Московский государственный технический университет им. Н.Э. Баумана

(национальный исследовательский университет)

**Отчет по лабораторной работе №1**

по курсу “Типы и структуры данных”

Выполнила: студентка группы ИУ7-35Б

Голикова С.М.

Москва, 2021

Задание

1. Условие задачи

Смоделировать операцию деления целого числа длиной до 30 десятичных цифр на действительное число в форме ±m.n Е ±K, где суммарная длина мантиссы (m+n) - до 30 значащих цифр, а величина порядка K - до 5 цифр. Результат выдать в форме ±0.m1 Е ±K1, где m1 - до 30 значащих цифр, а K1 - до 5 цифр.

1. Внешняя спецификация
2. Исходные данные и результаты

Исходные данные:

* В первой строке - целое число в формате ±m, где количество значащих цифр не больше 30.
* Во второй строке - действительное число в формате ±m.n E ±K, где суммарная длина мантиссы (m + n) до 30 значащих цифр, а величина порядка K - до 5 цифр.

Допущения:

1) Для действительного числа: мантисса должна быть определена всегда (т.е. должна быть определена хотя бы одна из ее частей: m, n или просто ‘.’); если вводится ‘e’ или ‘E’, должен быть определен порядок.

2) Знаки необязательны (знак по умолчанию - ‘+’).

3) При вводе чисел во входной строке должны отсутствовать посторонние символы, в том числе пробелы.

Выходные данные:

Результат деления выводится в виде действительного числа в формате ±0.m1 E ±K1, где m1 - до 30 значащих цифр, а K1 - до 5 цифр.

В случае некорректного ввода или невозможности произвести вычисления выводится сообщение об ошибке.

1. Задача, реализуемая программой

Программа выполняет деление целого числа на действительное при входных данных, соответствующих формату, и при значении делителя, не равном нулю.

1. Способ обращения к программе

Программа может быть вызвана через консоль (./main.exe) или запущена через любую среду разработки С.

1. Возможные аварийные ситуации и ошибки пользователя

Аварийные ситуации и ошибки при вводе: неверный формат числа.

Аварийные ситуации в процессе расчетов и преобразований: деление на ноль, переполнение порядка в процессе вычислений.

В случае аварийной ситуации выводится сообщение о конкретной ошибке.

3. Внутренние структуры данных

Для реализации задачи были созданы следующие структуры:

typedef struct

{

char sign;

int number\_len;

char number[MAX\_MANTISSA\_SIZE];

} int\_number;

typedef struct

{

char sign;

int mantissa\_len;

char mantissa[MAX\_MANTISSA\_SIZE + 3];

int exponent;

} real\_number;

Выбор массива символов в качестве структуры данных для решения данной задачи обусловлен форматом входных данных: программа хранит в виде массива символов те данные, которые не может хранить в виде числа из-за большого размера.

4. Описание алгоритма

1. Ввод чисел

Вводятся целое и вещественное число. Вся информация о них записывается в соответствующие структуры. По мере считывания производится проверка корректности входных данных.

1. Преобразования

Числа приводятся к нормализованному виду для дальнейшего удобства работы с ними (целое число записывается в структуру вещественного типа). Также происходит проверка на равенство делителя нулю.

1. Вычисления

Деление происходит в 2 этапа: сначала мантисса первого числа делится на мантиссу второго числа в столбик (для операции деления были также реализованы функции вычитания больших чисел и умножения большого числа на однозначное целое число), а потом на основе порядков этих двух чисел вычисляется порядок результата. Затем происходит преобразование полученного результата к указанному формату.

1. Округление и вывод

Если длина мантиссы полученного числа превышает максимальный размер мантиссы, производится ее округление. В случае переполнения порядка числа печатается сообщение об ошибке. Если же все действия были выполнены успешно, число выводится на экран.

5. Основные функции, используемые в программе

|  |  |
| --- | --- |
| Имя | void normalize\_int(int\_number \*init\_int, real\_number \*normal\_int) |
| Функция | Приведение целого числа к нормализованному виду. |
|  | |
| Имя | int normalize\_real(real\_number \*init\_real, real\_number \*normal\_real) |
| Функция | Приведение вещественного числа к нормализованному виду. |
|  | |
| Имя | int division(real\_number \*dividend, real\_number \*divisor, real\_number \*quotient) |
| Функция | Деление первого числа на второе и запись результата в quotient. |
|  | |
| Имя | int greater\_or\_equal(char \*first\_num, int first\_len, char \*second\_num, int second\_len) |
| Функция | Сравнение первого числа со вторым (если первое больше или равно, чем второе, возвращается 1, иначе - 0). Числа записаны в массивы символов. |
|  | |
| Имя | int multiplication(char \*num, int len, int count, char \*new\_num) |
| Функция | Умножение числа, записанного в массив символов, на однозначное положительное число count. Результат записывается в массив символов \*new\_num. |
|  | |
| Имя | int difference(char \*num1, int len1, char \*num2, int len2, char \*res) |
| Функция | Вычитание из первого числа второго. Результат записывается в массив символов \*res. |
|  | |
| Имя | void rounding(real\_number \*num) |
| Функция | Циклическое округление числа. |

