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: Ю. Ю. Обыденкова
Преподаватель: А. Н. Ридли
Группа: М8О-308Б-18
Дата:
Оценка:
Подпись:

Москва, 2021

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.

Список арифметических операций:

- Сложение
- Вычитание
- Умножение
- Возведение в степень
- Деление

Список условий :

- Больше
- Меньше
- Равно

1 Описание

Длинные числа в программе представлены специальным классом BigInt, внутри которого находится vector<int>, хранящий разряды числа в обратном порядке. Числа хранятся в системе счисления с основанием $10^{basePower}$, где basePower - статическая константа класса(со значением 6, но это значение может быть изменено в тексте программы, изменение повлияет на быстродействие). Для длинных чисел реализованы все необходимые арифметические операции.

1 Сложение

Числа складываются поразрядно. При переполнении(в данном контексте имеется в виду, что сумма двух разрядов становится не меньше основания системы счисления) в ячейку сохраняется результат по модулю основания системы счисления, а частное результата и основания системы счисления сохраняется в специальную переменную и будет учтено при обработке следующих разрядов.

2 Вычитание

Реализовано вычитание меньшего числа из большего, если это условие не выполняется, то выбрасывается исключение. Вычитание производится поразрядно, в случае если текущий разряд уменьшаемого меньше аналогичного разряда вычитаемого, это учитывается в специальной переменной.

3 Умножение

В программе есть две функции для умножения чисел: одна из них перемножает два длинных числа, другая умножает длинное число на короткое(эта функция используется в делении). Умножение совершается наивным образом за сложность $O(n^2)$.

4 Деление

Самая сложная часть программы. Сам алгоритм деления описан в книге «Искусство программирования. Том 2» под авторством Дональда Кнута. Алгоритм основывается на теореме о том, что число $\widehat{q} = min\left(\lfloor\frac{v_n b+u_{n-1}}{v_{n-1}}\rfloor, b-1\right)$, является достаточно хорошим приближением к реальному значения конкретного разряда частного. Оно не может быть меньше его, и не превышает его больше, чем на два. Сложность деления - $O(n^2)$

5 Возведение в степень

Здесь используется алгоритм бинарного возведения в степень. В этой операции активно используется умножение длинных чисел.

6 Операторы сравнения

В классе есть функция Compare, сравнивающая числа сначала по размеру их векторов с разрядами, затем лексикографически, если векторы одного размера. Операторы сравнения используют эту функцию.

2 Исходный код

1 Описание программы

Программа состоит из четырех файлов. В трех из них(BigInt.h, BigInt.hpp, BigInt.cpp) хранится описание и реализация длинных чисел. В main.cpp хранится код, отвечающий за обработку ввода/вывода.

2 Таблица функций и методов

main.cpp	
class BigInt	
BigInt()	Конструктор по умолчанию, создает число, равное нулю
explicit BigInt(const std::string& str)	Конструктор от строки
template <typename T> explicit BigInt(T number, typename std::enable_if<std::is_integral<T>::value>::type* = 0)	Шаблонный конструктор для всех типов, представляющих целые числа
friend std::ostream& operator « (std::ostream& os, const BigInt& number)	Оператор вывода длинного числа
friend std::istream& operator » (std::istream& is, BigInt& number)	Оператор ввода длинного числа
BigInt operator + (const BigInt& other) const	Оператор сложения двух длинных чисел
BigInt operator - (const BigInt& other) const	Оператор вычитания двух длинных чисел

BigInt operator * (const BigInt& other) const	Оператор умножения двух длинных чисел
BigInt operator / (const BigInt& other) const	Оператор деления двух длинных чисел
BigInt operator ^(const BigInt& other) const	Оператор возведеня в степень длинного числа
BigInt operator * (long long other) const	Оператор умножения длинного числа на короткое
BigInt operator / (long long other) const	Оператор деления длинного числа на короткое
bool operator < (const BigInt& other) const	Оператор сравнения длинных чисел. Проверяет, что первое меньше второго
bool operator > (const BigInt& other) const	Оператор сравнения длинных чисел. Проверяет, что первое больше второго
bool operator == (const BigInt& other) const	Оператор проверки длинных чисел на равенство
bool operator != (const BigInt& other) const	Оператор проверки длинных чисел на неравенство
std::string ToString() const	Переводит длинное число в строку
BigInt(std::vector<int> v)	Приватный конструктор от вектора целых чисел
void DeleteZeros()	Удаляет ведущие нули
static int Compare(const BigInt& lhs, const BigInt& rhs)	Сравнивает числа, возвращает -1, если первое число меньше второго, 0, если они равны, и 1 в оставшемся случае

3 Консоль

```
julia@julia21novo:~/CLionProjects/da_06/cmake-build-debug$ ./da_06
38943432983521435346436
354353254328383
+
3894343333787468967 4819
90409438473849324729384738 43
2343543
-
90409438473849324729361303 00
972323
2173937
>
false
2
3
-
Error
```

4 Выводы

Благодаря данной работе я узнала о нескольких новых алгоритмах и научилась тщательнее тестировать свои программы на предмет багов и ошибок. Длинные числа в программе представлены специальным классом BigInt, внутри которого находится vector<int>, хранящий разряды числа в обратном порядке. Для длинных чисел реализовала все необходимые арифметические операции. Так же я лучше разобралась в вещах, используемых мной каждый день и узнала об очень интересной теореме, позволившей реализовать деление длинных чисел.

Длинная арифметика - способ реализации работы с неограниченно большими числами. Однако она мало где реализована на уровне языка. Например, в Python длинные числа и длинная арифметика поддерживается на языковом уровне. В том же C++ такой поддержки нет. Возможно, это связано с узкой сферой применения. Большинство решаемых задач укладываются в стандартные типы данных.

Одна из областей применения длинной арифметики - криптография. Большинство систем подписывания и шифрования данных используют целочисленную арифметику по модулю m, где m — очень большое натуральное число, не обязательно простое.

Из того, что можно добавить: извлечение квадратного корня алгоритмом «Karatsuba Square Root» (сложность $5O(n)$), взятие остатка.

Список литературы

[1] Дональд Кнут *Искусство программирования. Том 2. Получисленные алгоритмы. — Издательство «Диалектика», 2019. Перевод с английского: И. В. Красиков, В. Т. Тертышный, Ю. В. Козаченко — 832 с. (ISBN 978-5-8459-0081-4)*