Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное 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": обработка больших чисел

Вариант № 5

Студент: Бондарева **В. А.**

Группа: ИУ**7-34Б**

Преподаватель: **Никульшина Т. А.**

2025 г.

Условие задачи

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

Описание ТЗ

Исходные данные

Входные данные представляют из себя строки (максимальная длина входной строки – 50 символов), которые вводятся при помощи функции fgets(). Строки ввода интерпретируются как вещественные числа. Логика ввода подчиняется следующим правилам:

1. В начале строки ввода допускаются пробельные символы.
2. Первый символ строки (после пробелов) может быть знаком числа: + или -, если знак отсутствует, число считается положительным.
3. X – цифры в мантиссе числа. Длина мантиссы первого числа не превышает 35 значащих цифр, длина мантиссы второго числа не превышает значащих 40 цифр.
4. K – цифры в порядке числа. Значение в порядка должно быть в диапазоне от -99999 до +99999. Длина порядка не превышает 5 цифр.

При вводе допускаются следующие представления числа: 123, 123.456, .00025, +123001., 123.456, 1234567E-20.

Результат

Результат умножения двух чисел будет представлен в виде десятичного числа с плавающей точкой, записанного в формате +/-0.XE+/-K, где X – цифры мантиссы до 40 значащих цифр, K – цифры порядка до 5 значащих цифр.

Способ обращения к программе

Пользователь обращается к программе при помощи исполняемого файла app.exe.

Возможные аварийные ситуации и ошибки пользователя

1. Некорректный ввод: если строка ввода имеет некорректный формат, т.е. присутствуют символы отличные от символов знака числа, от числовых символов и от точки.

2. Превышение размера мантиссы: если размер мантиссы числа превышает максимально допустимый по условию.
3. Превышение размера порядка: если размер порядка числа превышает максимально допустимый по условию.
4. После знака экспоненты отсутствует число: указан знак экспоненты, однако после него нет цифр, отличных от нуля.
5. Иные ошибки и исключительные ситуации, связанные с ОС или другими программами.

Описание внутренних структур данных

Вещественное число lfloat_t (large float) имеет следующую структуру:

```
typedef struct
{
    int mantiss[MAX_MANTISS_LENGTH];
    int mant_size;
    int raw_order;
    int order;
    bool mant_sign;
} lfloat_t;
```

Листинг 1. Структура lfloat_t

- mantiss: статический массив целых чисел размера MAX_MANTISS_LENGTH, который представляет мантиссу числа

- mant_size: размер мантиссы

- raw_order: порядок, записанный после символа «E» (опциональное поле)

- order: порядок числа, определяемый из части числа до символа «E»

- mant_sign: знак мантиссы, при значении true считается, что число положительное, иначе – отрицательное

Структура lfloat_t хранит вещественное число, а именно его знак, мантиссу, порядок.

Алгоритм

Общий алгоритм (main.c)

```
#include "defines.h"
#include "input.h"
#include "process.h"
#include "output.h"


int main(void)
{
    exit_status status = SUCCESS_CODE;

    lfloat_t first_num;
    lfloat_t second_num;
    lfloat_t result_num;

    print_instructions();
    input_two_lfloats(&first_num, &second_num, &status);
    if (status == SUCCESS_CODE)
        lfloat_multiply(&first_num, &second_num, &result_num, &status);
    print_result(&result_num, &status);

    return status;
}
```

Листинг 2. Функция main()

Общий алгоритм программы, описанный в main.c, предполагает:

1. Объявление операндов и результирующего числа (типа lfloat_t).
2. Печать инструкций по работе с программой.
3. Ввод двух чисел типа lfloat_t и их запись в объявленные переменные.
4. Если не возникло ошибок при вводе чисел, то производится умножение чисел.
5. Сообщение о результате выполнения программы или сам результат выводится на экран.
6. Возвращается статус выполнения программы.

Алгоритм ввода чисел, input.c

```
/** @brief Функция для печати инструкций по работе с программой */
void print_instructions(void);
/** @brief Функция для ввода большого числа
*
* Осуществляет ввода "длинного числа": парсит входную строку, само число
(максимальная длина которого max_lfloat_len) записывает в структуру lfnum
*/
void input_lfloat(lfloat_t *lfnum, size_t max_lfloat_len, exit_status
*status);
/** @brief Функция для ввода двух больших чисел
*
* Осуществляет ввод двух "длинных чисел", последовательно вызывая
input_lfloat
* сначала для first_lfnum, а потом для second_lfnum
*/
void input_two_lfloats(lfloat_t *first_lfnum, lfloat_t *second_lfnum,
exit_status *status);
```

Листинг 3. Сигнатура функций ввода.

