|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **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 «ДЛИННАЯ АРИФМЕТИКА» (ВАР. 4)**

Студент Смирнов Иван Владимирович

Группа ИУ7 – 32Б

Проверил Силантьева Александра Васильевна

*2023 г.*

Оглавление

[Описание условия задачи 3](#_Toc145778035)

[Описание внутренних структур данных 5](#_Toc145778036)

[Описание алгоритма 6](#_Toc145778037)

[Тесты 7](#_Toc145778038)

[Ответы на контрольные вопросы 9](#_Toc145778039)

[Вывод 10](#_Toc145778040)

# Описание условия задачи

Смоделировать операцию деления целого числа длиной до 40 десятичных цифр на действительное число в форме +/-m.n Е+/-K, где суммарная длина мантиссы (m+n) - до 40 значащих цифр, а величина порядка K - до 5 цифр. Результат выдать в форме +/-0.m1 Е+/-K1, где m1 - до 40 значащих цифр, а K1 - до 5 цифр.Описание технического задания

1. **Исходные данные и результат**

**Входные данные:**

1. Пользователь вводит целое число, которое изначально представляется строкой. Перед самим числом может стоять знак (+ или -). Не допускается ввод в экспоненциальном формате (1Е+5). Количество цифр числа должно быть в диапазоне от 1 до 40.
2. Пользователь вводит вещественное число в форме +/-m.n Е+/-K. Длина мантиссы (m+n) не должна превышать 40 цифр, длина порядка не должна превышать 5 цифр. Перед самим числом может стоять знак (+ или -). Допускается наличие дробной части (n > 0), экспоненциального формата (К > 0). Число может вводиться в следующих форматах: (123.; .33). При задании числа в экспоненциальном формате буква Е должна быть прописной. При длине мантиссы 0 и при наличии экспоненты (Е+10) число не считается корректно-введенным.

**Выходные данные:**

1. На экран выводится число формата: +/-0.m1 Е+/-K1, где m1 - до 40 цифр, а K1 - до 5 цифр. Для удобства чтения так же дополнительно выводится размерная линейка, на которой цифрами указаны каждые 10 цифр после точки.
2. **Задачи, реализуемой программой**

**Программа выполняет деление целого числа на вещественное, введенных по формату. Результат деления в нормализованном виде выводится на экран. Незначащие нули не выводятся.**

1. **Способ обращения к программе**

**Программа запускается через терминал.**

1. **Возможные аварийные ситуации и ошибки пользователя**
2. **Пользователь ввел вместо числа пустую строку.**

**На выходе сообщение: «Слишком мало цифр! (меньше 1)»**

1. **Пользователь ввел целое или вещественное число, не следуя указанному формату.**

**На выходе сообщение: «Число введено некорректно! (содержит посторонние символы)»**

1. **Пользователь ввел целое число, состоящее более чем из 40 цифр или вещественное число, с длиной мантиссы больше 40 цифр.**

**На выходе сообщение: «Слишком много цифр! (больше 40)»**

1. **Пользователь ввел порядок в вещественном числе, состоящий более чем из 5 цифр.**

**На выходе сообщение: «Порядок выходит за границы допустимых значений! (больше 5 знаков)»**

1. **Пользователь ввел ноль в качестве вещественного числа, на которое делят**

**На выходе сообщение: «На ноль делить нельзя!»**

1. **Если в ходе деления получилось число, порядок которого состоит более чем из 5 цифр.**

**На выходе сообщение: «Порядок частного выходит за границы допустимых значений! (больше 5 знаков)»**

# **Описание внутренних структур данных**

После ввода числа хранятся в структуре LongDouble:

Числа хранятся в массиве char, длиной MAX\_ANSWER + 2. Сделано это для того, чтобы первые 40 цифр выводились на экран, 41-ая цифра нужна для возможного округления числа, 42-ое место занято конечным символом \0 для проверки возможной ошибки.

|  |
| --- |
| /\*\*  \* @brief LongDouble - cтруктура для хранения действительного числа,  \* @brief записанного в нормализованном виде (с терминала).  \* @param int sign - знак числа (плюс или минус)  \* @param char digits[MAX\_ANSWER] - массив для хранения цифр введенного числа  \* @param int exponent - порядок (экспонента) числа  \*/  typedef struct  {  int sign;  char digits[MAX\_ANSWER + 2];  int exponent;  } LongDouble; |

При выполнении операции деления над двумя числами формата LongDouble их цифры переписываются в массив int, длинной NUMS\_SIZE = 2 \* MAX\_ANSWER + 3. Сделано это для того, чтобы цифры числа можно было сдвигать вправо и таким образом записывать результат в новый разряд.

|  |
| --- |
| /\*\*  \* @brief nums - cтруктура для хранения цифр числа LongDouble,  \* @brief над которыми можно производить математические действия.  \* @param int digits[NUMS\_SIZE] - массив цифр  \*/  typedef struct  {  int digits[NUMS\_SIZE];  } nums; |

