



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет имени  
Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

## ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1 ПО ДИСЦИПЛИНЕ: ТИПЫ И СТРУКТУРЫ ДАННЫХ

*<Длинная арифметика>*

Студент *<Ермаков И.Г>*

Группа *<ИУ7-32Б >*

Название предприятия **НУК ИУ МГТУ им. Н. Э. Баумана**

Студент \_\_\_\_\_ **<Фамилия ИО>**

Преподаватель \_\_\_\_\_ **<Фамилия ИО>**

2024

## Оглавление

Условие задачи.....	3
Техническое задание.....	3
Структура данных.....	5
Алгоритм.....	7
Набор тестов.....	7

Смоделировать операцию умножения целого числа длиной до 40 десятичных цифр на действительное число в форме  $\pm m.n \text{ E } \pm K$ , где суммарная длина мантииссы ( $m+n$ ) - до 30 значащих цифр, а величина порядка  $K$  - до 5 цифр. Результат выдать в форме  $\pm 0.m1 \text{ E } \pm K1$ , где  $m1$  - до 40 значащих цифр, а  $K1$  - до 5 цифр.

## **Техническое Задание**

### **Входные данные**

Целое число до 40 цифр, знак опционален, если знак не был введен, по умолчанию считается что введено неотрицательное число

Вещественное число не обязательно вводится в нормализованном виде количество значащих цифр в мантиссе не превышает 30. Ведущие нули (не значащие – до точки и/или до первой цифры) в расчете длины числа не учитываются. Значащие нули после точки учитываются при подсчете длины числа. Десятичное число может представляться без точки: 123 (как целое). При наличии точки в числе возможны следующие варианты его представления: .00025, +123001., – 123.456. Также допускается представление числа в экспоненциальной форме: 1234567 E –20, 1234567E20 или 123.4567E23. Длина порядка  $\leq 5$

### **Выходные данные**

Нормализованное вещественное число выводится в форме  $\pm 0.m1 \text{ E } \pm K1$ , где  $m1$  - до 40 значащих цифр, а  $K1$  - до 5 цифр.

### **Обращение к программе**

Программа запускается по команде ./app.exe, далее вводится целое число, затем вещественное, после ввода данных программа выведет нормализованные значения, а затем выведет результат умножения.

### **Аварийные ситуации**

1) Длина целого числа больше 40

- 2)Ошибка считывания целого числа через терминал (результат fgets = NULL)
- 3)В целом числе(без учета знака) встретилось не число
- 4)Ошибка считывания вещественного числа через терминал (результат fgets = NULL)
- 5)Длина мантиссы вещественного числа больше 30
- 6)Введенное вещественное число не соответствует указанному формату
- 7)Длина порядка перед нормализацией введенного вещественного числа больше 99999 по модулю
- 8)Длина порядка после нормализации введенного вещественного числа больше 99999 по модулю
- 9)При перемножении двух чисел порядок превысил 99999 по модулю

Целое число вводится с клавиатуры и записывается в массив символов, изначально больше по размеру чем нужно (максимальный размер целого числа вместе со знаком 41 символ), а в программе задается значение 50.

Целое число вводится с клавиатуры и записывается в массив символов, изначально больше по размеру чем нужно (максимальный размер целого числа вместе со знаком 48 символ), ввод в формате  $\pm m.n E \pm K$  где  $m + n \leq 30$ , + 1 символ знака, плюс буква, обозначающая что число записано в нормализованной форме, символ знака порядка и сам порядок:  $K \leq 5$ , а в программе задается значение 50.

Затем после частичной работы алгоритма распределения и приведения чисел к общему стандарту, т.е в целом числе убираются все незначащие нули, добавляется знак, если его там нет – есть структура куда записываются компоненты числа

```
typedef struct
{
    char num_sign;
    char int_value[MAX_LEN_INT_I + 1];
} int_data;
```

В поле 'num\_sign' записывается знак числа (если не был введен, по умолчанию '+')

В поле 'int\_value' записывается целое число (без незначащих нулей)

Как описывалось ранее, полю 'int\_value' указано заведомо число большее длины, описанной в условии задачи

$\text{MAX\_LEN\_INT\_I} = 50$

Аналогично ниже представлена структура данных, которая хранит приведенные к стандарту компоненты вещественного числа

```
typedef struct
{
    char num_sign;
    char mantissa[MAX_LEN_MANTISSA_I + 1];
    char exp_sign;
    char order[MAX_LEN_ORDER_I + 1];
} double_data;
```

В поле 'num\_sign' записывается знак числа (если не был введен, по умолчанию '+')

В поле 'mantissa' записывается мантисса числа (m.n) без незначащих нулей

В поле 'exp\_sign' записывается знак порядка числа (если не был введен, по умолчанию '+')

В поле 'order' записывается порядок числа (если не был указан, генерируется)

Как описывалось ранее, полям 'mantissa' и 'order' указаны заведомо числа БОльшие числа длины, описанной в условии задачи

$\text{MAX\_LEN\_MANTISSA\_I} = 40$

$\text{MAX\_LEN\_ORDER\_I} = 10$

Так же ниже представлена структура данных, которая хранит в себе результат умножения двух чисел, она аналогична структуре хранения вещественного числа, все поля обрабатываются алгоритмом после умножения и если надо приводятся к стандартному виду

```
typedef struct
{
    char num_sign;
    char mantissa[MAX_LEN_MANTISS_RESULT_I + 1];
    char exp_sign;
    char order[MAX_LEN_ORDER_I + 1];
} result_data;
```

В поле 'num\_sign' записывается знак числа

В поле 'mantissa' записывается мантисса числа (m.n) без незначащих нулей

В поле 'num\_sign' записывается знак порядка числа

В поле 'order' записывается порядок числа

Как описывалось ранее, полям 'mantissa' и 'order' указаны заведомо числа бОльшие числа длины, описанной в условии задачи

MAX\_LEN\_MANTISS\_RESULT\_I = 50

MAX\_LEN\_ORDER\_I = 10

## Алгоритм программы

- 1) Считать числа, введенные с клавиатуры в массив символов

- 2) Привести к стандарту оба числа (убрать незначащие нули, добавить знак если отсутствует, для вещественного числа обновить порядок если требуется, вещественное число нормализовать)
- 3) Разбить числа на компоненты по структурам
- 4) Логически привести числа к целым
- 5) Развернуть числа, перемножить
- 6) Убрать все незначащие нули промежуточного результата, пересчитать порядок, нормализовать
- 7) Вывести результат в форме  $\pm 0.m1 E \pm K1$

## Тесты

Суть теста	Целое число	Действительно число	Результат
Умножение двух обычных чисел	2	2	+0.4E+1
Умножение двух обычных чисел	4	4	+0.16E+2
Второе число отрицательное	2	-2	-0.4E+1
Первое число отрицательное	-2	2	-0.4E+1
Оба числа отрицательные	-2	-2	+0.4E+1
Умножение двух обычных чисел	4	48	+0.192E+3
Первое число 0	0	4	+0.E+1
Второе число 0	4	0	+0.E+1
Оба числа 0	0	0	+0.E+1
У первого числа ведущие нули	002	2	+0.4E+1
У второго числа ведущие нули	2	002	+0.4E+1
У обоих чисел ведущие нули	002	002	+0.4E+1

