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

Обработка больших чисел

Студент **Ильченко Е. А.**

Группа **ИУ7-34Б**

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

Студент _____ **Ильченко Е. А.**

Преподаватель _____ **Никульшина Т. А.**

2024

Описание условия задачи

Смоделировать операцию деления действительного числа в форме ±m.n E ±K, где суммарная длина мантиссы (m+n) - до 35 значащих цифр, а величина порядка K - до 5 цифр, на целое число длиной до 35 десятичных цифр. Результат выдать в форме ±0.m1 E ±K1, где m1 - до 35 значащих цифр, а K1 - до 5 цифр.

Описание технического задания

1. Описание исходных данных

Данные передаются на вход при помощи чтения функцией fgets с типом char. На вход программы ожидается следующий формат данных:

1. Первая строка содержит вещественное число в формате [+-]m.n[Ee][+-]K
    - Первый символ указывает на знак числа “+” или “-”. Если знак отсутствует, то число считается положительным
    - Затем идут символы мантиссы. Мантисса может содержать или не содержать символ “.”. Количество значащих цифр мантиссы без учета символа “.” не более 35
    - Символы [Ee] указывают на экспоненту числа. Она может присутствовать или отсутствовать. Перед символами [Ee] могут стоять пробелы
    - Символы [+-] после символа экспоненты указывают на знак экспоненты. Если знак отсутствует, то экспонента считается положительной
    - K указывает на количество символов в экспоненте. Количество цифр экспоненты не более 5
    - В записи числа не должно содержаться пробелов
2. Вторая строка содержит целое число в формате [+-]m
    - Первый символ указывает на знак числа “+” или “-”. Если знак отсутствует, то число считается положительным
    - Затем идут символы мантиссы. Количество значащих цифр мантиссы не более 35.

2. Описание результатов программы

Результат представляется в нормальном виде [-]0.m e[-]K

- Первый символ указывает на знак числа. Если он отсутствует, то число принимается положительным
- Далее следует “0.” и символы мантиссы. Количество цифр мантиссы не более 35
- Символ “e” указывает на экспоненту числа. Она присутствует обязательно
- Символ [-] после экспоненты указывает на знак экспоненты. Если он отсутствует, то число принимается положительным

- K указывает на количество символов в экспоненте. Количество цифр экспоненты не более 5

3. Описание задачи, реализуемой в программе

Реализация арифметической операции деления действительного числа на целое число, выходящими за разрядную сетку персонального компьютера. Реализация интерфейса для данной операции.

4. Способ обращения к программе

Способ обращения к программе пользователем происходит через исполняемый файл app.exe.

```
./app.exe
```

5. Описание возможных аварийных ситуаций и ошибок пользователя

1. Некорректный формат числа: если пользователь передаст число в неправильном формате, то программа вернет ошибку IO_ERROR
2. Переполнение буфера при вводе числа: если пользователь введет число больше заданных ограничений, то программа вернет ошибку BUFFER_ERROR
3. Деление на ноль: если делитель будет нулем, то функция вернет ошибку ZERO_DIVISION_ERROR
4. Переполнение при вычислении результат: если в результате выполнения программы произойдет переполнение экспоненты, то программа вернет ошибку OVERFLOW

6. Описание внутренних структур данных

Числа хранятся в структуре long_num_t, определенной в заголовочном файле long_nums.h
Листинг структуры

```
#define MANTISSA_LEN 39

typedef struct
{
   int exp_nums;
   int mantissa_sign;
   char mantissa_nums[MANTISSA_LEN];
} long_num_t;
```

- `exp_nums` хранит экспоненту числа
- `mantissa_sign` хранит знак мантиссы
- `mantissa_nums` хранит символы мантиссы. `MANTISSA_LEN` задает максимальное количество допустимых символов для хранения в мантиссе

Выбор данной структуры обосновыывется тем, что такой вид хранения позволяет наиболее эффективно обрабатывать число и передавать информацию о числе в функции.
Экспонента не превышает диапазона значений типа int, поэтому в таком виде с ней удобнее производить арифметические операции.

