

Федеральное государственное автономное образовательное учреждение высшего  
образования

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

**Факультет информатики, математики и компьютерных наук**

**КУРСОВАЯ РАБОТА**

**Перемножение целых чисел «столбиком»**

по направлению подготовки 09.03.04 "Программная инженерия"

образовательная программа «Программная инженерия»

Выполнила:  
Студентка группы 18ПИ-2  
Мартынюк Олеся Олеговна

Руководитель:  
Лупанова Елена Александровна

Нижний Новгород 2019

## Содержание

Введение .....	2
Постановка задачи .....	3
Описание задачи .....	3
Входные и выходные данные .....	3
Инструменты .....	3
Пример .....	3
Обзор существующих решений.....	4
Примеры существующих решений .....	4
Проблемы в существующих решениях.....	5
Описание решения .....	6
Шаги и логика алгоритма.....	6
Описание структуры проекта .....	7
Тестирование алгоритма .....	8
Заключение.....	10
Список используемых источников .....	11
Приложение.....	12

## **Введение**

Умножение — одно из основных четырех математических действий. Умножать можно целые, рациональные, комплексные и вещественные числа.

Алгоритмы умножения существуют с момента появления десятичной системы. На данный момент широкой популярностью пользуются следующие алгоритмы:

1. Умножение «столбиком»;
2. Умножение Карацубы;
3. Японский (китайский) способ;
4. Итальянский способ.

В зависимости от размера чисел применяются разные алгоритмы их перемножения.

В данной курсовой работе будет подробнее рассмотрен алгоритм умножения чисел «столбиком». Данный метод подразумевает последовательное умножение меньшего числа на все цифры большего числа с соблюдением смещения в зависимости от разряда, а также дальнейшее сложение промежуточных сумм.

Рассматриваемая тема актуальна, так как алгоритм умножения «столбиком» имеет самую широкую популярность и, вместе с тем, простоту понимания: именно поэтому изучение данного метода включено в программу начальной школы.

## Постановка задачи

### Описание задачи

Задача данной курсовой работы – реализовать на языке программирования Си алгоритм умножения «столбиком».

Алгоритм должен выводить подробное решение примера, то есть выводить промежуточные произведения, соблюдая смещения относительно разряда. В алгоритме должно быть предусмотрено смещение «круглого» числа вправо, если длина его «некруглой» части, то есть части без нулей, меньше длины «некруглой» части другого числа. Также в выводе промежуточных сумм будет опускаться умножение на ноль.

### Входные и выходные данные

Входными данными для алгоритма являются целые числа типа данных `integer`, то есть все числа в диапазоне от -2 147 483 648 до +2 147 483 647.

Выходными данными – промежуточными вычислениями и итоговым произведением, – являются целые числа типа данных `long long`, то есть все числа в диапазоне от -9 223 372 036 854 775 808 до +9 223 372 036 854 775 807. Вывод вычислений происходит в окне консоли.

Результатом решения задачи являются выведенные в столбик входные числа, промежуточные вычисления и итоговое произведение под чертой.

### Инструменты

В ходе реализации алгоритма умножения чисел «столбиком» использовалась операционная система Windows 10 version 1803 и интегрированная среда разработки программного обеспечения Visual Studio 2017.

### Пример

Примерный вывод входных чисел, промежуточных вычислений и произведения:

$$\begin{array}{r} 2564 \\ * \quad 3500 \\ \hline 12820 \\ + \\ \quad 7692 \\ \hline 8974000 \end{array}$$

[illegible]

## Проблемы в существующих решениях

Несмотря на корректные вычисления, в данных калькуляторах существует несколько проблем, которые исправлены в алгоритме, разработанном в ходе курсовой работы.

В первом калькуляторе использован нестандартный вывод промежуточных вычислений с использованием незначащих нулей в старших и младших разрядах числа. Также в калькуляторе, размещенном на сайте, не соблюдены отступы в написании промежуточных вычислений и в записи вводимых чисел, когда первые справа цифры, отличные от нуля, располагаются друг под другом. Кроме того, в вычислениях выводится произведение, равное нулю. В случае, когда итоговое произведение отлично от нуля, такие расчеты целесообразно опустить для компактности записи.

