Министерство науки и образования РФ

Федеральное государственное автономное образовательное

учреждение высшего профессионального образования

«Санкт-Петербургский государственный электротехнический

университет «ЛЭТИ» им. В. И. Ульянова (Ленина)»

(СПбГЭТУ «ЛЭТИ»)

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

Пояснительная записка к курсовой работе

на тему:

“Длинная арифметика ”

по дисциплине “Программирование”

Выполнил: Быков И. В.

Группа 6383

Принял: Самойленко В. П.

Санкт-Петербург  
2016 г.

1. **Содержательная постановка задачи.**

**Дано:**

Многозначное целое число a представляется в виде последовательности (массива) целых чисел путем разбиения его цифр на группы по t цифр. Каждая группа цифр, обозначенная , является целым числом. Например, число 3 1415 9265 3509 7932 3846 при t = 4 может быть представлено в виде массива {0003, 1415, 9265, 3509, 7932, 3846}.

**Требуется:**

Реализовать следующие операции над многозначными целыми числами:

Сложение, Вычитание, Сравнение, Умножение.

1. **Анализ и пример решения задачи.**
   1. **Анализ задачи.**

Заданы 2 длинных числа. Нужно получить их сумму, разность, произведение, а также сравнить эти числа. На вход получаем целые неотрицательные числа(ввод доступен с экрана, либо с файла).

Входные числа будут записываться как строки типа string, так как это намного удобнее для пользователя, чем вводить массив. Затем строка будет преобразована в массив чисел. Результат выполнения программы в понимании пользователя есть целое число (выводится на экран и в выходной файл. Фактически – результат есть массив целых неотрицательных чисел, каждый элемент которого - число в определенной системе счисления BASE (у нас BASE = 1000). Так как мы используем умножение, то мы не можем выбрать систему счисления больше 10000. Объясним. Мы используем массив типа int. Максимальное число этого типа 2147483647 (10 разрядов). Пусть – количество разрядов в *c*. Учитывая, что количество разрядов в a и b у нас равны, получаем

. То есть . Но мы возьмем BASE = 1000, так как такая система удобнее для понимания.

* 1. **Пример решения задачи.**

Сложение, вычитание и умножение мы будем выполнять аналогично выполнению этих операций «столбиком», но в нашем случае цифрой будет являться каждый элемент массива a[i] = .

Покажем пример выполнения операции умножения.

Возьмем числа 11 111 и 22 222

1. Начинаем с младших разрядов:

1. Нужно исключить старшие лишние разряды (24). Для этого запомним в уме это число (В программе сохраняем его в буфер). Остается **642** (1).
2. Теперь младший разряд первого умножим на старший второго и прибавим 24.

1. Исключаем «2». Запоминаем. Остается 466 (2)
2. По второму числу прошлись полностью

2 запишем в отдельный разряд (3)

1. Теперь старший разряд первого числа умножаем на младший разряд второго числа, и суммируем результат с (2)

466 = 2 908

1. Исключаем «2» и запоминаем. Остается **908**
2. Старший разряд первого числа умножаем на старший разряд второго числа прибавляем двойку из предыдущего действия и прибавляем (3)

Получили: 246 908 642.

1. **Формальная постановка задачи.**
   1. **Исходные данные и ограничения на них.**

На вход подается 2 целых неотрицательных числа. Ограничения будут связаны с размером массива.

* 1. **Выходные данные.**

В зависимости от запроса пользователя выходными данными может являться массив результата (суммы, разности, произведения) либо результат выполнения функции сравнения (>, <, =).

* 1. **Связь результатов и исходных данных.**

sum = a + b;

def = a – b;

mult = a \* b;

где a и b входные числа.

1. **Спецификация программы.**
   1. **Входные данные.**
      1. **Перечень и основные характеристики исходных данных.**

На вход подаются 2 строки str1 и str2 типа string.

