	Министерство образования и науки 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 (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1 «ОБРАБОТКА БОЛЬШИХ ЧИСЕЛ»

Студент, группа **Рядинский К.В., ИУ7-33Б**

2020 г.

Описание условия задачи

Смоделировать операцию деления действительного числа в форме m.n E 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 E K1, где m1 - до 30 значащих цифр, а K1 - до 5 цифр.

Техническое задание

Входные данные:

Действительное число: строка, содержащая вещественное число в виде [+/-]m[.n][e[+/-]K]
Суммарная длина мантиссы (m+n) – до 31 цифры (считая точку), порядка K – до 5 цифр. Наличие точки, знака порядка и знака числа обязательно.

Целое число: строка, содержащая целое число в виде [+/-]m . Суммарная длина модуля числа (m) – до 30 цифр. Наличие знака перед числом обязательно.

Выходные данные:

- длинное число, нормализованное в виде [-]0.m1 e[-]K1, где длина m1 – до 30 цифр, K1 – до 5.
-

Функция программы: деление действительного числа на целое.

Обращение к программе: запускается из терминала.

Аварийные ситуации:

1. Некорректный ввод строки с целым числом.
   На входе: строка, хотя бы один символ в которой не цифра и не символ +/- (если это первый элемент в строке).
   На выходе: сообщение «Ошибка: встречен некорректный символ»
2. Некорректный ввод строки с действительным числом.
   На входе: строка, хотя бы один символ в которой не цифра и не символ из набора “+ - . e E”.
   На выходе: сообщение «Ошибка: встречен некорректный символ» или сообщение «Ошибка: в числе должен быть только один символ E» или «Ошибка: в числе должна быть только одна точка».

3. Превышение длины строки при вводе целого числа.
   На входе: корректное число, длина которого превышает 30 цифр.
   На выходе: сообщение «Ошибка: Вы ввели слишком длинное целое число.»
4. Превышение длины строки при вводе действительного числа.
   На входе: корректное число, длина мантиссы которого превышает 31 цифру (считая точку) или длина порядка превышает 5 цифр.
   На выходе: сообщение «Ошибка: был введен слишком длинный порядок»
5. Переполнение порядка.
   На входе: в процессе деления степень полученного в результате числа по модулю превышает 99999.
   На выходе: сообщение «Ошибка: переполнение порядка»
6. Деление на ноль.
   На входе: дробное число, равное 0.
   На выходе: сообщение «Ошибка: деление на ноль»

Структуры данных

Для хранения введенного числа используется тип char. Был выбран этот тип, так как он занимает всего 1 байт на большинстве машин, чего нам будет вполне достаточно для хранения цифр числа

Char* buff*[BUFF_SIZE];

где **BUFF_SIZE** = 60.

После ввода строки, она разбивается на составляющие и каждая часть описывается полем в структуре **bigfloat_t**

```
typedef struct bigfloat_s
{
   sign_t mant_sign;
   uint8_t mantissa[MANTISSA_LENGTH];
   uint8_t dot_pos;
   int exponent;
   int mant_len;
} bigfloat_t;
```

Поля структуры:

- **mant_sign** – знак числа. Принимает значения “+” или “-”;
- **mantissa[MANTISSA_LENGTH]** – значение мантиссы числа (часть между знаком числа и знаком экспоненты), **MANTISSA_LENGTH** = 31
- **dot_pos —** индекс точки в числе
- **exponent —** значение порядка. Принимает значения от -99999 до 99999
- **mant_len —** текущая длинна мантиссы. Так как хранится мантисса в виде массива цифр.

Алгоритм

1. Пользователь вводит две строки: сначала строка, содержащая действительное число, затем — целое число
2. Разбивка (парсинг) строк на составляющие и проверка всех компонентов структуры на валидность.
3. Если делитель равен нулю, то выводится сообщение об ошибке (см. Пункт 6 АС).
4. Если делимое равно нулю, то конечный результат — 0
5. Выполняется деление методом «деление столбиком», при этом контролируется округление
6. При переполнение порядка, выводится сообщение об ошибке (см. Пункт 4 АС)
7. Если процесс деления прошел без ошибок, то результат выводится пользователю на экран в нормализованном виде, согласно спецификации, указанной в ТЗ

Тесты

	Тест	Число 1	Число 2	Результат
1	Некорректный ввод	+q	-	Ошибка: встречен некорректный символ
2	Некорректный ввод	+123E+45	+a	Ошибка: Встречен некорректный символ
3	Некорректный ввод	+1.2.3E+0	+12345	Ошибка: точка должна быть только одна

4	Некорректный ввод	+123E+45.67	+123	Ошибка: встречен некорректный символ
5	Некорректный ввод	123E+0	+123	Ошибка: Первый символ должен быть + или -
6	Некорректный ввод	+1E+	-	Ошибка: вы не ввели порядок числа
7	Некорректный ввод	+e+0	-	Ошибка: вы не ввели мантиссу числа
8	Некорректный ввод	1E+00	+	Ошибка: вы не ввели длинное целое число
9	Превышение длины мантиссы	+999...999 (31 девятка)E+0	-	Ошибка: Вы ввели слишком длинную мантиссу
10	Превышение длины мантиссы	+123e+0	+0.999...999 (31 девятка)	Ошибка: Вы ввели слишком длинное целое число
11	Превышение длины порядка	+0.1E+123456	1	Ошибка: Вы ввели слишком длинный порядок
12	Деление нуля	+0E+0	+123	+0.0E+0
13	Деление на нуль	+123e+0	+0	Ошибка: деление на нуль