# Описание алгоритма

1. Вводится целое число в виде строки, которое при успешном прохождении проверок записывается в структуру LongDouble
2. Вводится вещественное число в виде строки, которое при успешном прохождении проверок записывается в структуру LongDouble
3. Если первое число – ноль, то оно выводится на экран и программа завершается. Если второе число – ноль, то программа выдает сообщение и аварийно завершается.
4. Если ни первое, ни второе число не равны нулю, то на основе обоих чисел LongDouble создаются две структуры nums.
5. Деление чисел реализуется делением в столбик двух массивов nums. Вводим переменную, отвечающую за цифру в очередном разряде частного digit. Если можно вычесть из первого массива второй – вычитаем и увеличиваем digit на 1, если нет, то записываем digit в очередной разряд частного, обнуляем digit и цифры второго массива сдвигаем на 1 единицу вправо. Таким образом происходит переход на новый разряд. Действия повторяются, пока в результате не заполнился 41 разряд.
6. Математически округляем результат, если 41-ая цифра больше или равна 5.
7. Знаки исходных чисел – делим, порядки – вычитаем. Записываем новое число в LongDouble.
8. С помощью форматного вывода выводим полученное частное на экран.

# Тесты

1. **Позитивные тесты**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Входные данные | Выходные данные | Что проверяется |
| pos\_01 | 50  10 | +0.5E+1 | Умеет ли программа выполнять целочисленное деление |
| pos\_02 | 1  6 | +0.1666666666666666666666666666666666666667E+0 | Умеет ли программа округлять конечное число |
| pos\_03 | 0  10 | +0.0E+1 | Умеет ли программа выводить ноль |
| pos\_04 | 1  9.24 | +0.1082251082251082251082251082251082251082E+0 | Умеет ли программа обрабатывать число с точкой |
| pos\_05 | 0007  0.007000 | +0.1E+4 | Умеет ли программа удалять лишние нули при вводе |
| pos\_06 | 1  .33 | +0.303030303030303030303030303030303030303E+1 | Умеет ли программа делить на число без целой части |
| pos\_07 | 1  33. | +0.303030303030303030303030303030303030303E-1 | Умеет ли программа делить на число без дробной части |
| pos\_08 | 5  1E-10 | +0.5E+11 | Умеет ли программа делить на экспоненту |
| pos\_09 | 9x40  0.00000001 | +0.9999999999999999999999999999999999999999E+48 | Ввод максимального кол-ва цифр |
| pos\_10 | 12345  12E99999 | +0.102875E-99995 | При нормализации введенного числа ошибки не должно возникнуть |
| pos\_11 | 9x40  9x40 | +0.1E+1 | Умеет ли программа делить 2 одинаковых числа |
| pos\_12 | 1024  50 | +0.2048E+2 | Тест на неизменность порядка при делении |
| pos\_13 | 9024  50 | +0.18048E+3 | Тест на перемену порядка (без первого сдвига) |

1. **Негативные тесты**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер теста | Входные данные | Выходные данные | Что проверяется |
| neg\_01 | - | Слишком мало цифр! (меньше 1) | Пустой ввод |
| neg\_02 | 13  - | Слишком мало цифр! (меньше 1) | Указано только первое число |
| neg\_03 | 12.4 | Число введено некорректно! (содержит посторонние символы) | Первое число – не целое |
| neg\_04 | 228 – NOT NUMBER | Число введено некорректно! (содержит посторонние символы) | Указано не число |
| neg\_05 | 123  . | Слишком мало цифр! (меньше 1) | Вместо второго числа введена точка |
| neg\_06 | 123  E-12 | Слишком мало цифр! (меньше 1) | Вместо второго числа введена Е |
| neg\_07 | 123  0 | На ноль делить нельзя! | Деление на ноль |
| neg\_08 | 123457890х4 1 | Слишком много цифр! (больше 40) | Превышение длины мантиссы |
| neg\_09 | 123  100Е123456 | Порядок выходит за границы допустимых значений! (больше 5 знаков) | Превышение длины порядка |
| neg\_10 | 100000000000000  1Е-99990 | Порядок выходит за границы допустимых значений! (больше 5 знаков) | В ходе деление получилось число, выходящее за границы |
| neg\_11 | 0  0.0E+5 | На ноль делить нельзя! | Сначала проверяется знаменатель, а потом числитель |

# Ответы на контрольные вопросы

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

Диапазон чисел, представляемых в компьютере, зависит от используемого типа данных. Например, целочисленные типы данных, такие как int или long, могут представлять числа от -2^31 до 2^31-1 или от -2^63 до 2^63-1 соответственно. Вещественные типы данных, такие как float или double, имеют более широкий диапазон, но меньшую точность.

1. **Какова возможная точность представления чисел, чем она определяется?**

Точность представления чисел в компьютере определяется размером типа данных, используемых для их хранения. Например, тип данных float обычно имеет точность около 7 десятичных знаков, в то время как double имеет точность около 15 десятичных знаков. Точность также может быть ограничена ошибками округления и представления чисел в формате с плавающей точкой.

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

Стандартные операции над числами включают сложение, вычитание, умножение, деление. В зависимости от типа данных, могут быть доступны и другие операции, такие как нахождение целочисленного остатка от деления.

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

Программист может использовать массив символов для хранения числа, а также отдельные переменные для знака и порядка.

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

Операции над числами, выходящими за рамки машинного представления, могут быть осуществлены с помощью специальных библиотек или алгоритмов.

# Вывод

В рамках задания был реализован способ хранения чисел, выходящих за рамки машинного представления, с точностью до 40 цифр, а также разработан алгоритм деления данных чисел.