**Отчёт лабораторной работы №1**

по курсу «Типы и структуры данных»

Отчёт выполнила:

Кондрашова Ольга

Группа ИУ7-35Б

Вариант 9

Задача 4

**Цель работы**:

Реализация арифметических операций над числами, выходящими за разрядную схему персонального компьютера, выбор необходимых типов данных для хранения и обработки указанных чисел.

**Задание (Вариант 9)**:

Составить программу умножения или деления двух чисел, где порядок имеет до 5 знаков: от -99999 до +99999, а мантисса — до 30 знаков. Программа должна осуществлять ввод чисел и выдавать либо верный результат в указанном формате (при корректных данных), либо сообщение о невозможности произвести счет.

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

**Применение:**

Длинная арифметика применяется в следующих областях:

* составление кода для процессоров (микроконтроллеров) низкой разрядности;
* криптография;
* математическое и финансовое ПО. Результат вычисления на бумаге должен совпадать с результатом работы компьютера с точностью до последнего разряда.

Для научных и инженерных расчётов длинная арифметика применяется редко, так как ошибки во входных данных обычно намного больше, чем ошибки округления.

**Входные данные:**

На вход подается 2 числа в десятичной системе счисления:

1. Целое число.

Длина до 30 десятичных цифр. Если число положительное, вводить знак «+» необязательно. Отрицательные числа обязательно со знаком «-». Ведущие нули тоже учитываются при проверке превышения лимита в 30 цифр. Допускаются пробелы между цифрами (программа самостоятельно производит склейку числа). Ввод любых других символов является недопустимым.

2. Вещественное число.   
Формат ввода: +-m.n E +-K. Мантисса длиной до 30 десятичных цифр. Если число положительное, вводить знак «+» необязательно. Отрицательные числа обязательно со знаком «-». Ведущие нули тоже учитываются при проверке превышения лимита в 30 цифр. Допускаются пробелы между цифрами (программа самостоятельно производит склейку числа). Ввод любых других символов является недопустимым. Вводить точку не обязательно, но ее можно ввести лишь один раз. Можно ввести е/Е (или не вводить). Порядок - целое число (со знаком или без, но одним). По модулю K не превышает 99999. Если введено e/E после этого обязательно нужно ввести число.

**Выходные данные:**

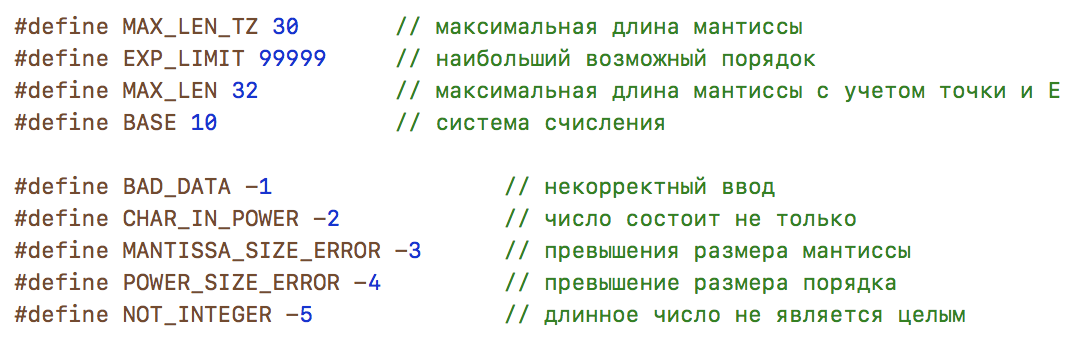
Результат выдается в форме: ± 0.m1 E K1, где m1 - до 30 значащих цифр, а K1 - до 5 цифр.

