

ФАКУЛЬТЕТ: Информатика и системы управления

КАФЕДРА: Программное обеспечение ЭВМ и информационные технологии

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1 «ДЛИННАЯ АРИФМЕТИКА»**

Студент: Зайцева Алена Андреевна

Группа ИУ7 – 32Б

*2020 г.*

Оглавление

Оглавление

[1. Описание условия задачи 2](#_Toc50736356)

[2. Описание ТЗ 3](#_Toc50736357)

[1) Описание исходных данных: 3](#_Toc50736358)

[2) Описание результатов: 3](#_Toc50736359)

[3) Описание задачи, реализуемой программой: 4](#_Toc50736360)

[4) Способ обращения к программе: 4](#_Toc50736361)

[5) Описание возможных аварийных ситуаций и ошибок пользователя. 4](#_Toc50736362)

[3. Описание внутренних структур данных 4](#_Toc50736363)

[4. Описание алгоритма 6](#_Toc50736364)

[5. Набор тестов с указанием, что проверяется 6](#_Toc50736365)

[1) Негативные тесты на неверный ввод первого числа (действительного) 6](#_Toc50736366)

[2) Позитивные тесты на ввод первого числа (действительного) 8](#_Toc50736367)

[3) Негативные тесты на неверный ввод второго числа (целого) (при верном вводе первого числа) 9](#_Toc50736368)

[4) Позитивные тесты на ввод второго числа (целого) (при верном вводе первого числа) 9](#_Toc50736369)

[5) Негативные тесты на проводимую операцию деления (при верном вводе обоих чисел) 9](#_Toc50736370)

[6) Позитивные тесты на проводимую операцию деления (при верном вводе обоих чисел) 10](#_Toc50736371)

[6. Выводы по проделанной работе 11](#_Toc50736372)

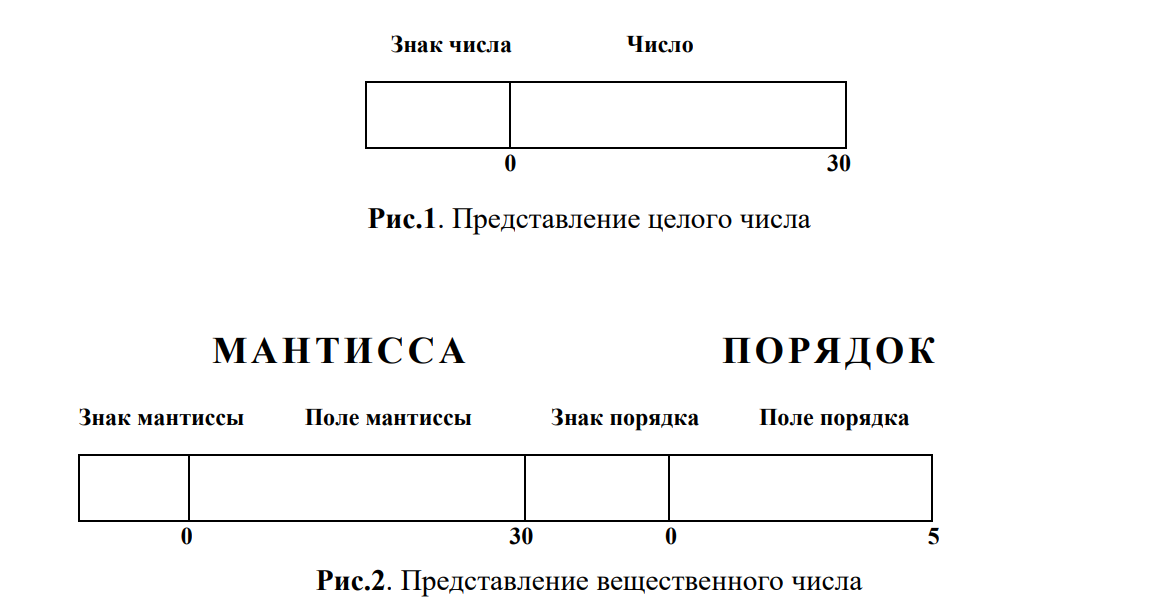
[7. Ответы на вопросы 11](#_Toc50736373)

# Описание условия задачи

Смоделировать операцию деления действительного числа в форме

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

При хранении чисел в оперативной памяти компьютера необходимо обеспечить следующий формат их представления



Десятичное число может представляться без точки: 123, при наличии десятичной точки в числе возможны следующие варианты его представления: .00025, +123001., – 123.456. Также допускается представление числа в экспоненциальной форме: 1234567 Е –20, 1234567 Е 20 или 123.4567 Е23. В программе должна быть реализована возможность ввода чисел в любом из перечисленных представлений. Результат при выдаче на печать должен быть нормализован в виде: знак 0.мантисса E знак порядок.

