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: В. В. Косогоров
Преподаватель: А. А. Кухтичев
Группа: М8О-206Б

Дата:
Оценка:
Подпись:

Москва, 2020

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.

1 Описание

Требуется написать реализацию основных операций в длинной арифметике.

Длинная арифметика — выполняемые с помощью вычислительной машины арифметические операции над числами, разрядность которых превышает длину машинного слова данной вычислительной машины. Эти операции реализуются не аппаратно, а программно, с использованием базовых аппаратных средств работы с числами меньших порядков.[1].

Выделив количество разрядов как базу, можно поместить каждый разряд числа в вектор, где поразрядно можно производить с большими числами действия, как с обычными числами. Часто алгоритмы стандартных операций длинной арифметики совпадают со знакомыми нам со школы арифметическими алгоритмами сложения, вычитания и т.д.

2 Исходный код

Для решения задачи я создал класс TBigInteger, в котором посредством перегрузки операторов реализованы основные операции длинной арифметики.

bigint.cpp	
void Print() const	Метод для вывода на экран большого числа
TBigInteger operator + (const TBigInteger &bi) const	Сложение двух больших чисел
TBigInteger operator - (const TBigInteger &bi) const	Вычитание. Не работает, если второе число больше первого
TBigInteger operator * (const TBigInteger &bi) const	Умножение
TBigInteger operator / (const TBigInteger &bi) const	Деление
TBigInteger operator ^ (const TBigInteger &bi) const	Возведение большого числа в большую степень
bool operator < (const TBigInteger &bi) const	Проверка, больше ли левое большое число, чем правое
bool operator == (const TBigInteger &bi) const	Проверка двух больших чисел на равенство
void DeleteExtraZeros()	Удаление ведущих нулей из большого числа

Класс для болших чисел:

```
class TBigInteger {
public:
    TBigInteger();      TBigInteger(int num);
    TBigInteger(const char *str);
    void Print() const;
    TBigInteger operator + (const TBigInteger &bi) const;
    TBigInteger operator - (const TBigInteger &bi) const;
    TBigInteger operator * (const TBigInteger &bi) const;
    TBigInteger operator / (const TBigInteger &bi) const;
    TBigInteger operator ^ (const TBigInteger &bi) const;
    bool operator < (const TBigInteger &bi) const;
    bool operator > (const TBigInteger &bi) const;
    bool operator == (const TBigInteger &bi) const;
private:
    std::vector<short> mNums;
```

```
    void DeleteExtraZeros();
    TBigInteger mPower(const TBigInteger &bi) const;
};
```

3 Консоль

```
4809489205429080923842589208424849 23
219482935892408490235209375023489235729034829042934720 2
+
219482935892408490283304267077780044967460721127183212 5

428394289042803482095902384902384092834092894 0
24213521342342
-
428394289042803482095902384902381671481958659 8

1
2141241221421312412421342354234234213423 42
-
Error

22342348283048290348294
0
*
0

29582948239482948293489238492384923424555555 5
5
/
591658964789658965869784769847698468491111 11

32348928394283
3
^
338516386079631438092857609338938278631 87

2423423423423423423423
234423423
>
true
```

4 Тест производительности

Для тестирования будем использовать генератор тестов, написанный на python. На вход программе будет подаваться случайные большие числа и знаки арифметических операций.

Тест 1: Количество операций 10000
Время 6 секунд.

Тест 2: Количество операций 50000
Время 32 секунды.

Тест 3: Количество операций 100000
Время 65 секунд.

Как видно из проведённых тестов, при увеличении количества операций вычисления в n раз, время работы программы увеличивается чуть больше, чем в n раз. Это связано с тем, что операции сложения и вычитания (O(n)), и возведения в степень (O(log n)) занимают большую часть тестовых данных.

5 Выводы

Выполнив шестую лабораторную работу по курсу «Дискретный анализ», я написал реализацию операций над большими числами, познакомился с алгоритмом быстрого возведения в степень.

Список литературы

[1] *Arbitrary-precision arithmetic*
URL: `https://en.wikipedia.org/wiki/Arbitrary-precision_arithmetic` (дата обращения: 14.04.2020)