* + 1. **Ограничения на входные данные.**

Входные строки есть представление целых неотрицательных чисел длиной

0 , где len – длина строки, DIGIT – количество разрядов в одном элементе массива, MAX\_SIZE – максимальный размер массива.

* + 1. **Место и форма представления входных данных.**

По выбору пользователя входные данные могут быть представлены в файлах «fscanf1.txt» и «fscanf2.txt» или введены с клавиатуры.

В первом случае в каждом из файлов считывается строка до пробела или переноса строки.

Во втором случае строка вводится после соответствующих подсказок в консоли. Строки вводятся через enter.

* 1. **Выходные данные.**
     1. **Состав выходных данных**

По запросу пользователя в консоль будет выводится массив чисел суммы, разности или произведения. Размер данного массива ≤ MAX\_SIZE.

Выводится массив инвертированным. При выводе требуется учесть, что в каждом элементе массива, за исключением последнего, требуется вывести все значащие нули.

Вывод результатов полностью дублируется в файл «fprintf.txt»

* 1. **Сценарий диалога.**
     1. **Общая схема диалога**

начало; выбор типа ввода

выбор операции

повтор операции

выход

* + 1. **Описание сцен диалога.**

Приведем пример диалога.

**Program:** Добрый день!

Ввод данных:

1. Из клавиатуры
2. Из файла
3. Выход

**User:** 1

**Program:** Введите первое число

**User:** 11111

**Program:** Введите второе число

**User:** 22222

**Program:** Введите требуемую операцию:

1) Сравнение ('>' '<' '=')

2) Сложение ('+')

3) Вычитание('-')

4) Умножение ('\*')

0) Выход

**User:** 3

**Program:** Разность = -11 111

**Program:** Выберите пункт:

1) Выбрать другую операцию

2) Ввести другие числа

0) Выход

**User:** 0

**Program:** До свидания!

1. **Алгоритм работы программы и описание функций.**
   1. **Алгоритм работы программы.**

Программа выполняет следующую последовательность действий.

1. Считывает строки с экрана/файла и переводит их в целочисленные массивы.
2. Выполняет задание по выбору пользователя:

Сравнение.

Сложение.

Вычитание.

Умножение.

1. Выводит результат на экран.

Подробно распишем алгоритм работы каждого задания.

Заметим, что мы «записываем» строку при считывании в массив в обратном порядке и в каждом элементе массива храним по DIGIT цифр.

Пример: пусть ввели строку 123456789. При DIGIT = 3 она будет храниться таким образом:

|  |  |  |
| --- | --- | --- |
| a[0] | a[1] | a[2] |
| 789 | 456 | 123 |

* + 1. **Сравнение**.

Передадим длины массивов len1 и len2, и указатели на массивы arr1 и arr2.

Найдем максимальную длину len\_max

Считаем, что числа равны.

Если len1>len2

то первое число больше;

Иначе если len1<len2

то второе число больше;

Иначе

Цикл( 0 <= i < len\_max; шаг +1)

Если arr1[i] > arr2[i]

то первое число больше

Если arr1[i] < arr2[i]

то второе число больше

* + 1. **Сумма**.

Известны BASE – основание с.с. и DIGIT – кол-во разрядов эл-та массива.

Передадим длины массивов len1 и len2, и указатели на массивы arr1 и arr2.

Найдем максимальную длину len\_max;

Зададим r = 0 – остаток;

Цикл (от i = 0 до i < len\_max или r != 0; шаг +1 )

arr\_sum[i] = arr1[i] + arr2[i]+r;

Если arr\_sum[i] >= BASE

То

arr\_sum[i] = arr\_sum[i] – BASE;

r = 1;

Иначе

r = 0;

Если последний элемент массива не равен нулю

То увеличиваем размер массива на единицу;

Выводим результат.