В калькуляторе “Long Division Calc” не соблюдается верная запись чисел. Необходимо записывать сверху то число, чья длина, начинающаяся от первой справа цифры, отличной от нуля, больше длины другого числа, например:

$$\begin{array}{r} 5340 \\ * \quad 43000 \\ \hline 1602 \\ 2136 \\ \hline 229620000 . \end{array}$$

## Описание решения

### Шаги и логика алгоритма

Алгоритм умножения двух чисел в столбик, если их итоговое произведение отлично от нуля:

1. Ввод первого (firstNum) и второго (secondNum) чисел.
2. Вычисление произведения обоих чисел (itog).
3. Подсчет количества нулей в обоих числах, длины каждого числа и длины каждого числа без нулей.
4. Число с большей длиной части без нулей становится firstNum и выводится первым, число с меньшей – secondNum, выводится вторым.
5. Вывод чисел происходит с отступом слева, который равен (20- (длина числа) – количество нулей в другом числе), и с отступом справа, который равен количеству нулей в другом числе.
6. Вывод черты под числами.
7. Для промежуточных вычислений используется модули чисел без их нулевой части. Например: у числа -254600 используется 2546, а у числа 100328 используется всё число, то есть 100328.
8. Создаем дубликаты этих модулей. Они будут нужны в промежуточных вычислениях.
9. Если модуль второго числа без нулевой части содержит больше одного знака, то начинаются промежуточные вычисления.
10. С помощью цикла, где переменная *i* (разряд второго числа) «идёт» от 0 до числа, равного длине второго числа, повторяются следующие действия:
  - 10.1. Остаток от деления дубликата второго числа на десять умножается на первое число.
  - 10.2. Если полученное произведение больше нуля, то оно выводится с отступом справа, который равен сумме всех нулей и значения переменной *i*.
  - 10.3. Когда переменная *i* равна нулю, выводится знак сложения «+» .
  - 10.4. Дубликат второго числа делится на 10.
11. Вывод черты под числами.
12. Вывод итогового произведения.

Необходимо отдельно написать ход алгоритма, если итоговое произведение равно нулю:

1. Ввод первого (firstNum) и второго (secondNum) чисел.
2. Вычисление произведения обоих чисел (itog).

3. Больше по модулю число становится первым (firstNum), меньше – вторым (secondNum).
4. Вывод чисел происходит с отступом слева, который равен (20- длина числа).
5. Вывод черты под числами.
6. Вывод итогового произведения.

## Описание структуры проекта

Данный проект состоит из трёх файлов:

1. Header.h Содержит прототипы функций и глобальных переменных, а также их описания в комментариях.
2. Source.cpp содержит следующие функции:
  - 2.1. `void change (int *valueFirst, int *valueSecond)` – функция, меняющая местами два значения.
  - 2.2. `int countZeros(int value)` – подсчет количества нулей в числе.
  - 2.3. `int searchPosition(long long int value)` – подсчет количества разрядов в числе.
  - 2.4. `void output(long long int value, int a)` – вывод числа с отступом a.
  - 2.5. `void makingCollumn(void)` – печать вводимых чисел в столбик.
  - 2.6. `void mathFunction(void)` – расчет и вывод промежуточных вычислений.
3. Multiplication.cpp содержит главное тело программы и сопроводительные комментарии.

В целом, в реализации алгоритма были использованы стандартные функции и операторы языка Си, такие как цикл for и оператора выбора if else. Также в функции `void change (int *valueFirst, int *valueSecond)` были использованы указатели на переменные.

В основном во всем проекте были использованы глобальные переменные вместо константных. Они были задействованы, так как в данном небольшом по объёму проекте они оказались удобнее в использовании.

Также, для корректного вывода произведения было использование преобразование типов:

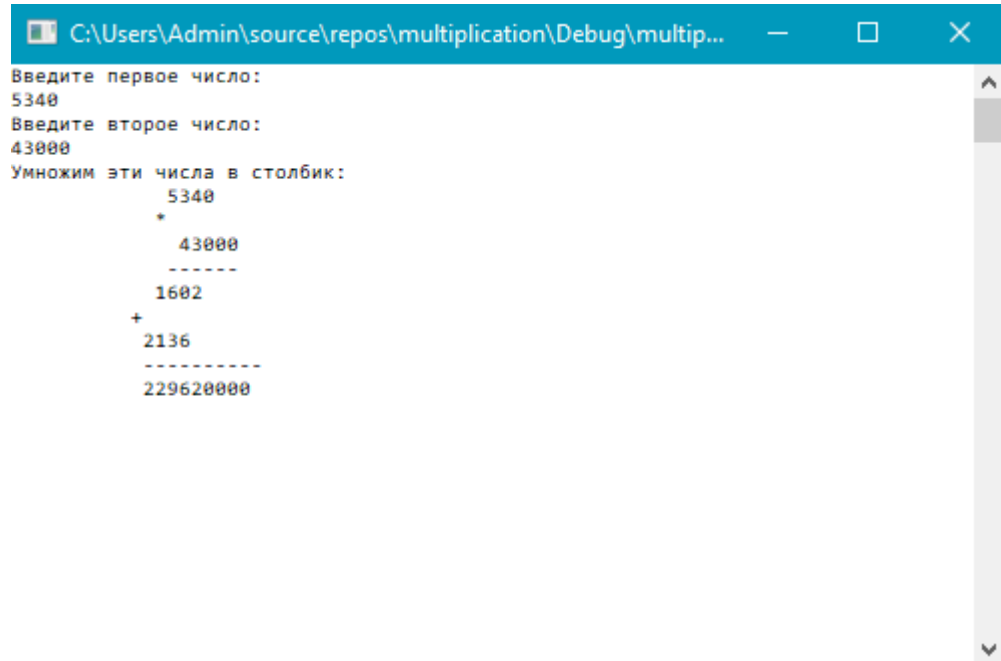
```
itog = (long long)firstNum * (long long)secondNum; //Считаем произведение заранее.
```