7. Алгоритм программы

Общий алгоритм программы:

1. Пользователю выводится информация о допустимом формате ввода для первого числа
2. Пользователь вводит первое число
3. Проверяется корректность длины вводимого числа. Если длина недопустима, то программа завершает работу и выводит сообщение об ошибке
4. Проверяется корректность ввода действительного числа. Если число введено неверно, то программа завершает работу и выводит сообщение об ошибке
5. Пользователю выводится информация о допустимом формате ввода для второго числа
6. Пользователь вводит второе число
7. Проверяется корректность длины вводимого числа. Если длина недопустима, то программа завершает работу и выводит сообщение об ошибке
8. Проверяется корректность ввода целого числа. Если число введено неверно, то программа завершает работу и выводит сообщение об ошибке
9. Выполняется деление вводимых чисел
10. Выполняется приведение частного к нормальному виду. Если возникает ошибка переполнения, то программа завершает работу и выводит сообщение об ошибке
11. Выводится частное от деления
12. Возвращается код ошибки

Фукнция check_long_float

1. Принимает строку, содержащую представление числа, и указатель на структуру типа long_num_t
2. Инициализирует переменные и флаги для работы с мантиссой и порядком числа.
3. Пропускает начальные пробелы в строке и обрабатывает возможный знак числа.
4. Пропускает ведущие нули перед значащими цифрами числа.
5. Определяет, содержится ли в строке десятичная точка. Если точка присутствует, начинается считывание мантиссы до символа точки.

6. Обрабатываются символы, пока не будет достигнута точка, конец строки или недопустимый символ. Цифры до точки записываются в массив мантиссы, обновляется индекс для массива мантиссы.
7. После нахождения точки начинается обработка части мантиссы, расположенной после неё.
8. Символы после точки считываются до появления экспоненты (символы 'E' или 'e'), конца строки или пробела.
9. Каждая цифра после точки добавляется в массив мантиссы, одновременно увеличивается счетчик количества цифр после десятичной точки.
10. По завершении считывания мантиссы строка завершается символом конца строки.
11. Если мантисса корректна, но экспоненты в строке нет, проверяется допустимость длины мантиссы, и в случае успешной проверки функция завершает работу
12. Проверяет наличие экспоненциальной части (порядка числа), начинающейся с символов 'E' или 'e'.
13. Считывает и обрабатывает порядок числа, включая возможный знак.
14. Проверяет корректность формата числа и допустимость размеров мантиссы и порядка.
15. Удаляет незначащие нули из начала мантиссы.
16. Вычисляет новое значение порядка, с учетом позиции десятичной точки.
17. Возвращает результат в виде кода успешного выполнения или ошибки ввода.

Функция check_long_int

1. Функция принимает строку, содержащую представление целого числа, и указатель на структуру типа long_num_t
2. Сначала инициализируется индекс для строки ind, устанавливается количество цифр экспоненты в ноль, а знак мантиссы по умолчанию присваивается как положительный
3. Функция пропускает начальные пробелы в строке с помощью цикла while.
4. Проверяется наличие знака числа
5. Функция считывает последовательность цифр, составляющих мантиссу, и записывает их в массив mantissa_nums структуры long_num_t, одновременно увеличивая индекс.
6. Когда встречается нецифровой символ или конец строки, массив мантиссы завершается символом конца строки.
7. Если индекс не соответствует длине строки, это указывает на наличие недопустимых символов, и функция возвращает код ошибки IO_ERROR.
8. Проверяется, не превышает ли длина мантиссы допустимый размер; в случае превышения возвращается ошибка.
9. Если все проверки пройдены успешно, функция завершает работу с кодом успешного выполнения OK.

