Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №6 по курсу «Дискретный анализ»

Студент: Ф. А. Иванов
Преподаватель: И. Н. Симахин
Группа: М8О-208Б-19

Дата: 9.04.2022
Оценка:
Подпись:

Москва, 2022

Лабораторная работа №6

Задача: Необходимо разработать программную библиотеку на языке C или C++, реализующую простейшие арифметические действия и 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.
Количество десятичных разрядов целых чисел не превышает 100000. Основание выбранной системы счисления для внутреннего представления «длинных» чисел должно быть не меньше 10000.

Формат входных данных: Входной файл состоит из последовательности заданий, каждое задание состоит из трех строк:

Первый операнд операции.

Второй операнд операции.

Символ арифметической операции или проверки условия (+, -, *, /, >, <, =).

Числа, поступающие на вход программе, могут иметь «ведущие» нули.

Формат результата: Для каждого задания из выходного файла нужно распечатать результат на отдельной строке в выходном файле:

Числовой результат для арифметических операций.

Строку Error в случае возникновения ошибки при выполнении арифметической операции.

Строку true или false при выполнении проверки условия.

В выходных данных вывод чисел должен быть нормализован, то есть не содержать в себе «ведущих» нулей.

1 Описание

Требуется разработать программную библиотеку на языке C или C++, реализующую длинную арифметику.

Как известно, длинная арифметика — выполняемые с помощью вычислительной машины арифметические операции (сложение, вычитание, умножение, деление, возведение в степень, элементарные функции) над числами, разрядность которых превышает длину машинного слова данной вычислительной машины.

В своей реализации я использую основание системы счисления равное 10^9.
Когда на вход подается длинное число, оно записывается как строка. Далее работа происходит именно со строкой. Строка разбивается на блоки в каждом из которых число не превышает 10^9. Теперь наше длиное число хранится ввиде массива таких блоков.

Переходим к операциям.
Сложение и вычитание. Самые легко реализуемые арифметические операции над такими числами. Применяю школьную логику сложения и вычитания в столбик. Задается избыток, кторый может возникнуть при сложении или при занятии разряда во время вычитания. Далее каждый блок из первого массива складывается/вычитается с блоком из второго массива.
Сложность: $O(n)$

Произведение. Здесь применяется тоже логика умножения в столбик.
Сложность: $O(n^m)$

Деление. Деление будет только целочисленным. Делить числа будем уголком. Будем сносить по одному разряду из делимого, начиная со старшего, и бинарным поиском определять разряды частного. Деление на 0 вызовет ошибку.
Сложность: $O(n * (n + m))$

Для возведения в степень воспользуемся бинарным алгоритмом, работа которого основывается на двух равенствах.
$a^k = (a^{k/2})^2$, при условии, что k - четное; $a^k = (a^{k-1} * a)$, при условии, что k - нечетное.
Сложность: $O(log(k) * n^2)$

2 Исходный код

```
class BigInt
{
private:
    vector<long long> digits;
    void removeZeroes();
public:
    BigInt(long long size);
    BigInt(long long value, bool flag);//flag -  ,
    BigInt(const string &value);
    ~BigInt();

    BigInt operator+(BigInt &second);
    BigInt operator-(BigInt &second);
    BigInt operator*(BigInt &second);
    BigInt operator/(BigInt &second);
    BigInt operator^(BigInt &second);

    bool operator>(BigInt &second);
    bool operator<(BigInt &second);
    bool operator==(BigInt &second);

    friend ostream &operator<<(ostream &out, BigInt number);

    long long getSizeNumber();
    long long getDigit(long long id);
};
```

3 Консоль

```
orion@orion-laptop:~/University/DA/Lab_06$ ./main
38943432983521435346436
354353254328383
+
3894343333787468967 4819
90409438473849324729384738 43
2343543
-
90409438473849324729361303 00
972323
2173937
>
false
2
3
-
Error
orion@orion-laptop:~/University/DA/Lab_06$
```

4 Тест производительности

Для сравнения времени работы с большой арифметикой обратимся к ЯП *python*.
Мною были написаны 2 программы для генирации тестов и для проверки скорости работы на *python*.
Прведем 3 теста.

1000 тестов.

```
orion@orion-laptop:~/University/DA/Lab_06$ python3 cheker.py
1294571
orion@orion-laptop:~/University/DA/Lab_06/solution$ ./main <test.txt
Время работы: 34120000
```

100000 тестов.

```
orion@orion-laptop:~/University/DA/Lab_06$ python3 cheker.py
101968725
orion@orion-laptop:~/University/DA/Lab_06/solution$ ./main <test.txt
Время работы: 2101068043
```

1000000 тестов.

```
orion@orion-laptop:~/University/DA/Lab_06$ python3 cheker.py
917393043
orion@orion-laptop:~/University/DA/Lab_06/solution$ ./main <test.txt
Время работы: 10316343092
```

Моя библиотека прогрывает *python*. Это происходит потому, что в *python* используются более оптимальные алгоритмы. Но по времени оно не превосходит 10^3 раз.

5 Выводы

Выполнив данную лабораторную работу, я научился работать со сколь угодно большими числами.
Основная идея заключается в том, что мы работаем с массивом, как с числом. А каждый элемент массива является единичным числом в своем диапазоне.

Я считаю, что C++ не самый удобный язык работы с большими числами. Как было показано на примере, *Python* более оптимально справляется с данной задачей.

Список литературы

[1] Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. *Алгоритмы: построение и анализ, 2-е издание.* — Издательский дом «Вильямс», 2007. Перевод с английского: И. В. Красиков, Н. А. Орехова, В. Н. Романов. — 1296 с. (ISBN 5-8459-0857-4 (рус.))