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

Группа ИУ7-35Б

Тема работы **Обработка больших чисел**

Студент Баранов Николай Алексеевич

Преподаватель Барышникова Марина Юрьевна

2022 г.

Цель работы: реализация арифметических операций над числами, выходящими за разрядную сетку персонального компьютера, выбор необходимых типов данных для хранения и обработки указанных чисел.

1. Описание условия задачи

Смоделировать операцию умножения целого числа длиной до 30 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 30 значащих цифр, а k1 – до 5 цифр.

Программа должна осуществлять ввод чисел и выдавать либо верный результат в указанном формате (при корректных данных), либо сообщение о невозможности вести счет.

2. Описание ТЗ

2.1. Описание исходных данных и результатов.

Сначала на вход программа принимает строку размером не более 64 символа. Строка заканчивается при нажатии клавиши Enter (символ переноса строки не включается в строку). Строка может содержать ведущие пробельные символы (необязательно), знак числа (+ или -, необязательно), от 1 до 30 подряд идущих цифр числа (обязательно), а также завершающие пробельные символы (необязательно). Далее на вход принимается аналогичная строка. Она может содержать ведущие пробельные символы (необязательно), знак числа (+ или -, необязательно), от 1 до 30 подряд идущих цифр числа (обязательно). В любом месте последовательности из чисел допускается однократное расположение точки (необязательно). Далее может идти пробел (необязательно), затем латинская буква E (регистр не важен, наличие необязательно), обозначающая начало экспоненциальной части, после которой может идти пробел, (необязательно), знак числа (+ или -, необязательно, только при наличии буквы E), и от 1 до 5 подряд идущих цифр (обязательно при наличии буквы E, иначе недопустимо).

На выход программа выдаёт строку, которая начинается с подстроки «0.», либо знака «-» (при наличии), затем идут от 1 до 30 десятичных цифр, затем идёт подстрока « E », затем знак «-» (при наличии), затем от 1 до 5 десятичных цифр.

2.2. Описание задачи, реализуемой программой.

Программа моделирует операцию умножения целого числа длиной до 30 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 выдает в форме ±0.m1 E ±k1, где m1 – до 30 значащих цифр, а k1 – до 5 цифр.

2.3. Способ обращения к программе.

Запуск программы:

./app.exe

В этом случае программа начнёт выполнять свою задачу. Ввод осуществляется вручную с клавиатуры, но можно перенаправить потоки ввода/вывода в файлы для автоматического запуска программы.

Вывод справки:

./app.exe –help

2.4. Описание возможных аварийных ситуаций и ошибок пользователя.

2.4.1. Переполнение буфера. Возникает, если строка при чтении оказалась длиннее 64 символов.

2.4.2. Неожиданный символ. Возникает, если в строке присутствуют символы, не предусмотренные программой, или из-за неправильного формата входных данных (например, лишний пробел, лишняя точка в числе, пробел посреди числа и т. д.).

2.4.3. Слишком большой размер числа. Возникает, если в целом числе или мантиссе больше 30 цифр, а в экспоненте более 5 цифр.

2.4.4. Переполнение экспоненты. Может возникнуть в результате перемножения чисел.

2.4.5. Отсутствие числа. Может возникнуть, если в строке нет чисел, либо отсутствует мантисса, либо отсутствует экспонента при наличии символа начала экспоненциальной части. Также может возникнуть, если поставить лишние пробельные символы после символа начал экспоненциальной части (по условию допускается не более 1 символа).

2.4.6. Неизвестный аргумент. Возникает при запуске с неправильными ключами командной строки.

3. Описание внутренних структур данных.

Для хранения больших чисел была создана отдельная структура bignum.

```
struct bignum
{
    int32_t mnt[__ARR_SIZE];
    int32_t exp;
    size_t size;
    bool msign;
};
```

Её поля:

3.1. mnt – массив размера 10 целого типа размера 4 байт (int32_t). Хранит само число. В каждой ячейке может храниться до 6 цифр (что так же позволяет спокойно перемножать 2 числа, не опасаясь переполнения).

3.2. exp – целое число размера 4 байт. Хранит экспоненту.

3.3. size – целое число типа size_t. Хранит количество цифр в мантиссе. Используется для вывода числа.

3.4. msign – логическое значение. Хранит знак числа.

4. Описание алгоритма.

Первоначально при запуске программы просматриваются ключи. Если ключи неправильные, программа уведомляет об этом пользователя и предлагает возможные варианты запуска. Если есть ключ «—help», программа выводит справку. При запуске без ключей просит пользователя сначала считать в строку целое число. После его считывания оно проверяет строку на корректность, после чего переводит число из строки в bignum. Аналогично для вещественного числа. Далее программа перемножает их и выводит результат.

При проверке строк программа сначала пропускает все пробельные числа в начале, затем смотрит наличие знака, после чего начинает считывать цифры, при этом в случае вещественного числа программа проигнорирует первую точку. Количество цифр запоминается, из-за чего раньше всего обнаруживаются переполнение и отсутствие цифр. Далее у вещественных чисел проверяется наличие экспоненты по наличию символа её начала, после чего программа проверяет аналогичным образом экспоненту. Далее для каждого случая программа смотрит оставшиеся символы. Любые отличные от пробельных символов считаются за неожиданные. Таким образом, проверка строки происходит за один проход.