Функция division_float_on_int

1. Функция division_float_on_int выполняет деление вещественного числа на целое число и возвращает результат в виде структуры long_num_t.
2. Сначала у делимого убирается экспонента (переменная exp_nums становится равной нулю), а старое значение экспоненты сохраняется с обратным знаком в переменной ln_exp – переменная указывает на экспоненту делителя после онулирование экспоненты делимого путем домножения
3. С помощью функции make_len_divisible_one_greater_len_divisor приводится длина мантиссы делимого к длине делителя, так что длина делимого на один больше, если делитель больше делимого, иначе равна длине делителя
4. Вычисляется длина мантисс делимого и делителя
5. Находится неполное делимое, вызовом функции find_incomplete_dividend, результат сохраняется в массиве incomplete_dividend
6. Далее начинается основной цикл деления, который продолжается до тех пор, пока все цифры делимого не будут обработаны. В каждой итерации вычисляется частное с помощью функции find_quotient
7. Вычисленное частное умножается на делитель вызовом функции multiply_long_num_on_short, и результат сравнивается с неполным делимым
8. Если произведение больше делимого, частное уменьшается, и операция умножения повторяется до тех пор, пока произведение не станет меньше либо равно неполному делимому
9. Разность между неполным делимым и произведением вычисляется с помощью функции substract_long_nums, после чего обновляется значение неполного делимого
10. В каждую итерацию цикла вычисленное частное записывается в массив мантиссы результата
11. Когда делимое заканчивается, проверяется, равен ли остаток нулю. Если остаток равен нулю, то завершается формирование результата, устанавливается знак результата и его экспонента, и функция возвращает результат
12. Если остаток не равен нулю, функция добавляет десятичную точку и продолжает деление, заполняя мантиссу дробной частью, пока не будет достигнут допустимый размер мантиссы MANTISSA_LEN или неполный делитель не станет равным нулю
13. В каждой итерации цикла для дробной части снова находится частное, производится умножение на делитель, и вычисляется новая разность с обновлением неполного делимого
14. По завершении деления устанавливаются окончательные значения экспоненты и знака мантиссы, и функция возвращает результат деления в структуре long_num_t

Функция normal_format

1. Функция normal_format преобразует мантиссу числа, хранящегося в структуре long_num_t, в нормализованный формат с учётом допустимой длины и необходимости округления
2. Сначала инициализируется временный массив mantissa для хранения нормализованной мантиссы, который включает два символа для нуля и десятичной точки. Индекс массива устанавливается на 2, так как первые два символа – это '0' и '.'
3. Если мантисса числа начинается с '0.', то пропускается первый ноль и точка, после чего происходит копирование оставшихся символов мантиссы до достижения конца строки или максимальной длины мантиссы
4. Если в мантиссе есть десятичная точка, то происходит следующее: символы до точки копируются в новый массив, при этом увеличивается экспонента (exp_nums) на каждую цифру перед точкой. После этого пропускается точка, и оставшиеся цифры также копируются в мантиссу до конца строки или достижения лимита длины
5. Если в мантиссе нет десятичной точки, все символы копируются в массив, а экспонента увеличивается для каждой скопированной цифры, пока не будет достигнут конец строки или лимит длины мантиссы
6. После копирования символов в массив мантиссы строка завершается символом конца строки
7. Если мантисса имеет максимальную допустимую длину, проверяется последний символ. Если он находится в диапазоне от '0' до '4', последний символ удаляется для предотвращения округления
8. Если последний символ от '5' и выше, выполняется округление: начиная с предпоследнего символа, к нему добавляется единица. Если результат сложения больше 9, то происходит перенос разряда, и символ устанавливается в '0', операция повторяется, пока перенос не закончится или не достигнется десятичная точка
9. Если в результате округления все символы до десятичной точки становятся равными '0', то первый символ мантиссы изменяется на '1', что указывает на увеличение порядка. После этого функция вызывает себя рекурсивно для повторной нормализации
10. После завершения всех операций нормализованная мантисса копируется обратно в структуру