* + 1. **Разность.**

*Данная функция будет вычитать из большего числа меньшее. Чтобы она работала и для вычитания из меньшего большее, мы зададим дополнительное условие при включении функции в main. Условно: Если a > b, то решаем a – b. Если a < b, то решаем b – a и выводим “-”(b - a).*

Известны BASE – основание с.с. и DIGIT – кол-во разрядов эл-та массива.

Передадим длины массивов len1 и len2, и указатели на массивы arr1 и arr2.

Найдем максимальную длину len\_max

Цикл ( от i = 0 пока i < len\_max; шаг +1)

Если i < len\_max - 1

То

arr[i+1] = arr[i+1] – 1;

arr\_def[i] = arr\_def[i] + arr1 + BASE;

Иначе

arr\_def[i] = arr\_def[i] + arr1;

arr\_def[i] = arr\_def[i] – arr2[i];

Если arr\_def[i]/BASE > 0

То

arr\_def[i+1] = arr\_def[i+1] + 1;

arr\_def[i] = arr\_def[i] % BASE;

Пока последний элемент результирущего массива равен нулю

Уменьшаем длину массива;

Выводим результат.

* + 1. **Произведение**.

Известны BASE – основание с.с. и DIGIT – кол-во разрядов эл-та массива.

Передадим длины массивов len1 и len2, и указатели на массивы arr1 и arr2.

Частный случай, если arr1 или arr2 = 0, то произведение равно нулю;

Найдем len\_max = len1+len2;

Цикл (от i = 0 пока i < len1; шаг +1)

r = 0; - остаток

Цикл (от j = 0 пока j < len2 или r != 0; шаг +1)

arr\_mult[i+j] = arr1[i]\*arr2[j] + r;

r = arr\_mult[i+j]/BASE;

arr\_mult[i+j] = arr\_mult[i+j] – r\*BASE;

Пока len\_max > 0 и arr\_mult[len\_max] = 0

len\_max = len\_max – 1;

len\_max = len\_max +1;

* 1. **Описание функций.**

Программа использует следующие функции:

* + 1. **Ввод строки из файла**

int Read\_Arr(std::istream &in, int \*arr);

**Назначение:**

По желанию пользователя считывает строку из файла или с экрана и записывает ее в массив так, как указано в пункте 5.1. Определяет размер полученного массива.

**Параметры:**

in – поток ввода строки

\*arr – указатель на начало массива

**Возвращаемое значение:**

len – длина массива

* + 1. **Вывод результата**

void Write\_Arr(ostream &out, const int \*arr, int len)

**Назначение:**

Вывод результата на экран и в файл.

**Параметры**:

out – поток вывода результата

\*arr – константный указатель на начало массива

len – длина массива

**Внешний эффект:**

Вывод на экран и в файл.

* + 1. **Функция сравнения длинных чисел.**

int Comparison(const int \*arr1, const int \*arr2, int len1, int len2)

**Назначение:**

Функция сравнивает 2 длинных числа и записывает результат работы в flag.

**Параметры:**

\*arr1 – константный указатель на первый массив

\*arr2 – константный указатель на второй массив

len1 – длина первого массива

len2 – длина второго массива

**Возвращаемое значение:**

flag – результат работы программы, указывает отношение длинных чисел

* + 1. **Функция нахождения суммы длинных чисел**

void Sum(const int \*arr1, const int \*arr2, int len1, int len2, ofstream& outfile)

**Назначение**:

Функция находит сумму двух длинных чисел

**Параметры:**

\*arr1 – константный указатель на первый массив

\*arr2 – константный указатель на второй массив

len1 – длина первого массива

len2 – длина второго массива

outfile – поток вывода

**Внешний эффект:**

Вывод на экран и в файл.

* + 1. **Функция нахождения разности длинных чисел**

void Subtraction(const int \*arr1, const int \*arr2, int len1, int len2, ofstream& outfile)

**Назначение**:

Функция находит разность двух длинных чисел

**Параметры:**

\*arr1 – константный указатель на первый массив

\*arr2 – константный указатель на второй массив

len1 – длина первого массива

len2 – длина второго массива

outfile – поток вывода

**Внешний эффект:**

Вывод на экран и в файл.