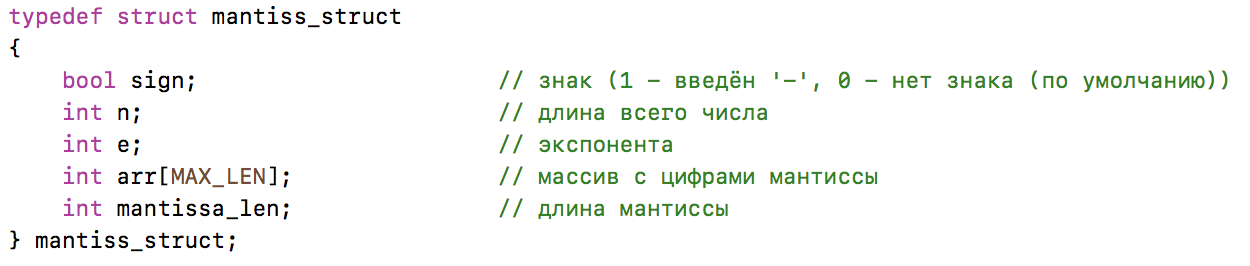
Если делитель равен нулю, программа выдает соответствующее сообщение.

**Обращение к программе:**Через консоль

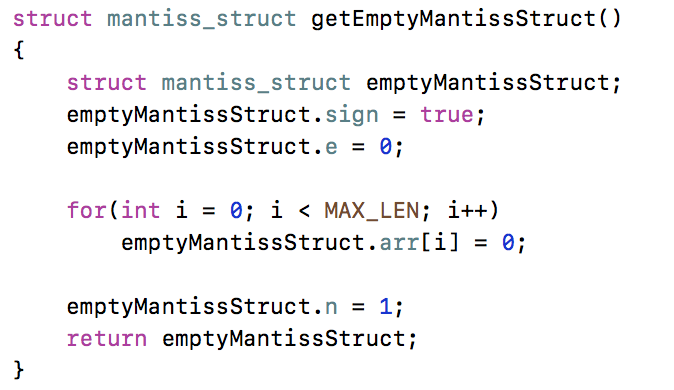
**Внутренняя структура данных:**

Длинные числа представлены в виде структуры mantiss\_struct.





А также структура пустого шаблона длинного действительного числа 0. E 0.



**Функции:**

void reverse(struct mantiss\_struct \*res)

Вход: структура.

Выход: структура в перевернутом виде.

Назначение: функция принимает на вход структуру и переворачивает цифры в массиве, входящем в неё.

bool isDigit(char value)

Вход: символ, вводимый с клавиатуры.

Выход: true, если символ – цифра, в противном случае false.

Назначение: функция проверяет, является ли введённый символ цифрой.

int inputMantissInteger(struct mantiss\_struct \*array)

Вход: структура пустой шаблон длинного числа.

Выход: структура – длинное целое число.

Назначение: функция заполняет нулевую структуру в соответствии с введёнными данными.

int inputMantissFloat(struct mantiss\_struct \*array)

Вход: структура пустой шаблон длинного числа.

Выход: структура – длинное действительное число.

Назначение: функция заполняет нулевую структуру в соответствии с введёнными данными.

void addNulls(struct mantiss\_struct \*value)

Вход: структура длинного действительного числа.

Выход: структура длинного действительного числа с нулями на конце.

Назначение: функция добавляет нули в конец массива, чтобы делимое было одной длины с делителем.

void extend(struct mantiss\_struct \*tmp)

Вход: структура длинного действительного числа.

Выход: структура длинного действительного числа, с расширенным массивом.

Назначение: функция расширяет массив на один разряд.

mantiss\_struct multiply(const struct mantiss\_struct \*number, int multiplier)

Вход: структура длинного действительного числа, целое короткое число.

Выход: структура длинного действительного числа, с массивом, умноженным на число multiplier.

Назначение: функция умножает массив в структуре на целое число.

bool compare(struct mantiss\_struct number\_1, struct mantiss\_struct number\_2)

Вход: две структуры длинных действительных чисел.

Выход: результат сравнения данных чисел (true – если первое число больше, false – в противном случае).

Назначение: функция сравнивает два длинных действительных числа.

mantiss\_struct difference(struct mantiss\_struct number\_1, struct mantiss\_struct number\_2)

Вход: две структуры длинных действительных чисел.

Выход: разность данных чисел (структура).

Назначение: функция находит разность двух длинных действительных чисел.

int delUselessNulls(struct mantiss\_struct \*value)

Вход: структура длинного действительного числа.