Если при делении чисел длина мантиссы стала больше 30 знаков, то необходимо произвести округление (если 31-й разряд больше или равен 5, то к 30-му разряду добавляется единица, если меньше 5, то 31-й разряд отбрасывается).

Все логически завершенные фрагменты алгоритма (ввод, вывод, обработка и т.п.) необходимо оформить в виде подпрограмм.

При разработке интерфейса программы следует предусмотреть:

• указание операции, производимой программой,

• указание формата и диапазона вводимых данных,

• указание формата выводимых данных,

• наличие пояснений при выводе результата.

# Описание ТЗ

Внешняя спецификация:

## Описание исходных данных:

Вводятся строго по очереди в отдельных строках 2 числа:

сначала действительное, затем целое.

* Действительное число:

Формат: ±m.n Е ±K, (обозначать экспоненту допускается как маленькой ‘e’, так и большой ‘E’)

Ограничения: суммарная длина мантиссы (m+n) - до 30 значащих цифр, а величина порядка K - до 5 цифр (от -99999 до 99999).

Знак мантиссы указывать обязательно (‘+’ или ‘-‘), знак порядка указывается по усмотрению: ‘+’ или отсутствие знака трактуется как ‘+’,

‘-‘ как ‘-‘. Число может представляться без точки: 123, при наличии десятичной точки в числе возможны следующие варианты его представления: .00025, +123001., – 123.456. Также допускается представление числа в экспоненциальной форме: 1234567 Е –20, 1234567 Е 20 или 123.4567 Е23.

* Целое число:

Формат: ±d

Ограничения: длина до 30 десятичных цифр, знак указывать обязательно (‘+’ или ‘-‘).

## Описание результатов:

* Если входные данные корректны, и операция деления прошла успешно, то выводится результат этой операции:

Формат: ±0.m1 Е ±K1, где m1 - до 30 значащих цифр, а K1 - до 5 цифр.

(нормализованный вид).

* Если же исходные данные некорректны или в результате деления произошла ошибка, то пользователь получит строку с описанием возникшей ошибки.

## Описание задачи, реализуемой программой:

Программа реализует операцию деления действительного числа на целое, если при делении чисел длина мантиссы стала больше 30 знаков, то производит округление.

## Способ обращения к программе:

Программа запускается из папки, содержащей исполняемый файл, через терминал.

## Описание возможных аварийных ситуаций и ошибок пользователя.

Возможные ошибки пользователя:

1. Не указан знак, где это требуется
2. Пустой ввод
3. Попытка ввести 2 числа в одну строку
4. Превышение ограничения по длине мантиссы вещественного числа (по значащим цифрам)
5. Превышение ограничения по величине порядка вещественного числа
6. Превышение ограничения по длине целого числа
7. Введение нечислового значения или пробельного символа, где в описании исходных данных ожидается числовое значение
8. Попытка произвести деление на 0

(при вышеперечисленных ошибках программа выдаст соответствующее сообщение об ошибке)

1. При вводе вещественного числа была указана экспонента, за которой ничего не последовало: программа не продолжит выполнение, пока не встретит непробельный символ.

Возможные аварийные ситуации:

1. В результате деления получится ‘машинный нуль’ или переполнение порядка: программа выведет сообщение об этой ошибке.

# Описание внутренних структур данных

1. Для представления вещественного числа (вводимого и вычисляемого) используется следующая структура:

typedef struct

{

char sign;

short int mant[31];

int order;

} float\_struct;

Для представления знака выбран «символьный тип», так как он занимает меньше всего памяти и может представить оба допустимых значения: «+» или «-».

Для представления мантиссы числа используется массив длиной 31 (допустимая длина мантиссы – 30 значащих цифр, незначащие отбрасываются программой при вводе, 31 дополнительный символ используется для округления), с элементами типа «короткое целое», так как он будет использован только для хранения цифр от 0 до 9 и числа 10. Он занимает по памяти больше, чем символьный массив (который также мог бы представить цифры, а числа 10 при изменении алгоритма можно избежать), однако для реализации деления потребовалось бы вызывать функцию преобразования символа в число очень часто, что значительно увеличило бы время выполнения программы.