Алгоритм ввода двух вещественных чисел, описанный в input.c, предполагает:

1. Ввод первой строки, в которой, соответственно, содержится первый операнд, ее парсинг и запись в структуру first_lfnum.
2. Проверка статуса выполнения ввода первого операнда. Если ввод успешен, ввод может быть продолжен.
3. Печать результата парсинга строки в структуру, т.е. печать полей результирующей структуры.
4. Ввод второй строки, в которой, соответственно, содержится второй операнд, ее парсинг и запись в структуру second_lfnum.
5. Проверка статуса выполнения ввода второго операнда.
6. Печать результата парсинга строки в структуру, т.е. печать полей результирующей структуры.

Алгоритм перемножения двух вещественных чисел

```
/**
 * @brief Сдвигает элементы массива влево на указанное количество позиций
 *
 * Функция перемещает элементы массива влево, заполняя освободившуюся
 * правую часть нулями. Эквивалентно удалению первых `shift` элементов
 * и сдвигу оставшихся в начало массива
 */
static void shift_array_left(int *arr, size_t size, size_t shift);
/**
* @brief Умножает два длинных вещественных числа
*
* Функция выполняет умножение двух чисел в формате lfloat_t, обрабатывает
* мантиссы и порядки, выполняет нормализацию результата и округление
* при превышении максимальной длины мантиссы
```

```
*/
void lfloat_multiply(lfloat_t *first_lfloat, lfloat_t *second_lfloat,
lfloat_t *result_lfloat, exit_status *status);
```

Листинг 4. Сигнатура функций для перемножения двух больших чисел.

Ниже приведено словесное описание алгоритма, который описывает функцию lfloat_multiply:

1. Объявляется массив arr для хранения промежуточных результатов умножения, он инициализируется нулями.
2. Реализуется алгоритм умножения двух чисел в столбик: вложенные циклы последовательно перемножают каждую цифру первого числа на каждую цифру второго, сохраняя промежуточные результаты во временном массиве. При этом осуществляется контроль переполнения разряда: если значение в текущей ячейке превышает основание системы счисления, старшая часть переносится в следующий разряд, а младшая остаётся на текущей позиции, что обеспечивает корректное формирование итогового произведения.
3. Определяется фактическое количество значащих разрядов в результате умножения, выполняется поиск с конца временного массива. Первый найденный ненулевой элемент определяет реальный размер результата.
4. Выполняется нормализация результата умножения, если его длина превышает максимально допустимую, происходит округление. Затем обрабатываются цепочки переносов, возникших при округлении, и выполняется сдвиг массива цифр для удаления незначащих нулей.
5. Размер результирующей мантиссы и ее знак записываются в соответствующую структуру.
6. Вычисляется итоговый порядок результата умножения. Если один из множителей равен нулю, порядок результата обнуляется. В противном случае итоговый порядок формируется как сумма порядков сомножителей с добавлением коррекции.
7. Выполняется финальная проверка корректности вычисленного порядка результата: если значение порядка выходит за допустимые границы, устанавливается статус ошибки, в противном случае производится копирование рассчитанной мантиссы из временного массива в результирующую структуру.

Листинг 5. Алгоритм перемножения двух ЧПТ

Алгоритм вывода результата, output.c

Вывод результата осуществляется с помощью функции print_result(): в зависимости от статуса выполнения основного алгоритма либо печатается сообщение об ошибке, либо с помощью представленной ниже функции результирующее число выводится на экран в необходимой форме:

```
/** @brief Функция для вывода большого числа num в нормализованном виде */
void print_normalized_lfloat(const lfloat_t *num);
```

Листинг 6. Сигнатура функции print_normalized_lfloat.

Тесты

Позитивные тесты

Проверка	Ввод	Вывод
Максимальное значение порядка	4E99995 1000	+0.4000E+99999
Минимальное значение порядка	0.01E-99998 1	+0.1E-99999
Положительное на положительное	45 2	+0.90E+2
Положительное на отрицательное	33 -3	-0.99E+2
Отрицательное на отрицательное	-2.1E1 -2	+0.42E+2
Отрицательное на положительное	-65 3	-0.130E+3
Неявный знак	125 5	+0.625E+3
Умножение на 1	100 1	+0.100E+3
Умножение на 0	0 12345	+0.E+0
Максимально длинная мантисса (1)	11111111111111111111111111111111 1	+0.11111111111111111111111111111111E+35
Максимально длинная мантисса (2)	11111111111111111111111111111111111 1	+0.11111111111111111111111111111111111E+40
Циклическое округление	2 99999999999999999999999999999999999	+0.2E+1
Округление	5 77777777777777777777777777777777777	+0.38888888888888888888888888888888888889E+40

