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»

Лабораторная Работа №1 «Длинная арифметика»

Вариант 5

Студент **Поляков Андрей Игоревич**

Группа **ИУ7-32Б**

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

Студент **Поляков А.И.**

Оценка _____

2023 г.

Описание условия задачи

Смоделировать операцию умножения действительного числа на действительное число в форме ±m.n E ±K, где суммарная длина мантиссы первого сомножителя (m+n) - до 40 значащих цифр, второго – до 35 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

Исходные данные:

Программе на вход подаются два действительных числа в следующем формате: [+-]?m.nE[+-]?k, каждое в отдельной строке. В первом числе длина части до "e" не превышает 40 значащих цифр, не считая точку и знаков "+" или "-", а во втором случае эта часть может содержать до 35 значащих цифр. В обоих случаях длина части после "k" не превышает 5 цифр, не считая знаков "+" или "-".

Результат:

Программа вычисляет действительное число, которое получается путем умножения введенных пользователем чисел. Это число представлено в формате [+-]0.mE[+-]k, где длина m не превышает 40 цифр, а длина k - 5 цифр.

Описание задачи:

Перемножение двух больших действительных чисел, с использованием длинной арифметики.

Способ обращения к программе:

Ввод чисел производится в терминале в указанном формате.

Аварийные ситуации и ошибки:

1) Одно из введенных чисел не подходит под заданный формат – код ошибки 1
2) Порядок какого-либо числа (одного из введенных или результата) превышает допустимый лимит длины – код ошибки 2
3) Мантисса какого-либо числа (одного из введенных) превышает допустимый лимит длины – код ошибки 3

Описание внутренних структур данных

```
// Максимальная длина мантиссы
#define MAX_MANTISSA_LEN 40
// Структура для представления действительного числа

struct num_t
{
    char sign; // Знак числа: '+' или '-'
    int mantissa[MAX_MANTISSA_LEN]; // Мантисса
    size_t mantissa_size;  // Размер мантиссы
    int exponent; // Порядок
};
...
int res_tmp[MAX_MANTISSA_LEN * 2];
// Временный массив для записи результата
```

Структура

- sign – Знак мантиссы – ‘+’ или ‘-‘.
- mantissa – Мантисса числа, хранится в виде массива целых, в каждом хранится одна цифра.
- mantissa_size – размер записанной мантиссы.
- exponent – порядок числа.

Описание алгоритма

1. Ввод первого числа(поочередно считывая символы из stdin) и проверка на ошибки.
   a. Считывается первый символол. Если символ равен "+" или "-", он сохраняется как знак числа. Если знака нет, то по умолчанию используется значение "+".
   b. Затем считываются и отбрасываются все начальные нули.
   c. Считываются цифры числа до десятичной точки, если таковые имеются, и сохранются в массиве мантиссы структуры. Если количество цифр превышает максимально допустимый размер, возвращается сообщение об ошибке.
   d. Если программа встречает десятичную точку, она считывает цифры после десятичной точки и также сохраняет их в массиве мантиссы, при этом записывая количество цифр после точки. Если она

встречает нецифровый символ сразу после десятичной точки, она возвращает сообщение об ошибке.

e. Если программа встречает букву "E", она считывает показатель степени этого числа. Если показатель находится за пределами допустимого диапазона, она возвращает ошибку.
f. Она корректирует показатель степени, вычитая количество цифр после запятой и добавляя общее количество цифр.
g. Она устанавливает размер мантиссы равным общему количеству цифр.
h. Программа нормализует число, убирая из начала и конца мантиссы нули.
i. Она пропускает все конечные пробелы.
j. Если следующий символ не является новой строкой, она возвращает ошибку.
k. Если она успешно считывает и обрабатывает номер, возвращается OK.

2. Ввод второго числа и проверка на ошибки.

Аналогично первому

3. Если оба числа считаны успешно, они умножаются:
    a. Инициализируется временный результирующий массив res_tmp для хранения результата умножения.
    b. Определяется знак результата. Если знаки двух чисел совпадают, результат положительный; в противном случае он отрицательный.
    c. Умножаются мантиссы двух чисел способом, аналогичным традиционному методу умножения, используемому в арифметике, при этом результат записывается в обратном порядке, начиная с конца res_tmp.
    d. Затем обрабатываются любые переносимые значения из умножения.
    e. Проверяется, превышает ли результат максимально допустимую длину. Если это так, он корректирует результат таким образом, чтобы он соответствовал максимальной длине, последовательно округляя последнее число мантисссы и обрезая ее.
    f. Затем результат записывается в структуру res, в обратном порядке.
    g. Результат нормализуется, убирая из начала и конца мантиссы нули.
    h. Наконец, проверяется, находится ли результат в пределах допустимого диапазона для показателя степени. Если это не так,

возвращается код ошибки ERR_RANGE_EXP. Если мантисса равна нулю, программа устанавливает показатель степени равным нулю. Если все в порядке, он возвращает OK.