## Тестирование алгоритма

Данный алгоритм был протестирован на различных входных данных:

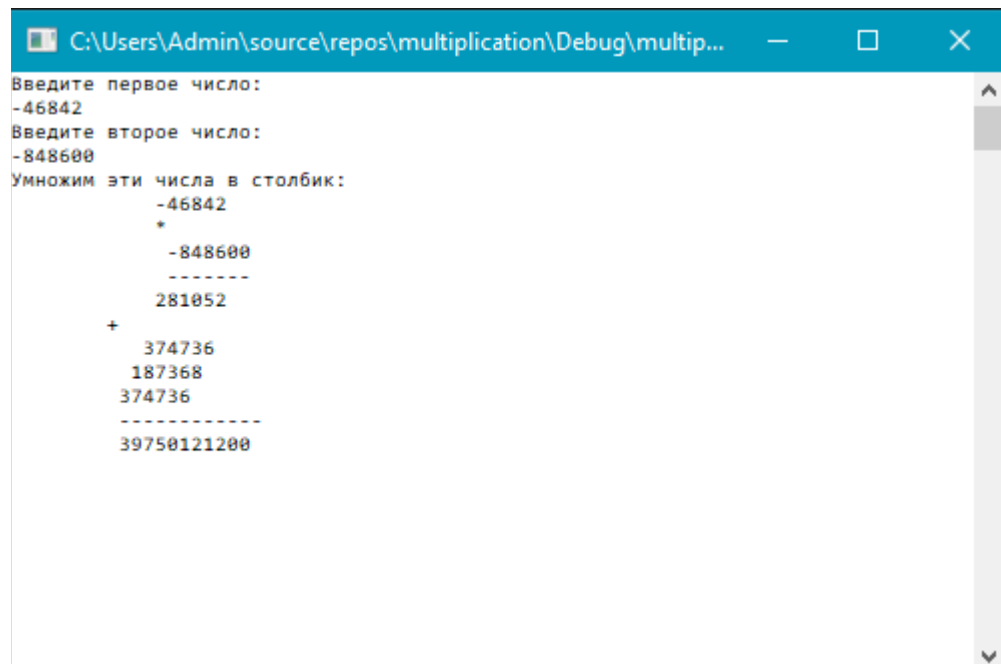
1. Используя пример из главы «Постановка задачи», протестируем алгоритм:



```
C:\Users\Admin\source\repos\multiplication\Debug\multip...
Введите первое число:
5340
Введите второе число:
43000
Умножим эти числа в столбик:
      5340
    * 43000
    -----
      1602
    +
    2136
    -----
  229620000
```

Как видно на скриншоте, вычисления и запись чисел верны.

2. Тестирование алгоритма на отрицательных числах:



```
C:\Users\Admin\source\repos\multiplication\Debug\multip...
Введите первое число:
-46842
Введите второе число:
-848600
Умножим эти числа в столбик:
      -46842
    * -848600
    -----
      281052
    +
    374736
    187368
    374736
    -----
  39750121200
```

Вычисления и запись чисел верны.

3. Тестирование алгоритма на отрицательном и положительном числах:

```
C:\Users\Admin\source\repos\multiplication\Debug\multip...
Введите первое число:
15446468
Введите второе число:
-4468
Умножим эти числа в столбик:
      15446468
        *
        -4468
        -----
      123571744
    +
      92678808
      61785872
      61785872
      -----
     -69014819024
```

Вычисления и запись чисел верны.