Таблица 1. Позитивные тесты

Негативные тесты

Проверка	Ввод	Вывод
Мантисса содержит лишние точки	3.2.1	Неверный символ, пожалуйста, проверьте ввод
Порядок содержит точки	11E45.4	Неверный символ, пожалуйста, проверьте ввод
Мантисса содержит лишние +	+34+44	Неверный символ, пожалуйста, проверьте ввод
Мантисса содержит лишние -	-3475-5	Неверный символ, пожалуйста, проверьте ввод
Порядок содержит лишние +	45E+7+9	Неверный символ, пожалуйста, проверьте ввод
Порядок содержит лишние -	47E-5-5-5-5	Неверный символ, пожалуйста, проверьте ввод
В строке ввода есть буквы	1aboba1	Неверный символ, пожалуйста, проверьте ввод
Превышение длины мантиссы (1)	99999999999999999999999999999999999999	Превышен размер мантиссы, пожалуйста, проверьте ввод

Превышение длины мантиссы (2)	99999999999999999999999999999999999999999999	Превышен размер мантиссы, пожалуйста, проверьте ввод
Превышение длины порядка	5E2222222222222	Некорректное значение порядка, пожалуйста, проверьте ввод
Переполнение порядка	1E99999 10	Некорректное значение порядка, пожалуйста, проверьте ввод

Таблица 2. Негативные тесты

Оценка эффективности

Скорость работы

Программа работает достаточно быстро благодаря алгоритму умножения "в столбик". Данный алгоритм подходит для работы с большими числами и не требует сложных вычислений. Время работы программы пропорционально размеру вводимых чисел - чем больше цифр в числах, тем дольше умножение, но для допустимых размеров чисел (до 40 цифр) скорость остается оптимальной.

Использование памяти

Программа эффективно использует память: все данные хранятся в заранее подготовленных статических массивах размера.

Надежность

Программа проверяет правильность введенных данных, обрабатывает специальные случаи (например, умножение на ноль) и контролирует, чтобы результаты не выходили за допустимые пределы.

Выводы по проделанной работе

В процессе работы был успешно реализован алгоритм умножения вещественных чисел. Основа алгоритма: статические массивы для хранения цифр мантиссы и система вложенных циклов для поэтапного вычисления произведения с учетом переносов. Данный подход, а именно хранение и обработка длинного числа «по частям», позволяет корректно обрабатывать большие числа, правильно определять знак и порядок результата, а также выполнять нормализацию при превышении максимального размера мантиссы, однако использование этого подхода не оправдано в случаях, когда возможностей встроенных типов данных языка достаточно для конкретных арифметических вычислений.

Контрольные вопросы

1. Каков возможный диапазон чисел, представляемых в ПК?
   Диапазон чисел, представляемых в ПК, зависит от длины машинного слова. Для 32-х разрядных систем диапазон целых знаковых чисел составляет от -2^{31} до $2^{31}-1$, а в 64-разрядной системе от -2^{63} до $2^{63}-1$.

2. Какова возможная точность представления чисел, чем она определяется?
   Точность представления числа зависит от длины мантиссы. Наиболее точное представление числа возможно, если длина мантиссы соответствует размеру машинного слова: в 64-разрядной системе максимальная точность – 52 двоичных разряда, т.е. до 15 значащих цифр.
3. Какие стандартные операции возможны над числами?
   Сложение, вычитание, умножение, деление.
4. Какой тип данных может выбрать программист, если обрабатываемые числа превышают возможный диапазон представления чисел в ПК?
   Программист может выбрать такой тип данных, позволяющий работать с числами большего диапазона, например, можно подключить специальные библиотеки, которые позволяют работать с произвольной точностью. Также можно реализовать собственный типа данных, если никакой из существующих не способен удовлетворить запросы программиста.
5. Как можно осуществить операции над числами, выходящими за рамки машинного представления?
   Операции над числами, выходящими за рамки машинного представления, можно реализовать с помощью представления тех в виде их составных частей. То есть каждое вычисление происходит при помощи «составных частей» этого числа. Также они могут быть осуществлены при помощи специальных библиотек, разработанных для конкретных задач, и возможностей, которые они представляют.