Выход: структура длинного действительного числа без незначащих нулей.

Назначение: функция удаляет незначащие нули в начале массива.

void roundArr(struct mantiss\_struct \*value)

Вход: структура длинного действительного числа.

Выход: структура длинного действительного с округлённым массивом.

Назначение: функция округляет массив длинного действительного числа

mantiss\_struct division(struct mantiss\_struct \*dividend, struct mantiss\_struct \*divider)

Вход: две структуры длинных действительных чисел (делимое и делитель).

Выход: структура частного данных чисел.

Назначение: функция находит частное двух длинных действительных чисел.

В функции вводятся дополнительные переменные:

res – конечный результат;

tmp – промежуточное делимое;

cur – промежуточный делитель;

x – искомое число такое, что cur \* x < tmp, записывается в res;

mul – произведение делителя и x.

bool isZero(struct mantiss\_struct \*divider)

Вход: структура длинного действительного числа.

Выход: true – если делитель равен нулю, false – в противном случае.

Назначение: функция определяет, является ли делитель числом.

void writeMantiss(struct mantiss\_struct \*res)

Вход: структура длинного действительного числа.

Выход: строка, выведенная на экран.

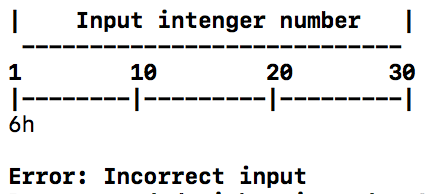
Назначение: функция выводит на экран результат деления.

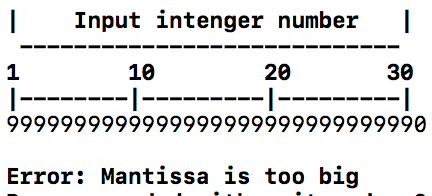
**Описание алгоритма:**

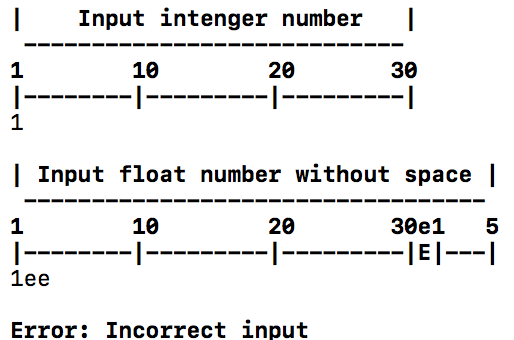
1. Объявление нулевых структур действительных чисел.
2. Ввод целого числа в структуру с проверкой правильности ввода и проверкой на целое.
3. Ввод действительного числа в структуру с проверкой правильности ввода.
4. Перевод целого числа в действительное.
5. Проверка деления на ноль.
6. Деление чисел друг на друга.
   1. Вводим дополнительные переменные: res – конечный результат, tmp – промежуточное делимое, cur – промежуточный делитель, mul – произведение делителя и x.
   2. Подбираем такой x, что cur \* x < tmp
   3. Вычитаем из делимого произведение делителя и x.
   4. Записываем найденный x в массив в res.
   5. Считаем экспоненту res.
   6. Определяем знак ответа res.
7. Нормализация полученного числа.
8. Вывод конечного результата.

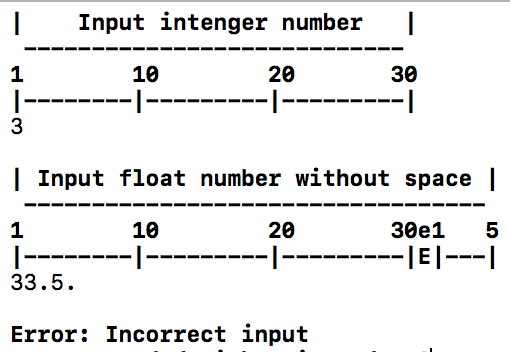
**Тесты:**

Проверка некорректного ввода.

