье, теоретико-числовой метод перемножения полиномов, познакомился с алгоритмом Фюрера.