Для представления порядка выбран тип «целое», так как компиляторе это наименьший тип, который сможет вместить допустимые значения порядка (от -99999 до 99999).

1. Для представления вводимого целого числа используется следующая структура:

typedef struct

{

char sign;

short int integer[30];

} integer\_struct;

Для представления знака выбран «символьный тип», так как он занимает меньше всего памяти и может представить оба допустимых значения: «+» или «-».

Для представления мантиссы числа используется массив длиной 30 (допустимая длина целого числа – 30 значащих цифр, незначащие отбрасываются программой при вводе), с элементами типа «короткое целое», так как он будет использован только для хранения цифр от 0 до 9. Он занимает по памяти больше, чем символьный массив (который также мог бы представить цифры, однако для реализации деления потребовалось бы вызывать функцию преобразования символа в число очень часто, что значительно увеличило бы время выполнения программы.

1. Для переменных, которые используются в программе для хранения цифр и числа 10, а также в качестве индексов в массивах из вышеперечисленных структур используется тип «короткое целое», как минимальный по занимаемой памяти тип, способный представить значения от 0 до 31.

# Описание алгоритма

1. Вводится первое (действительное) число и проверяется его корректность (если оно введено некорректно, программа выводит соответствующее сообщение и завершается)
2. Первое число нормализуется
3. Вводится второе число и проверяется его корректность (если оно введено некорректно, программа выводит соответствующее сообщение и завершается)
4. Производится деление “в столбик” первого числа на второе
5. Результат округляется
6. Результат нормализуется и проверяется его попадание в пределы допустимых значений (если произошло переполнение порядка или машинный нуль, программа выводит соответствующее сообщение и завершается)
7. Выводится результат деления в нормализованном виде

Код программы содержит комментарии и спецификацию используемых в программе подпрограмм.

# Набор тестов с указанием, что проверяется

## Негативные тесты на неверный ввод первого числа (действительного)

|  |  |  |
| --- | --- | --- |
| Что проверяет | Тест | Результат |
| Пустой ввод | ENTER | ERROR: empty input |
| Не введен знак мантиссы | 1 | ERROR: didn't get a sign where expected |
| Превышено максимально допустимое количество значащих цифр в мантиссе (на границе: 30+1, и больше) | -1234567890123456789012345678901 | ERROR: the mantissa is too long (m+n) is more than 30 significant digits |
| -0.01234567890123456789012345678901 |
| +123456789012345.67890123456789012E12 |
| При вводе мантиссы введен неизвестный нечисловой символ (не «e», не «Е», не «.») | -12d | ERROR: met an unknown symbol |
| +1?4 |
| Введен слишком маленький порядок | -1e-123456 | ERROR: incorrect order input: less than -99999 or more than 99999 or met an unknown symbol |
| Введен слишком большой порядок (на границе 99999+1) | +0.1e100000 |
| При вводе порядка введено нечисловое значение | -1er |

## Позитивные тесты на ввод первого числа (действительного)

|  |  |  |
| --- | --- | --- |
| Что проверяет | Тест | Результат |
| При подсчете цифр в мантиссе учитываются только значащие. Проверка на граничном значении (30) и меньше | -000123456789012345678901234567890 | (программа переходит к считыванию второго числа) |
| -0.0123456789012345678901234567890 |
| +000000000000000000123456789012345.6789E12 |
| Число введено без десятичной точки | -12 |
| Число введено в экспоненциальной форме | +13e0 |
| Сразу за знаком следует точка | -.1 |
| В мантиссе нет значащих цифр | +0.0e10 | The mantissa has no significant digits: 0 divided by any number is 0 |

## Негативные тесты на неверный ввод второго числа (целого) (при верном вводе первого числа)

|  |  |  |
| --- | --- | --- |
| Что проверяет | Тест | Результат |
| Пустой ввод | ENTER | ERROR: empty input |
| Не введен знак | 1 | ERROR: didn't get a sign where expected |
| Превышено максимально допустимое количество значащих цифр (на границе: 30+1 и больше) | -12345678901234567890123456789012 | ERROR: the integer has more than 30 significant digits |
| +001234567890123456789012345678901 |
| Введено нечисловое значение | -one | ERROR: met an unknown symbol |
| +1t2 |
| Попытка ввести дробное число | -1.2 |
| Введен нуль | +0 | ERROR: can't divide by zero |

## Позитивные тесты на ввод второго числа (целого) (при верном вводе первого числа)

|  |  |  |
| --- | --- | --- |
| Что проверяет | Тест | Результат |
| При подсчете цифр в числе учитываются только значащие. Проверка на граничном значении (30) и меньше | -000123456789012345678901234567890 | (программа переходит к делению) |