5. Тесты

|  |  |  |
| --- | --- | --- |
| Негативные тесты | | |
| Тест | Ввод | Вывод |
| Ввод вещественного числа вместо целого | 234.56 | Ошибка: некорректный ввод делимого |
| Некорректные символы в целом числе | 3424f | Ошибка: некорректный ввод делимого |
| 31 цифра в целом числе | 1234567890123456789012345678901 | Ошибка: переполнение в делимом |
| Отсутствие мантиссы в вещественном числе | 1234  e12 | Ошибка: некорректный ввод делителя |
| Отсутствие порядка в вещественном числе при условии появления символа ‘e’ | 1234  12e | Ошибка: некорректный ввод делителя |
| Неверный формат вещественного числа | 1234  1234.56.7 | Ошибка: некорректный ввод делителя |
| Некорректные символы в вещественном числе | 1234  54376fg34 | Ошибка: некорректный ввод делителя |
| 31 цифра в мантиссе вещественного числа | 1234  1234567890.123456789012345678901 | Ошибка: переполнение в мантиссе делителя |
| 6 цифр в порядке вещественного числа | 1234  1234e123456 | Ошибка: переполнение в порядке делителя |
| Делитель - ноль | 1234  0.0 | Ошибка: деление на ноль |
| Переполнение порядка в итоговом значении в верхнюю сторону | 99999  1e-99999 | Ошибка: переполнение порядка в результате вычислений |
| Переполнение порядка в итоговом значении в нижнюю сторону | 1  99999e99999 | Ошибка: переполнение порядка в результате вычислений |

|  |  |  |
| --- | --- | --- |
| Позитивные тесты | | |
| Тест | Ввод | Вывод |
| Делимое - ноль | 0  87.56 | 0 |
| Оба числа положительные | 625  25.0 | 0.25e2 |
| Первое число отрицательное | -1024  2 | -0.512e3 |
| Второе число отрицательное | 10625  -6.25e2 | -0.17e2 |
| Оба числа отрицательные | -175  -25 | 0.7e1 |
| Округление последней цифры | 10  6 | 0.166666666666666666666666666667e1 |
| Циклическое округление | 999999999999999999999999999999  2 | 0.500000000000000000000000000000e30 |
| Максимальное делимое и делитель с максимальной мантиссой | 999999999999999999999999999999  999999999999999999999999999999 | 0.1e1 |
| Минимальное делимое и делитель с минимальной мантиссой | -999999999999999999999999999999  -999999999999999999999999999999 | 0.1e1 |
| Максимальный порядок | 1  1e99999 | 0.1e-99998 |
| Минимальный порядок | 1  10e-99999 | 0.1e99999 |

6. Выводы

Обычные типы данных, такие, как int или float, не всегда подходят для хранения и обработки чисел. Большие числа, выходящие за разрядную сетку персонального компьютера, лучше хранить в виде массива символов, а обрабатывать их удобно поэлементно (например, для деления больших чисел удобно использовать алгоритм деления в столбик).

7. Контрольные вопросы

**1. Каков возможный диапазон чисел, представляемых в ПК?**

Диапазон чисел зависит от типа данных и разрядности машины.

Для целого положительного числа максимально возможное значение числа равно 2^n - 1, где n - разрядность машины (например, для 64 разрядов максимально возможное значение содержит 20 десятичных разрядов).

Если число вещественное, то под представление мантиссы отводится 52 двоичных разряда, а под представление порядка – 11 разрядов, в этом случае возможные значения чисел находятся в диапазоне от 3.6 E –4951 до 1.1 E +4932.

**2. Какова возможная точность представления чисел?**

Точность представления вещественного числа определяется максимальной длиной мантиссы (52 двоичных разряда).

**3. Какие стандартные операции возможны над числами?**

Сложение, вычитание, умножение, деление, сравнение.

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

Для представления каждой цифры большого числа можно выбрать символьный тип char и хранить все char’ы в массиве.

Вместо char можно использовать другой тип, например, int, но это будет менее выгодно по памяти.

**5. Как можно осуществить операции над числами, выходящими за рамки машинного представления?**

Хранить числа в виде массивов символов и осуществлять операции поэлементно, т.е. с каждой цифрой числа.