4. Результат работы программы выводится.

Тестовые данные

Позитивные тесты

Тест	Входные данные	Результат
Число без точки и порядка	1 * 2	+0.2 E1
ЧПТ без порядка	1.2 * 2	+0.24 E1
Число без точки с порядком	2E2 * 2	+0.4 E3
ЧПТ с порядком	2.2E2 * 2	+0.44 E3
Незначащие нули перед числом	000002 * 2	+0.4 E1
Незначащие нули за точкой	2.20000000 * 2	+0.44 E1
Незначащие нули перед числом, в сумме с которыми количество чисел > 40	00000000000000000000000000000000000000002.2 * 2	+0.44 E1
Незначащие нули за точкой, в сумме с которыми количество чисел > 40	2.2000000000000000000000000000000000000000* 2	+0.44 E1
Минус перед числом	-2E2 * 2	-0.4 E3
Умножение двух малых по модулю чисел	9 * 9	+0.81 E2
Большое число на маленькое	111111111111E45 * 2	+0.222222222222 E57
Маленькое число на большое	2 * 111111111111E45	+0.222222222222 E57
Отрицательное на	-2 * 111111111111E45	-0.222222222222 E57

положительное		
Отрицательное на отрицательное	-2 * -111111111111E45	+0.222222222222 E57
Два больших числа	11111111111111E45 * 22222222222E54	+0.246913580244441975308642 E123
Число длиной в 40 цифр и число длиной в 35	99999999999999999999 9999999999999999 E123 * 99999999999999999999 999999999999999 E321	+0.9999999999999999999 99999999999999899999 E519
Проверка округления – последовательное округление до начала числа	999999999999999999 99999999999999999999 *2	+0.2 E41
Проверка округления – последняя цифра	999999999999999999 99999999999999999999 *6	+0.5999999999999999 99999999999999999999999 E41
Число с длиной порядка 5	123E10000 * 2	+0.246 E10003
Предельный порядок	`0.01E-99999 * 10`	`+0.1 E-99999`
Ноль на число	0 * 2	+0.0 E0
Число на ноль	2 * 0	+0.0 E0

Негативные тесты

Пустой ввод		Incorrect input
Слишком длинная мантисса	999999999999999999.999999 99999999999999999E3 * 3	Range error occured with mantissa
Слишком длиный порядок	34E123456 * 34	Range error occured with exponent
Буквы вместо числа	abc * 2	Incorrect input
Буквы в числе	2a1bcE4 * 2	Incorrect input
Две точки в числе	2.3.4E4 * 23	Incorrect input
Точка в порядке	2E3.4 * 2	Incorrect input

Нет мантиссы	E34 * 3	Incorrect input

Ответы на вопросы

1. Какой диапазон чисел можно представить в компьютере?

   Максимальный возможый диапазон чисел зависит от системы и выбранного типа. Например для большинства 64-разрядных систем для целых чисел это от -9223372036854775807 до 9223372036854775807 при выборе long long int или от 0 до 18 446 744 073 709 551 615 для беззнакового типа.

2. Какова точность представления чисел, и что влияет на неё?

   Точность числа зависит от количества памяти, выделенной под его мантиссу. Первый бит памяти выделяется под знак числа. Затем часть памяти занимает порядок, затем мантисса. Например для типа данных double в C для большинства 64-разрядных систем под порядок выделено 11 бит, а под мантиссу 52. Такое число имеет точность от 15 до 18 цифр мантиссы.

3. Какие операции можно выполнять над числами в стандартных условиях?

   Каждый тип данных содержит в себе операции, которые можно над ним совершать. Для целочисленных типов возможны операции сравнения, сложения, вычитания, деления, умножения, взятия остатка, а также бинарные сдвиги.

4. Какой тип данных можно выбрать программисту, если числа выходят за пределы машинного представления?

   Можно использовать либо массивы символов, либо создавать собственные структуры данных.

5. Каким образом можно проводить операции над числами, выходящими за рамки машинного представления?

   Данные операции должны быть реализованы программистом вручную, основываясь на принципах длинной арифметики.

Вывод

Когда программе необходимо обрабатывать числа, не помещающиеся в стандартные типы данных, программист должен самостоятельно реализовывать структуры для работы с такими данными, а также реализовывать операции с этими данными. В процессе работы я освоил методики работы с числами, которые выходят за пределы обычного диапазона значений, и разработал свой собственный вариант длинной арифметики.