* + 1. **Функция нахождения произведения длинных чисел**

void Multiplication(const int \*arr1, const int \*arr2, int len1, int len2, ofstream& outfile)

**Назначение**:

Функция находит произведение двух длинных чисел

**Параметры:**

\*arr1 – константный указатель на первый массив

\*arr2 – константный указатель на второй массив

len1 – длина первого массива

len2 – длина второго массива

outfile – поток вывода

**Внешний эффект:**

Вывод на экран и в файл.

1. **Текст программы.**

**BigInt.h**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

АВТОР: БЫКОВ ИЛЬЯ, 6383

ПРОГРАММА: НЕКОТОРЫЕ ОПЕРАЦИИ НАД "ДЛИННЫМИ" ЧИСЛАМИ

НАЗНАЧЕНИЕ: ДАННАЯ ПРОГРАММА УМЕЕТ СРАВНИВАТЬ, СКЛАДЫВАТЬ, ВЫЧИТАТЬ И УМНОЖАТЬ ДЛИННЫЕ ЧИСЛА

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef BIGINT\_H\_INCLUDED

#define BIGINT\_H\_INCLUDED

#include <iostream> // cin , cout etc

#include <fstream> // ifstream , outstream etc

#include <cstring> // substr

#include <stdlib.h> // atoi

#include <iomanip> // setw

using namespace std;

// максимальный размер длинного числв

extern const int MAX\_SIZE;

// система счисления длинного числа

extern const int BASE;

// количество цифр в одном разряде

extern const int DIGIT;

// функция чтения из файла/экрана

// считываем строку и переводим в числово массив

// передает указатель на массив arr

// возвращает длинну массива

int Read\_Arr(std::istream &in, int \*arr);

// функция вывода результата в файл/на экран

// передает костантный указатель на массив arr

// предусловие; длина массива len

void Write\_Arr(std::ostream &out, const int \*arr, int len);

// функция нахождения суммы длинных чисел

// передает костантные указатели на исходные массивы arr1 и arr2

// предусловие; длины массивов len1 и len2

void Find\_Sum(const int \*arr1, const int \*arr2, int len1, int len2, std::ofstream& outfile);

// функция сравнения длинных чисел

// передает костантные указатели на исходные массивы arr1 и arr2

// предусловие; длины массивов len1 и len2

// возвращает flag - результат работы функции. (0: ==, 1: >, 2: <)

int Comparison(const int \*arr1, const int \*arr2, int len1, int len2);

// функция нахождения разности длинных чисел

// Вычитаем из большего меньшее.

// передает костантные указатели на исходные массивы arr1 и arr2

// предусловие; длины массивов len1 и len2

void Subtraction(const int \*arr1, const int \*arr2, int len1, int len2, std::ofstream& outfile);

// Функция нахождения произведения длинных чисел

// передает костантные указатели на исходные массивы arr1 и arr2

// предусловие; длины массивов len1 и len2

void Multiplication(const int \*arr1, const int \*arr2, int len1, int len2, std::ofstream& outfile);

#endif // BIGINT\_H\_INCLUDED

**BigInt.cpp**

#include "BigInt.h"

extern const int MAX\_SIZE = 1000;

extern const int BASE = 1000;

extern const int DIGIT = 3;

int Read\_Arr(istream &in, int \*arr)

{

string str;

int len = 0;

in >> str;

for (int i = str.size() - 1; i >= 0; i -= DIGIT)

{

int start = i - DIGIT + 1;

if (start < 0)

start = 0;

string tmp = str.substr(start, i - start + 1);

arr[len++] = atoi(tmp.c\_str());

}

return len;

}

void Write\_Arr(ostream &out, const int \*arr, int len)

{

out << arr[len-1] << " ";

for (int i = len - 2; i >= 0; --i)

out << setw(DIGIT) << setfill('0') << arr[i] << " ";

out << endl;

return;