14	Деление на нуль	+0e+e	+0	Ошибка! Деление на ноль!
15	Округление	+2e+0	+3	+0.666...667E+0
16	Переполнение порядка	+1e-99999	+10	Ошибка: переполнение порядка
17	Деление целых чисел	+100e+0	+5	+0.2E+2
18	Деление чисел разных знаков	+5e+0	-1	-0.5E+1
19	Граничные значение	+999...999.E+99999 (30 девяток)	+999...999 (30 девяток)	+0.1E-99998
20	Граничные значения	+99..99e+0	+2	+0.5E+30
21	Граничные значения	+10E+99999	+1000	+0.1E-99997

Функции

Void hello_print(void)

Функция выводит поясняющую информацию о работе программы на экран

Аргументы

—

Возвращаемые значения

—

*my_error_t parse_long_int(const char *src, bigfloat_t *dst)*

Разбивает строку *src* на составляющие и кладет их в dst. Проводит проверки на валидность данных.

Аргументы

Src — исходная строка

Dst — возвращаемая структура (длинное целое число)

Возвращаемое значение

Код ошибки

*my_error_t parse_long_float(const char *src, bigfloat_t *dst)*

Разбивает строку *src* на составляющие и кладет их в dst. Проводит проверки на валидность данных.

Аргументы

Src — исходная строка

Dst — возвращаемая структура (длинное действительное число)

Возвращаемое значение

Код ошибки

*my_error_t find_dot_ind(const char *src, bigfloat_t *dst)*

Ищет точку в строке src и сохраняет ее индекс

Аргументы

Src — исходная строка

Dst — возвращаемая структура (длинное действительное число)

Возвращаемое значение

Код ошибки

*my_error_t input_long_int(bigfloat_t *dst)*

Запрашивает у пользователя строку и далее «парсит» ее, чтобы получить длинное целое число

Аргументы

Dst — структура длинного числа

Возвращаемое значение

Код ошибки

*my_error_t input_long_float(bigfloat_t *dst)*

Запрашивает у пользователя строку и далее «парсит» ее, чтобы получить длинное целое число

Аргументы

Dst — структура длинного числа

Возвращаемое значение

Код ошибки

Void print_res(bigfloat_t res)

Печатает результат деления на экран

Аргументы

Res — структура, содержащая результат деления

Возвращаемое значение

—

*Char *mystrerr(my_error_t err)*

Возвращает сообщение об ошибке по номеру ошибки *err*

Аргументы

err — номер ошибки

Возвращаемое значение

Строка с текстом об ошибке

Void normalise(bigfloat_t *src)

Проводит нормализацию числа *src*

Аргументы

src — число, которое требуется нормализовать

Возвращаемое значение

—

my_error_t division(bigfloat_t *divided, bigfloat_t *divider bigfloat_t *res)

Делит divided на diviser и помещает результат в res

Аргументы

Divided — делимое

Diviser — делитель

Res — результат

Возвращаемое значение

Код ошибки

Void str_to_int(const char *str, int *dst, int len)

Перевод str в число dst

Аргументы

Str — искомая строка

Dst — результат перевода

Len — длина str

Возвращаемое значение

—

Void remove_last_zeroes(bigfloat_t *f)

Удаляет нули справа у числа f

Аргументы

f — число

Возвращаемое значение

—

Int count_int_iters(bigfloat_t *divided, bigfloat_t *diviser)

Считает очередную цифру числа результата деления

Аргументы

Divided — делимое

Diviser — делитель

Возвращаемое значение

Цифра числа

my_error_t mant_sub(bigfloat_t *divided, bigfloat_t *diviser)

Подготовительный алгоритм для деления

Аргументы

Divided — делимое

Diviser — делитель

Возвращаемое значение

Код ошибки

Int is_nil(const bigfloat_t num)

Проверяет, является ли число нулем

Аргументы

Num — длинное число

Возвращаемое значение

1 — если число — 0

0 — иначе

Void right_shift(bigfloat_t *f)

Сдвигает массив цифр числа — мантиссу — f вправо

Аргументы

f — длинное число

Возвращаемое значение

—

Void remove_lead_zeros(bigfloat_t *src)

Удаляет лидирующие нули

Аргументы

Src — длинное число

Возвращаемое значение

—

Контрольные вопросы

1. Каков возможный диапазон чисел, представляемых в ПК?

Возможный диапазон чисел зависит от их типа, размера выделенной для их хранения памяти, разрядности процессора. Для беззнакового целого числа выделяется 64 двоичных разряда, то есть его максимальное значение – 18 446 744 073 709 551 615 (long long unsigned int).

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

Точность представления вещественных чисел зависит от количества памяти, выделенного для хранения мантиссы. Для мантиссы типа double выделяется 52 бита, то есть мантисса может принимать значения до 4 503 599 627 370 496.

3. Какие стандартные операции возможны над числами?

Предусмотрено выполнение сравнения, сложения, вычитания, умножения, деления, взятия остатка от деления.

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

Можно создать структуру, хранящую массив из цифр — мантисса, — порядок, знак мантиссы.

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

Для этого можно использовать некоторые языки — где уже есть поддержка длинных чисел — или библиотеки, или написать свое решение.

Вывод

При выполнении данной лабораторной работы я реализовал деление действительного числа на целое числа. Я получил опыт в работе с массивами, структурами и типами данных. При необходимости обрабатывать числовые данные выходящие за пределы разрядной сетки, следует использовать тип структуры для хранения и обработки.