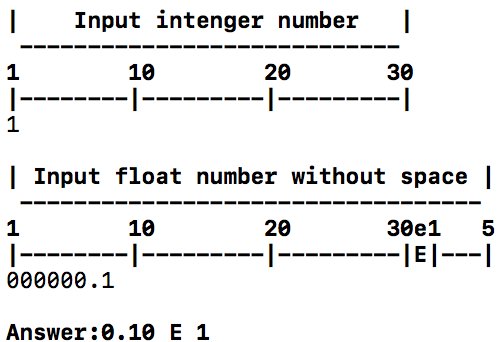


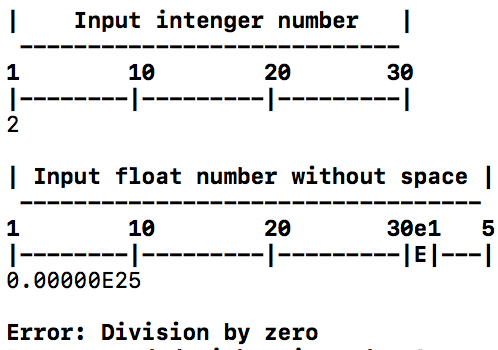


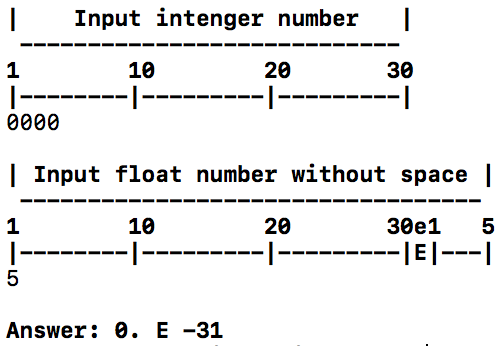


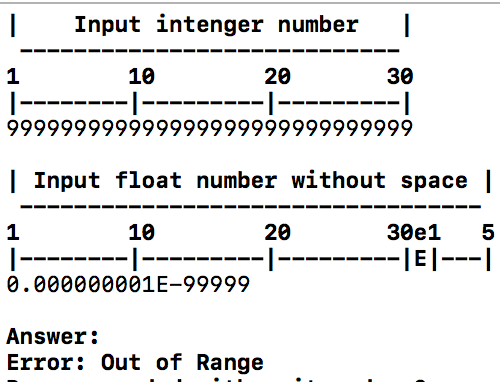


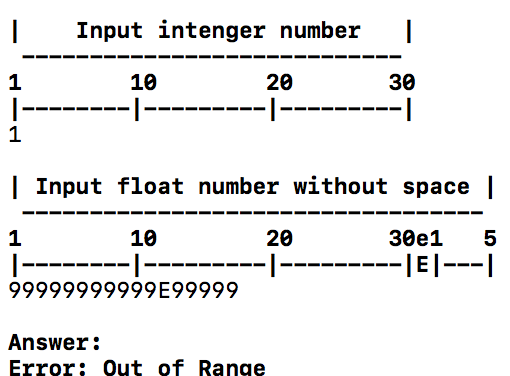
Проверка деления.