}

void Find\_Sum(const int \*arr1, const int \*arr2, int len1, int len2, ofstream& outfile) // сложение

{

int len\_max\_sum = max(len1, len2);

int i = 0;

int r = 0;

int arr\_sum[len\_max\_sum]; // объявим массив суммы

for(i = 0; i < len\_max\_sum + 1; i++)

arr\_sum[i] = 0;

for(i = 0; i < len\_max\_sum | r; i++)

{

arr\_sum[i] = arr1[i] + arr2[i] + r;

if (arr\_sum[i] >= BASE)

{

arr\_sum[i] -= BASE;

r = 1;

}

else

r = 0;

}

if (arr\_sum[len\_max\_sum])

len\_max\_sum++;

cout << "Сумма = ";

outfile << "Сумма = ";

Write\_Arr(cout, arr\_sum, len\_max\_sum);

Write\_Arr(outfile, arr\_sum, len\_max\_sum);

return;

}

int Comparison(const int \*arr1, const int \*arr2, int len1, int len2) // сравнение

{

int len\_max = max(len1, len2);

int flag = 0; // числа одинаковой длины

if (len1 > len2)

{

flag = 1; // первое число длиннее

return flag;

}

else if (len2 > len1)

{

flag = 2; // второе число длиннее

return flag;

}

else

for(int i = 0; i < len\_max; i++) // если длины равны, сравниваем поразрядно

{

if (arr1[i] > arr2[i])

{

flag = 1;

return flag;

}

if (arr2[i] > arr1[i])

{

flag = 2;

return flag;

}

}

}

void Subtraction(const int \*arr1, const int \*arr2, int len1, int len2, ofstream& outfile) //вычитание

{

int len\_max = max(len1, len2);

int i = 0;

int arr\_def[len\_max]; // объявим массив разности

int arr1\_def[len1];

for (i = 0; i < len\_max; i++)

arr\_def[i] = 0;

for(i = 0; i < len1; i++)

arr1\_def[i] = 0;

for(i = 0; i < len1; i++)

arr1\_def[i] = arr1[i];

for(i = 0; i <= (len\_max - 1); i++)

{

if (i < (len\_max - 1)) // если разряд не первый

{

arr1\_def[i+1]--; // то в следующем разряде займем 1

arr\_def[i] += (BASE + arr1\_def[i]); // запишем сумму значения разряда большего числа

}

else // если - первый

arr\_def[i] += arr1\_def[i]; // cуммируем значение разряда большего числа

arr\_def[i] -= arr2[i]; // вычитаем значение разряда меньшего числа

if (arr\_def[i] / BASE > 0) // если значение разряда больше 10

{

arr\_def[i + 1]++; // след. разряд +1

arr\_def[i] %= BASE; // отсекаем ее

}

}

while (arr\_def[len\_max - 1] == 0)

len\_max--;

Write\_Arr(cout, arr\_def, len\_max);

Write\_Arr(outfile, arr\_def, len\_max);

return;

}

void Multiplication(const int \*arr1, const int \*arr2, int len1, int len2, ofstream& outfile)

{

if (((len1 == 1) && (arr1[0] == 0)) || ((len2 == 1) && (arr2[0] == 0)))

{

cout << "Произведение = 0" << endl;

return;

}

int len\_mult = len1 + len2 + 1;

int arr\_mult[len\_mult];

for (int i = 0; i < len\_mult; i++)

arr\_mult[i] = 0;

/\*for (int i = 0; i < len1; i++) // 1-ый алгоритм

{

for (int j = 0; j < len2; j++)

arr\_mult[i + j] += arr1[i] \* arr2[j];

}

for (int i = 0; i < len\_mult; i++)

{

arr\_mult[i + 1] += arr\_mult[i] / BASE;

arr\_mult[i] %= BASE;