#### 4. Тестирование алгоритма на умножение на ноль:

```
C:\Users\Admin\source\repos\multiplication\Debug\multip...
Введите первое число:
6846684
Введите второе число:
0
      6846684
        *
         0
        -----
         0
```

Вычисления и запись чисел верны.

## **Заключение**

В процессе работы над проектом мною были закреплены знания и умения программирования на языке Си. В процессе работы над реализацией алгоритма я приобрела дополнительный опыт работы с интегрированной средой разработки программного обеспечения Visual Studio 2017, с заголовочными файлами, глобальными переменными и преобразованием типов.

Данная программа может быть полезна и интересна учащимся начальных классов, так как с помощью нее ученики смогут проверять свои вычисления. Также проект может быть интересен тем, кто только начал изучать программирование.

В целом можно сказать, что моя проектная работа не несет в себе новизны в вычислительном процессе, но исправляет недочеты в записи чисел в существующих калькуляторах. Результаты проделанной работы с точки зрения поставленных задач удовлетворительны.

### Список используемых источников

1. Умножение в столбик онлайн [Электронный ресурс] URL:  
<https://www.calc.ru/umnozhenie-v-stolbik.html> (дата обращения: 26.05.2019)
2. Колюжнов В. Решение столбиком [Мобильное приложение] URL:  
<https://itunes.apple.com/ru/app/решение-столбиком/id1257528605?mt=8>  
(26.05.2018)
3. Умножение «Википедия» [Электронный ресурс] URL:  
[https://ru.wikipedia.org/wiki/Умножение#Целые\\_числа](https://ru.wikipedia.org/wiki/Умножение#Целые_числа) (дата обращения:  
25.05.2019)
4. Multiplication algorithm «Википедия» [Электронный ресурс] URL:  
[https://en.wikipedia.org/wiki/Multiplication\\_algorithm](https://en.wikipedia.org/wiki/Multiplication_algorithm) (дата обращения:  
25.05.2019)
5. Умножение натуральных чисел столбиком, примеры, решения. «Clever students» [Электронный ресурс] URL:  
[http://www.cleverstudents.ru/numbers/long\\_multiplication.html](http://www.cleverstudents.ru/numbers/long_multiplication.html) (дата  
обращения: 24.05.2019)
6. Мусина А.А. Способы умножения [Электронный ресурс] URL:  
<https://school-science.ru/5/7/34501> (дата обращения: 24.05.2019)

## Приложение

### Header.h

```
#ifndef Header_h
#define Header_h

//-----Глобальные переменные-----
int firstNum; //первое перемножаемое число
int secondNum; //второе перемножаемое число
int firstNum_twin; //необходимо для парсинга второго числа
int secondNum_twin; //необходимо для второго числа
int zeros_f = 0; //кол-во нулей в первом числе по умолчанию
int zeros_s = 0; //кол-во нулей во втором числе по умолчанию
int zeros_all = 0; //zeros_f+zeros_s
long int chislo;
long long itog;

//-----Используемые функции-----
void change(int *valueFirst, int *valueSecond);
int searchPosition(long long int value);
void output(long long int value, int sign);
void makingCollumn(void);
void mathFunction(void);
int countZeros(int value);
#endif
```

