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 _____ **<Ермаков И.Г>**

Преподаватель _____ **<Фамилия ИО>**

2024

Оглавление

Условие задачи

Смоделировать операцию умножения целого числа длиной до 40 десятичных цифр на действительное число в форме ±m.n E ±K, где суммарная длина мантиссы (m+n) - до 30 значащих цифр, а величина порядка K - до 5 цифр. Результат выдать в форме ±0.m1 E ±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 нормализированном 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 <= 5

Выходные данные

Нормализированное вещественное число выводится в форме ±0.m1 E ±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 результат умножения в форме ±0.m1 E ±K1, в противном случае сообщение об ошибке.

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

Описание структур данных

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 символов), ввод в формате ±m.n E ±K где m + n <= 30, + 1 символ знака, плюс буква, обозначающая что число записано в нормализированной форме, символ знака порядка и сам порядок: K <= 5, а в программе задается значение 50.

Затем числа распределяются в структуры.

```
typedef struct
{
   int num_sign;
   int int_value[MAX_LEN_INT_I + 1];
   size_t len;
} int_data;
```

В поле ‘num_sign’ записывается знак числа (если не был введен, по умолчанию ‘+’)
В поле ‘int_value’ записывается целое число

Как описывалось ранее, полю ‘int_value’ указано заведомо число большее длины, описанной в условии задачи
MAX_LEN_INT_I = 50

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

```
typedef struct
{
   int num_sign;
   int mantissa[MAX_LEN_MANTISSA + 1];
   int exp_sign;
   int exp;
   size_t len;
} double_data;
```

В поле ‘num_sign’ записывается знак числа (если не был введен, по умолчанию ‘+’)
В поле ‘mantissa’ записывается мантисса числа без незначащих нулей
В поле ‘num_sign’ записывается знак порядка числа (если не был введен, по умолчанию ‘+’)
В поле ‘order’ записывается порядок числа (если не был указан, по умолчанию 0)

Как описывалось ранее, полям ‘mantissa’ и ‘order’ указаны заведомо числа бОльшие числа длины, описанной в условии задачи
MAX_LEN_MANTISSA_I = 40
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
{
   int num_sign;
   int mantissa[MAX_LEN_MANTISS_RESULT_I + 1];
   int exp_sign;
   int exp;
   size_t len;
} result_data;
```

В поле ‘num_sign’ записывается знак числа
В поле ‘mantissa’ записывается мантисса числа без незначащих нулей
В поле ‘num_sign’ записывается знак порядка числа
В поле ‘order’ записывается порядок числа

Как описывалось ранее, полю ‘mantissa’ указано заведомо число бОльшее числа длины, описанной в условии задачи
MAX_LEN_MANTISS_RESULT_I = 50
MAX_LEN_ORDER_I = 10

Описание основных сигнатур функций

Функция ввода, осуществляет ввод целого и вещественного числа

```
int input(char *int_num, char *double_num)
```

Функция для проверки валидности и для распределения по структуре введенного вещественного числа

```
short mantissa_check(char *input, double_data *data)
```

Функция для проверки валидности и для распределения по структуре введенного целого числа

```
short check_int(char *input, int_data *new_int)
```

Основная функция для умножения

```
short multiply(double_data *num1, int_data *num2, result_data *result)
```

Функция для вывода результата

```
void print_result(result_data *result)
```

Алгоритм программы

1) Считать числа, введенные с клавиатуры в массив символов
2) Проверить числа на валидность
3) Разбить числа на компоненты по структурам
4) Развернуть числа, перемножить
5) Убрать все незначащие нули промежуточного результата, пересчитать порядок
6) Проверить на длину мантиссы и значения порядка в соответствии с условием
7) Вывести результат в форме ±0.m1 E ±K1

Тесты

Суть теста	Целое число	Действительно число	Результат
Умножение двух обычных чисел	2	2	+0.4 E 1
Умножение двух обычных чисел	4	4	+0.16 E 2
Второе число отрицательное	2	-2	-0.4 E 1
Первое число отрицательное	-2	2	-0.4 E 1
Оба числа отрицательные	-2	-2	+0.4 E 1
Умножение двух обычных чисел	4	48	+0.192 E 3
Первое число 0	0	4	+0.0 E 1
Второе число 0	4	0	+0.0 E 1
Оба числа 0	0	0	+0.0 E 1
У первого числа ведущие нули	002	2	+0.4 E 1
У второго числа ведущие нули	2	002	+0.4 E 1
У обоих чисел ведущие нули	002	002	+0.4 E 1

Второе число не целое	3	0.15	+0.45 E 0
Первое число имеет максимальную длину	99..99 (40 девяток)	1	+0.99999999999999999999999999999999999999 E40
Проверка на округление	99..99 (40 девяток)	2	0.2 E 41
Проверка на округление	99..99 (40 девяток)	99..99 (30 девяток)	+0.99999999909999999999999999989999999999 E 70
У обоих чисел ведущий плюс	+2	+2	0.4 E 1
У обоих чисел неверный формат	+-2	+-2	ERROR_MATCHING_REG_EXPR
У обоих чисел неверный формат	++2	++2	ERROR_MATCHING_REG_EXPR
У обоих чисел неверный формат	--2	--2	ERROR_MATCHING_REG_EXPR
У второго числа неверный формат	2	2..2	ERROR_MATCHING_REG_EXPR
У первого числа неверный формат	2.2	2	ERROR_MATCHING_REG_EXPR
У первого числа неверный формат	2e2	2	ERROR_MATCHING_REG_EXPR
У второго числа неверный формат	2	2ee2	ERROR_MATCHING_REG_EXPR
Первое число слишком длинное	99..99 (31 девятка)	2	ERROR_TOO_LONG_MANTISSA
У второго числа слишком длинная мантисса	2	99..99 (31девятка)	ERROR_TOO_LONG_MANTISSA
В результате получается порядок больше 99999	1	9e99999	IMPOSSIBLE_TO_MULTIPLY_NUMBERS
В результате порядок равный 99999	1	9e99998	+0.9 E 99999
При перемножении порядок превышает 99999	1	9e99999	ERROR_TOO_LARGE_EXP
Проверка на ведущий плюс у мантиссы	2	9e+9	+0.18 E 11
Проверка на отрицательный порядок	2	9e-9	+0.18 E -7

Проверка на ведущие плюсы у мантиссы и экспоненты	**2**	**+9e+9**	**+0.18 E 11**
Проверка на альтернативный ввод вещественного числа	**1**	**123.**	**+0.123 E 3**
Проверка на альтернативный ввод вещественного числа	**1**	**.123**	**+0.123 E 0**