}

while (arr\_mult[len\_mult - 1] == 0)

len\_mult--;\*/

for (int i = 0; i < len1; i++) // 2-ой алгоритм. Быстрее, т.к. не используем %

{

int r = 0;

for (int j = 0; j < len2 | r; j++)

{

arr\_mult[i + j] += arr1[i] \* arr2[j] + r;

r = arr\_mult[i + j] / BASE;

arr\_mult[i + j] -= r \* BASE;

}

}

int len\_mult\_2 = len1 + len2;

while (len\_mult\_2 > 0 && !arr\_mult[len\_mult\_2])

len\_mult\_2--;

len\_mult = len\_mult\_2 + 1;

cout << "Произведение = ";

outfile << "Произведение = ";

Write\_Arr(cout, arr\_mult, len\_mult);

Write\_Arr(outfile, arr\_mult, len\_mult);

return;

}

**main.cpp**

#include "BigInt.h"

int main()

{

setlocale(LC\_CTYPE , "rus");

ifstream infile1("fscanf1.txt");

ifstream infile2("fscanf2.txt");

ofstream outfile("fprintf.txt");

if (!infile1)

{

cout << "Не удалось открыть первый файл ввода!\n";

return 1;

}

if (!infile2)

{

cout << "Не удалось открыть второй файл ввода!\n";

return 1;

}

if (!outfile)

{

cout << "Не удалось открыть файл вывода!\n";

return 1;

}

int arr1[MAX\_SIZE]; // заполним исходные массивы нулями

for (int i = 0; i < MAX\_SIZE; i++)

arr1[i] = 0;

int arr2[MAX\_SIZE];

for (int i = 0; i < MAX\_SIZE; i++)

arr2[i] = 0;

int choise, len1, len2;

bool flag\_choise = false;

cout << "Добрый день!" << endl;

while (flag\_choise == false)

{

cout << "Ввод данных:\n1) C клавиатуры\n2) Из файла\n0) Выход" << endl;

cin >> choise;

if (choise == 2)

{

len1 = Read\_Arr(infile1, arr1);

len2 = Read\_Arr(infile2, arr2);

cout << "В первом массиве считано " << len1 << " элементов:";

Write\_Arr(cout, arr1, len1);

outfile << "В первом массиве считано " << len1 << "элементов:";

Write\_Arr(outfile, arr1, len1);

cout << "В втором массиве считано " << len2 << "элементов: ";

Write\_Arr(cout, arr2, len2);

outfile << "В втором массиве считано " << len2 << " элементов: ";

Write\_Arr(outfile, arr2, len2);

flag\_choise = true;

}

else if (choise == 1)

{

cout << "Введите первое число: " << endl;

len1 = Read\_Arr(cin, arr1);

cout << "Введите второе число: " << endl;

len2 = Read\_Arr(cin, arr2);

outfile << "В первом массиве считано " << len1 << " элементов: ";

Write\_Arr(outfile, arr1, len1);

outfile << "В втором массиве считано " << len2 << " элементов: ";

Write\_Arr(outfile, arr2, len2);

flag\_choise = true;

}

else if (choise == 0)

{

cout << "До свидания!" << endl;

return 0;

}

else

{

cout << "Введены неправильные данные. Попробуйте еще раз." << endl << endl;

}

if (choise == 1 || choise == 2)

{

int len\_max\_sum = max(len1, len2) + 1;

if((len1 == 0) or (len2 == 0))

{

cout << "Входной(-ые) файл(-ы) пуст(-ы)!";

outfile << "Входной(-ые) файл(-ы) пуст(-ы)!";

return 1;

}

int flag = Comparison(arr1, arr2, len1, len2);

int operation\_choise;

bool flag\_operation\_choise = false;

bool flag\_end = false;

int end\_choise;

while (flag\_operation\_choise == false)

{

cout << "Введите требуемую операцию:\n1) Сравнение ('>' '<' '=')\n2) Сложение ('+')\n3) Вычитание('-')\n4) Умножение ('\*')\n0) Выход" << endl;

cin >> operation\_choise;

if (operation\_choise == 1)