### Source.cpp

```
#include "Header.h"
#include "pch.h"
#include <stdio.h>
#include <math.h>

//-----Глобальные переменные-----
extern int firstNum;
extern int secondNum;
extern int firstNum_twin;
extern int secondNum_twin;
extern int zeros_f;
extern int zeros_s;
extern int zeros_all;
extern long int chislo;
extern long long itog;

//-----Меняем местами две переменные-----
void change(int *valueFirst, int *valueSecond) {
    int change = *valueFirst;
    *valueFirst = *valueSecond;
    *valueSecond = change;
}

//-----Подсчет количества нулей в числе-----
int countZeros(int value) {
    int zeros = 0;
    if (value != 0) {
        while (value % 10 == 0) {
            zeros++;
            value = value / 10;
        }
        return zeros;
    }
}

//-----Подсчет количества разрядов в числе-----
```

```

int searchPosition(long long int value) {
    int position = 0;
    if (value == 0) { //Если второе число = 0.
        position = 1;
    }
    while (value != 0) //Если же изначально число не ноль, то делим его на 10 и считаем
число разрядов.
    {
        value = value / 10;
        position = position + 1;
    }
    return position; //Возвращаем число разрядов.
}

//-----Вывод числа с отступами-----
void output(long long int value, int a) {
    int k;
    if (value < 0) { //Поправка отступа для отрицательных чисел.
        k = 1;
    }
    else {
        k = 0;
    }
    for (int i = 0; i < 20 - searchPosition(value) - a - k; i++) { //Отступаем
необходимое количество слева.
        printf(" ");
    }
    printf("%lli\n", value);
}

//-----Печать вводимых чисел в столбик (вместе с * и ---) -----
void makingCollumn(void) {
    output(firstNum, zeros_s); // Печать первого числа.
    for (int i = 0; i < 20 - searchPosition(firstNum) - 1 - zeros_s; i++) { // Печать *.
        printf(" ");
    }
    printf("*\n");
    output(secondNum, zeros_f); // Печать второго числа.
    for (int i = 0; i < 20 - searchPosition(firstNum) - zeros_s; i++) { // Печать ---.
        printf(" ");
    }
    for (int i = 0; i < searchPosition(firstNum) + (abs(zeros_f - zeros_s)); i++) {
        printf("-");
    }
    printf("\n");
}

//-----Расчет и вывод промежуточных чисел-----
void mathFunction(void) {
    int digit; //Число в разряде второго числа.

    for (int i = 0; i < searchPosition(secondNum); i++) {
        digit = secondNum_twin % 10;
        chislo = digit * firstNum;
        if (chislo > 0) {
            output(chislo, i + zeros_all);
            if (i == 0) {
                for (int k = 0; k < 20 - searchPosition(itog) - 1; k++) { //
Печать +.
                    printf(" ");
                }
                printf("+\n");
            }
        }
        secondNum_twin = secondNum_twin / 10;
    }
}

```

```

    }

}

```

## Multiplication.cpp

```

#include "pch.h"
#include "Header.h"
#include <stdio.h>
#include <Windows.h>
#include <conio.h>
#include <locale.h>
#include <math.h>
#include <limits.h>

//-----Глобальные переменные-----
extern int firstNum;
extern int secondNum;
extern int firstNum_twin;
extern int secondNum_twin;
extern int zeros_f;
extern int zeros_s;
extern int zeros_all;
extern long int chislo;
extern long long itog;

int main() {
    system("color F0");
    setlocale(LC_ALL, "Rus");
    printf("Введите первое число: \n");
    scanf_s("%d", &firstNum);
    printf("Введите второе число: \n");
    scanf_s("%d", &secondNum);
    itog = (long long)firstNum * (long long)secondNum; //Считаем произведение заранее.
    Далее будем "рубить" числа:
    if (itog != 0) {
        zeros_f = countZeros(firstNum); //Считаем кол-во нулей в первом числе.
        zeros_s = countZeros(secondNum); //Считаем кол-во нулей во втором числе.
        zeros_all = zeros_f + zeros_s; //Общее число нулей.
        firstNum_twin = firstNum / pow(10, zeros_f);
        secondNum_twin = secondNum / pow(10, zeros_s);
        if (abs(firstNum_twin) < abs(secondNum_twin)) { //Меняем местами большее по
модулю число с меньшим и их копии.
            change(&firstNum, &secondNum);
            change(&firstNum_twin, &secondNum_twin);
            change(&zeros_f, &zeros_s);
        }
        printf("Умножим эти числа в столбик: \n"); //пожалуйста, поставьте 10
makingCollumn();
        firstNum_twin = abs(firstNum_twin);
        secondNum_twin = abs(secondNum_twin);
        secondNum = secondNum_twin; //Используем модули чисел для промежуточного
умножения.
        firstNum = firstNum_twin;
        if (searchPosition(secondNum) > 1) {
            mathFunction();
            for (int i = 0; i < 20 - searchPosition(itog); i++) {
                printf(" ");
            }
            for (int i = 0; i < searchPosition(itog) + 1; i++) {
                printf("-");
            }
            printf("\n");
        }
        output(itog, 0); //Вывод произведения.
    }
}

```

```

else {
    if (abs(firstNum) < abs(secondNum)) { //Меняем местами большее по модулю число
с меньшим и их копии.
        change(&firstNum, &secondNum);
    }
    output(firstNum, 0);
    for (int i = 0; i < 20 - searchPosition(firstNum) - 1; i++) {// Печать *.
        printf(" ");
    }
    printf("*\n");
    output(secondNum, zeros_f);
    for (int i = 0; i < 20 - searchPosition(firstNum) - zeros_s; i++) {// Печать -
--.
        printf(" ");
    }
    for (int i = 0; i < searchPosition(firstNum); i++) {
        printf("-");
    }
    printf("\n");
    output(0, 0);
}
_getch();
}

```