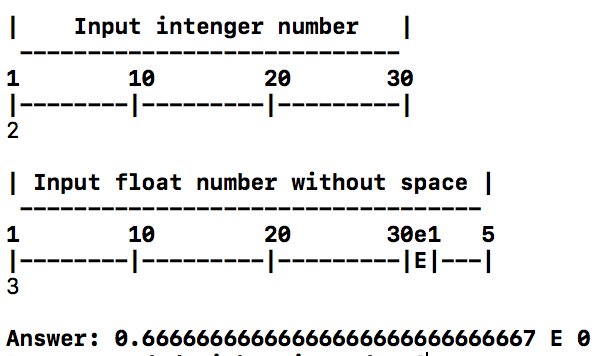


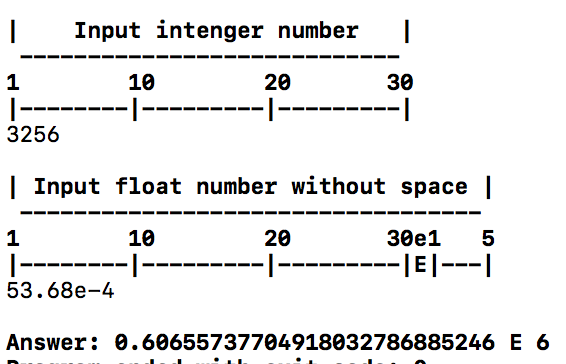


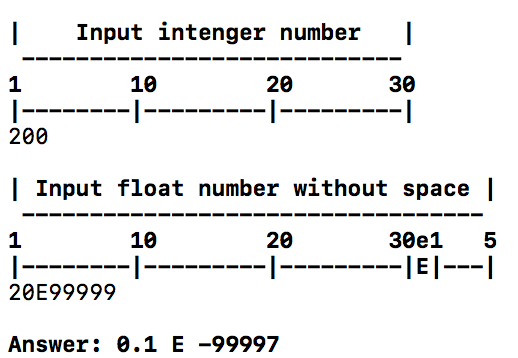


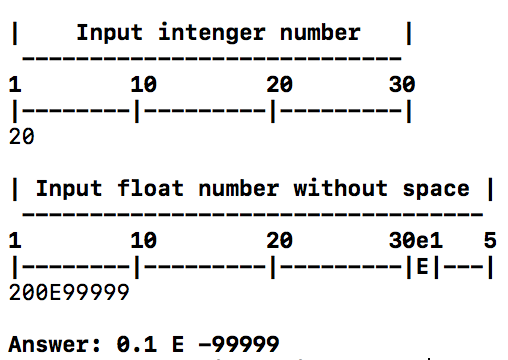


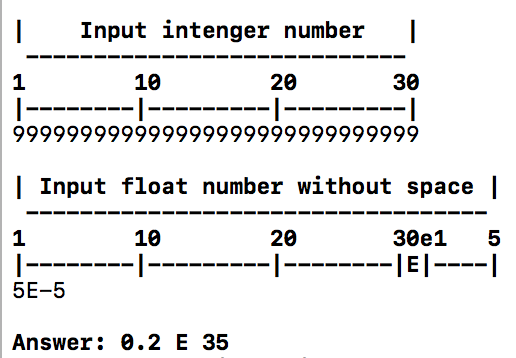












**Вывод:**

Для хранения длинных чисел были выбраны структуры, т.к. этот тип данных хранит всю информацию о длинном числе: мантисса, экспонента, знак. Само число хранится внутри структуры в массиве из элементов int. В ходе реализации поставленной задачи был создан тип данных для хранения и обработки длинных чисел, выходящих за пределы разрядной сетки персонального компьютера. Были реализованы арифметические операции над данными числами.

**Ответы на вопросы.**

1. Каков возможный диапазон чисел, представляемых в ПК?

Диапазон значений чисел зависит от размера области памяти, выделяемой под хранение переменной этого типа, от наличия знака в числе и от типа представления числа (целое или вещественное). Под хранение целого положительного числа, для которого выделено 64 разряда, максимально возможное значение числа равно 2^64 – 1 = 18 446 744 073 709 551 615.

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

Вещественные числа хранятся в представлении с плавающей точкой в виде X = M \* E^p, где M – мантисса, представленная правильной дробью (в интервале [0.1..1)). Точность представления вещественного числа зависит от максимально возможной длины мантиссы, которая зависит от области выделяемой памяти и наличия знака. Если длина мантиссы выходит за границы разрядной сетки, то происходит округление. Для 64-разрядного числа на мантиссу отводится 53 разряда.

3. Какие стандартные операции возможны над числами?

Сложение, вычитание, умножение, деление.

4. Какой тип данных может выбрать программист, если обрабатываемые числа

превышают возможный диапазон представления чисел в ПК?

Можно воспользоваться типами данных из дополнительно подключаемых библиотек, предназначенных для работы с большими числами. Также большие числа можно хранить в виде структуры, в которой можно поразрядно хранить число в массиве, а также информацию о знаке, количестве цифр и т.д..

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

Если это операция деления или умножения, то в таком случае операции проводятся поразрядно, как при вычислениях столбиком.