{

if (flag == 0)

{

cout << "Числа равны (=)" << endl;

outfile << "Числа равны (=)" << endl;

}

if (flag == 1)

{

cout << "Первое число больше (>)" << endl;

outfile << "Первое число больше (>)" << endl;

}

if (flag == 2)

{

cout << "Первое число меньше (<)" << endl;

outfile << "Первое число меньше (<)" << endl;

}

flag\_operation\_choise = true;

}

else if (operation\_choise == 2)

{

Find\_Sum(arr1, arr2, len1, len2, outfile);

flag\_operation\_choise = true;

}

else if (operation\_choise == 3)

{

if (flag == 0)

{

cout << "Разность = 0" << endl;

outfile << "Разность = 0" << endl;

}

if (flag == 1)

{

cout << "Разность = ";

outfile << "Разность = ";

Subtraction(arr1, arr2, len1, len2, outfile);

}

if (flag == 2)

{

cout << "Разность = -";

outfile << "Разность = -";

Subtraction(arr2, arr1, len2, len2, outfile);

}

flag\_operation\_choise = true;

}

else if (operation\_choise == 4)

{

Multiplication(arr1, arr2, len1, len2, outfile);

flag\_operation\_choise = true;

}

else if (operation\_choise == 0)

{

cout << "До свидания!" << endl;

return 0;

}

else

{

cout << "Введена неправильная команда. Попробуйте еще раз." << endl;

flag\_operation\_choise = false;

continue;

}

cout << endl << "Выберите пункт:\n1) Выбрать другую операцию\n2) Ввести другие числа\n0) Выход" << endl;

cin >> end\_choise;

if (end\_choise == 1)

flag\_operation\_choise = false;

else if (end\_choise == 2)

{

flag\_choise = false;

continue;

}

else

{

cout << "До свидания!";

return 0;

}

}

}

}

return 0;

}

1. **Тестирование.**

|  |  |  |  |
| --- | --- | --- | --- |
| № | 1 | 2 | 3 |
| число 1 | 100000 | 0 | 999999 |
| число 2 | 0 | 100000 | 1 |
| сравнение | > | < | > |
| сумма | 100 000 | 100 000 | 1 000 000 |
| разность | 100 000 | -100 000 | 999 998 |
| произведение | 0 | 0 | 999 999 |

|  |  |
| --- | --- |
| № | 4 |
| число 1 | 1234567891011 |
| число 2 | 1234567891011 |
| сравнение | = |
| сумма | 2 469 135 782 022 |
| разность | 0 |
| произведение | 1 524 157 877 515 348 374 602 121 |

|  |  |
| --- | --- |
| № | 5 |
| число 1 | 1000 |
| число 2 |  |
| сравнение | Вывод: «Входной(-ые) файл(-ы) пуст(-ы)!» |
| сумма |
| разность |
| произведение |

|  |  |
| --- | --- |
| № | 6 |
| число 1 | 999999999999999999999999999999999999999 |
| число 2 | 11111111111111111111111111111111111111111111 |
| сравнение | < |
| сумма | 11 112 111 111 111 111 111 111 111 111 111 111 111 111 110 |
| разность | -11 110 111 111 111 111 111 111 111 111 111 111 111 111 112 |
| произведение | 11 111 111 111 111 111 111 111 111 111 111 111 111 099 999 888 888 888 888 888 888 888 888 888 888 888 888 889 |

1. **Вывод**

По ходу выполнения курсовой работы мною были созданы алгоритмы по нахождению суммы, разности, произведения длинных чисел, а так же их сравнения. Я научился преобразовывать строки в массив, добавляя в каждый элемент массива нужное количество символов. Так же ознакомился с библиотекой fstream, которая упростила и расширила мои возможности ввода и вывода данных. Так же я ознакомился с оформлением заголовочных файлов.