## Негативные тесты на проводимую операцию деления (при верном вводе обоих чисел)

|  |  |  |
| --- | --- | --- |
| Что проверяет | Тест | Результат |
| Получен машинный нуль (порядок ниже граничного значения: -99999-1) | -0.1e-99999  +10 | ERROR: met a machine zero |
| Происходит переполнения порядка (порядок выше граничного значения: 99999+1) | +100e99999  +10 | ERROR: as a result of normalization, an order overflow occured |

## Позитивные тесты на проводимую операцию деления (при верном вводе обоих чисел)

|  |  |  |
| --- | --- | --- |
| Что проверяет | Тест | Результат |
| Правильность определения знака | -1  -1 | +0.1E1 |
| +1  +1 |
| -1  +1 | -0.1E1 |
| +1  -1 |
| Деление числа на самого себя должно давать единицу | -12e0  -12 | +0.1E+1 |
| Деление числа на единицу должно давать то же число | +0.1E100  +1 | +0.1E+100 |
| Проверка реализации нормализации: порядок должен уменьшиться | +0.0001E100  +1 | +0.1E+97 |
| Проверка реализации нормализации: порядок должен увеличиться | +1.1E100  +1 | +0.11E+101 |
| Выводятся только значащие цифры | -1  +2 | -0.5E0 |
| В случае необходимости выводится максимальное кол-во значащих цифр (30) | -31  -3 | +0.103333333333333333333333333333E+2 |
| Результат округляется | +2  +3 | +0.66666666666666666666666666667E0 |
| Работа с числами “средней длины” | -0.369246123e-10000  +0123 | -0.3002001E-10002 |
| Работа с числами на границе допустимых значений разряда | -123123123123123.123123123123123e99984  +123 | -0.1001001001001001001001001001E+99997 |
| +0.999999999999999999999999999999e-99970  +999999999999999999999999999999 | +0.1E-99999 |
| Происходит  циклическое округление  (с 31 разрядом получится 0.19999999199999991999999919999999199E-1 | +0.19999999  -99999999 | +0.19999999199999992E-8 |

# Выводы по проделанной работе

Для обработки чисел, превышающих возможный диапазон представления чисел в ПК, нельзя применять стандартные типы данных и встроенные в язык программирования операции над числами.

Для произведения таких вычислений следует разрабатывать структуры данных, которые смогут хранить числа нужной нам точности и величины. Тогда для хранения знаков удобно использовать символьный тип, для хранения мантиссы – числовой массив, для хранения порядка обычно достаточно целого типа.

Также нужно разрабатывать собственные функции, которые будут учитывать требования к точности и диапазону. При этом их проще реализовывать, если все входящие данные проходят промежуточный этап – нормализацию.

Минусы данного подхода в необходимости тратить время на написание дополнительных функций и их отладку, а также в том, что они не будут лишены трудно уловимых ошибок, которые могут привести к непредвиденным последствиям. Однако если подойти к реализации собственных функций тщательно и протестировать их как можно лучше, мы получим главный плюс – возможность обрабатывать любые необходимые нам числа.

# Ответы на вопросы

1. Каков возможный диапазон чисел, представляемых в ПК?

Диапазон чисел зависит от наличия/отсутствия знака у числа, выбранного типа, отведенной для него памяти, разрядности процессора. Например: знакового короткого целого, под который выделено 2 байта диапазон будет - [−32768, +32767].

Максимально под представление мантиссы отводится 52 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.

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

Точность вещественного числа определяется количеством разрядов, отведённых для хранения мантиссы. Например: из 4 байт, выделенных под хранение цифры, 1 (8 разрядов) отдается под данные о порядке и его знаке, а 3 байта (24 разряда) уходят на хранение мантиссы и её знака по тем же принципам, что и для целочисленных значений. Тогда точность будет равна 7 десятичным знакам.

Максимально под представление мантиссы отводится 52 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.

1. Какие стандартные операции возможны над числами?

Сложение, вычитание, умножение, деление, сравнение.

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

Для представления такого числа программист может разбить число на мантиссу и порядок, на числа/цифры (с помощью массива чисел/символов/строк), отдельно хранить знаки мантиссы и порядка. Каждый из этих элементов должен удовлетворять диапазону представления чисел в ПК.

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

Можно реализовать собственные функции, которые выполняли бы указанные операции, используя представление числа в заранее выбранном формате (см пункт 4).