8. Набор тестов

Тест	**Делимое**	**Делитель**	**Вывод**
Максимальное значение порядка	1.2e99999	2	0.6e99999

Минимальное значение порядка	1.2e-99999	2	-0.6e99999
Максимальная длина мантиссы	1.23456789012345678901234567890 12345	1	0.12345678901234567890123456789012345e1
Превышает максимальную длину мантиссы	1.23456789012345678901234567890123456	-	Ошибка и сообщение “I/O error”
Превышает максимальное значение порядка	1.2e100000	-	Ошибка и сообщение “I/O error”
Меньше минимального значения порядка	1.2e-100000	-	Ошибка и сообщение “I/O error”
Деление на ноль	1	0	Ошибка и сообщение “Zero division error”
Деление нуля на ноль	0	0	Ошибка и сообщение “Zero division error”
Наличие букв в записи числа	1a2e3	-	Ошибка и сообщение “I/O error”
Две точки в записи числа	1.1.1	-	Ошибка и сообщение “I/O error”
Две “E” в записи числа	1.1eE2	-	Ошибка и сообщение “I/O error”
Деление двух положительных чисел	1.2	2	0.6e0
Деление двух отрицательных	-1.2	-2	0.6e0

чисел			
Деление чисел с разными знаками	-1.2	2	-0.6e0
Мантисса результата превышает ограничения на длину	20e99999	1	Ошибка и сообщение “Overflow in result. Exponent has more than 5 digits”
Деление чисел с использованием отрицательной экспоненты	1.2e-2	345	0.03478260869565217391304347826086957e-3
Деление чисел с использованием положительной экспоненты	.3e40	4	0.75e39
Вещественное число с экспонентой отрицательное	-.3e40	4	-0.75e39
Округление числа	100	37	0.27027027027027027027027027027027e1
Превышение максимального значения экспоненты при округлении	100e99999	100	Ошибка и сообщение “Overflow in result. Exponent has more than 5 digits”
Граничный случай при округлении	100e99999	1000	0.1e99999
Округление вверх	1	6	0.16666666666666666666666666666666667e0

Отбрасывание последнего числа при округлении	1	9	0.111111111111111111111111111111111e0
Цикличное округление	999999999999999999999999999999999	4	0.250000000000000000000000000000000e35
Граничный случай при округлении, если делимое отрицательное	-100e99999	1000	-0.1e99999
Округление вверх, если оба числа отрицательные	-10	-6	0.166666666666666666666666666666667e1
Цикличное округление, если делитель отрицательный	999999999999999999999999999999999	-4	-0.250000000000000000000000000000000e35

Ручное тестирование проходит успешно.
Также стоит отметить, что почти все функции в коде покрыты юнит тестами.

Выводы

Для работы с числами превышающими разрядную сетку возможно создать тип данных для поддержания произвольного количества цифр. Арифметические операции для данного типа данных могут быть реализованы поразрядно

Контрольные вопросы

1. **Каков возможный диапазон чисел, представляемых в ПК?**
   В 64 битных системах максимальное значение, которое принимает число, 2^{64} (тип ulong). Минимальное значение – -2^{63} (тип long)
2. **Какова возможная точность представления чисел, чем она определяется?**
   Точность числа определяется:
   - разрядностью системы
   - длиной мантиссы

   В 64 битных системах мантисса у double обычно представлена 52 битами, что позволяет представлять числа с точностью до 2^{52}
3. **Какие стандартные операции возможны над числами?**

- сложение
- вычитание
- умножение
- деление
- возведение в степень
- целочисленное деление
- взятие остатка при делении
- сравнение
- побитовые операции

4. **Какой тип данных может выбрать программист, если обрабатываемые числа превышают возможный диапазон представления чисел в ПК?**
   Программист может написать свой тип данных с обработкой чисел поразрядно
5. **Как можно осуществить операции над числами, выходящими за рамки машинного представления?**
   Операции над числами, выходящими за рамки машинного представления, выполняются поразрядно