При переводе строки алгоритм похож на алгоритм проверки, но здесь уже программа сначала игнорирует ведущие нули (до точки), при этом изменяя экспоненту, чтобы все числа до точки оказались за ней. Если все значащие цифры за точкой, и есть ведущие нули, то после встречи первого значащего числа программа тоже меняет экспоненту для нормализации числа. Затем программа обрезает нули в конце числа и вычисляет его длину. Далее программа вычисляет экспоненту, и тут может произойти переполнение из-за ранее произведённой нормализации. У данного подхода к решению есть один минус – число 0 приходится обрабатывать отдельно, чтобы не попасть в бесконечный цикл при обрезании.

Алгоритм перемножения похож на стандартное умножение «в столбик». Если одно из чисел – 0, то в результат сразу записывается 0. Иначе после умножения число нормализуют, мантиссу обрезают, если она слишком длинная, после чего число округляют в нужную сторону, а далее опять нормализуют.

5. Набор тестов.

Входные данные	**Выходные данные**	**Что проверялось**
2...2 (>64 символов)	Ошибка: переполнение буфера при чтении.	Переполнение буфера.
2...2 (>30 цифр)	Ошибка: переполнение числа.	Переполнение целого числа.
2e2	Ошибка: встречен некорректный символ: ‘e’, позиция символа: 2.	Некорректный символ, попытка подставить экспоненту в целое число.
where	Ошибка: число не найдено.	Отсутствие числа.

20 2..2 (>64 символов)	Ошибка: переполнение буфера при чтении.	Переполнение буфера для 2 числа, целое число без знака.
-20 2...2 (>30 цифр)	Ошибка: переполнение мантиссы.	Переполнение мантиссы, целое число со знаком «-».
-20 2d2	Ошибка: встречен некорректный символ: ‘d’, позиция символа: 2.	Некорректный символ, попытка подставить неправильный символ ка начало экспоненциальной части.
21 E300	Ошибка: мантисса не найдена.	Отсутствие мантиссы.
+21 21e	Ошибка: экспонента не найдена.	Отсутствие экспоненты, целое число со знаком «+».
21 21. E +199999	Ошибка: переполнение экспоненты.	Переполнение экспоненты, нет цифр после точки.
21 +0.1E 99999	Ошибка: переполнение мантиссы при вычислении произведения.	Переполнение при умножении.
0 222	0.0 E 0	Целое - 0
222 0	0.0 E 0	Вещественное - 0
0 0	0.0 E 0	Оба - 0
-1 1	-0.1 E -2	Две единицы разных знаков.
99...9 9...9 E -99999	0.9..98 E -99939	Максимальные размерности мантиссы, округление.
-00012 -001.2 E -3	0.144 E 0	Два отрицательных, ведущие нули.
+12 +0.00012 E 1	0.144 E 0	Ведущие нули в дробной части, плюсы в начале чисел.
1 .0..01 E -99970	0.1 E -99998	Минимальные размерности, нет цифр до точки

11 0.013	0.143 E 1	Проверка умножения, число без экспоненты.
22 11 e-8	0.242 E -4	Число без дробной части

6. Выводы по проделанной работе.

В ходе выполнения работы я изучил методы работы с длинными числами и реализовал умножение длинных чисел. Для хранения чисел я создал отдельную структуру, полями которой были массив, хранящий мантиссу, 2 числа, хранящие экспоненту и размер мантиссы, а также переменная, хранящая знак числа. Для умножения использовался стандартный алгоритм «столбиком». Из-за квадратичной сложности он является не очень эффективным, но при этом очень простым в реализации, и в рамках данной задачи имеет достаточную скорость работы из-за сравнительно небольшого размера обрабатываемых данных.

7. Ответы на вопросы.

7.1. Каков возможный диапазон чисел, представляемых в ПК?

Зависит от размера области памяти и типа значения. Для целых чисел размером n байт диапазон от -2^(8n-1) до 2^(8n-1)-1 (от 0 до 2^8n-1 для беззнаковых). Числа с плавающей точкой могут хранить числа в более широком диапазоне, но при этом значение представляется приближённо.

7.2. Какова возможная точность представления чисел, чем она определяется?

Для вещественных чисел размером мантиссы. Например, для числа размера 8 байт под мантиссу отводится 52 разряда, а под представление порядка – 11 разрядов. Целые числа не имеют погрешности, но диапазон их значений существенно меньше.

7.3. Какие стандартные операции возможны над числами?

Сложение, умножение, вычитание, деление. Для целых чисел также взятие остатка от деления.

7.4. Какой тип данных может выбрать программист, если обрабатываемые числа превышают возможный диапазон представления чисел в ПК?

Проще всего выбрать массив, т. к. если не хватает памяти для цифр, то можно взять больше чисел для хранения, и к тому же числа в массиве располагаются в памяти подряд друг за другом.

7.5. Как можно сделать операции над числами, выходящими за рамки машинного представления?

Все стандартные операции можно выполнять «в столбик». Для сложения и вычитания алгоритм эффективный, но при умножении/делении сложность возрастает квадратично, поэтому во многих случаях нужно использовать более эффективные алгоритмы.