МГТУ им. Н.Э. Баумана

**Дисциплина Типы и Структуры данных.**

**Лабораторный практикум №1**

**по теме: «**Обработка больших чисел**»**

Работу выполнила:

студентка группы ИУ7-35Б

Шелия София

Работу проверил

**Цель работы**: реализация арифметических операций над числами, выходящими за разрядную сетку персонального компьютера, выбор необходимых типов данных для хранения и обработки указанных чисел.

1. **Описание условия задачи. Вариант№4.**

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

1. **Техническое задание.**

**Исходные данные и результат.**

**Ввод.**

С консоли вводится строка (делимое). Mаксимальное количество цифр в строке 30. Перед строкой можно ставить знак делимого(+/-) или не ставить (по умолчанию +).

*Примеры корректного ввода:* -123239

+12345678932626722736826628

2

С консоли вводится строка(делитель). Максимальная длина мантиссы – 30, максимальная длина порядка – 5.

Перед строкой можно ставить знак делителя(+/-) или не ставить (по умолчанию +).

Экспонента может присутствовать в делителе или быть опущена.

Допускается запись экспоненты в двух формах: e, E.

После экспоненты можно ставить знак (+/-) или не ставить (по умолчанию +).

Допускается запись без точки.

*Примеры корректного ввода: -13537*

*+2*

*367*

*-123456789123456789123456789123e+91234*

*+932E-56789*

*123456789123455478945123456789123*

*+338.3838*

*-3387e20*

*+3387e-20*

*-3387e+20*

*35.478E-12*

*0.000025*

*.000025*

*+13638.*

**Вывод.**

Результат деления выводится в формате (+/-)0.Me(+/-)K, где M – максимальная длина мантисcы – 30, а K – максимальная длина порядка – 5.

*Примеры вывода: +0.23434e+45*

*-0.374e-12*

*+0.12345679381387237232083083134e-12345*

**Описание задачи, реализуемой программой.**

Программа получает на вход два числа: целое и действительное. Далее она:

1. Нормализует эти числа.
2. Выводит нормализованные числа на экран.
3. Выполняет деление одного числа на другое.
4. Выводит нормализованный результат на экран.

**Cпособ обращения к программе.**

Запуск приложения возможен через терминал MSYS2, а именно.

1. gcc -std=c99 -Wall -Werror -c \*.c
2. gcc -o main.exe \*.o
3. ./main.exe

**Описание возможных аварийных ситуаций и ошибок пользователя.**

**Ошибки пользователя при вводе делимого.**

* Использование посторонних символов

*Пример*: 123ewe34

* Ввод не целого числа.

*Пример:* 123.44

* Некорректный ввод числа.

*Пример: -12+56*

* Количество символов выходит за пределы допустимых.

Пример: *+12345678912345678912345678912335*

* Пустой ввод.

**Ошибки пользователя при вводе делителя.**

* Использование посторонних символов

*Пример*: 12.3ewe34

* Количество символов выходит за пределы допустимых.

*Примеры*: *+12345678912345678912345678912335*

*-32898e2324420*

* Некорректный ввод числа.

*Пример: -12+56*

* Пустой ввод.

**Аварийные ситуации.**

* Переполнение порядка результата. Если в результате деления порядок получился меньше -99999 или больше 99999.
* Введён делитель равный 0.

1. **Описание внутренних структур данных.**

Для решения данной задачи существуют несколько способов хранения данных, например, в виде структуры или массива. Мной был выбран вариант хранения в виде структуры по нескольким причинами:

* Число условно разделено на 3 части: знак мантиссы, мантисса, порядок. При таком разделении для восприятие более удобно отдельное хранение каждой части.
* Раздельное хранение мантиссы и порядка упрощает работу с ними.

**Сама структура (используется в программе для делимого, делителя и результата деления):**

struct **number**

{

int mantisa\_sign; //Знак мантиссы

int mantisa\_field[31];//Элементы мантисcы

int order;//Порядок.

};

1. **Описание алгоритма.**

Сначала производится ввод данных. Для считывания используется getchar(), который позволяет уловить все возможные ошибки при вводе данных. Далее значения добавляются в соответствующие им поля структуры. Из числа удаляются лидирующие нули и происходит его нормализация.

После этого начинается деление чисел. Для его реализации мною были написаны несколько дополнительных функций.

* multiplication\_by\_number – умножает массив на заданную цифру (от 1 до 9) результат записывает в другой массив.
* subtraction – Выполняет “вычитание” массивов.
* compare\_the\_numbers – Сравнивает делимое с делителем.
* comparison\_with\_the\_result\_of\_multiplication **–** Сравнивает результат умножения с делимым.
* find\_order\_of\_result **–** Cчитает значение порядка результата до выполнения деления.

Для решения задачи мной было реализовано “деление столбиком”. Алгоритм реализации.

1. Порядку результата присваивается значение равное разности порядков делимого и делителя.
2. Определяется знак мантиссы результата.
3. Запускается цикл, который завершается при выполнении одного из двух условий: размер мантиссы делимого <=0 или размер мантиссы результата >30.
4. Далее происходит развилка для двух случаев: если делимое больше делителя (то есть мы ищем цифры стоящие “до точки”, и если делимое меньше делителя (то есть мы дошли до остатка, цифры “после точки”)

Для первого случая.

* Проверяется сколько цифр с делимого нужно “опустить вниз”, чтобы выполнить деление. Если значение > 1, то в результат в соответствующие место добавляются нули.
* Сравниваются первые n элементов делимого и делителя, где n – длина делителя. Если делимое < делителя, то это значит, что результат умножения делителя на нужную цифру может оказаться на единицу длинней, чем сам делитель, что важно учесть в дальнейшем. (Если делимое > делителя lack = 0, иначе lack = 1);
* Дальше запускается цикл в теле которого происходит умножения делителя на цифру. Цикл завершается, если выполняется какое-либо из условий: длина делителя + lack > длины результата умножения или длина делителя + lack = длине результата, но при этом часть делимого длинной (длина делителя + lack) меньше чем результат умножения.
* После завершения цикла соответствующая цифра записывается в мантиссу результата и значение длинны мантиссы результата увеличивается на 1.
* Далее выполняется вычитание из делимого результата умножения, с последующим смещением цифр влево при получении лидирующих нулей.
* Также в конце производится проверка меньше ли длина делимого длины делителя (то есть переходим ли мы на следующем шаге ко второму случаю), если да, то в порядок результата добавляется длина мантиссы (то есть количество цифр перед точкой)

Для второго случая производятся аналогичный действия, но с учетом того что само делимое по сути закончилось и для выполнения деления мы постоянно “дописываем” нули.

1. После завершения цикла происходит проверка порядка. Если он <-99999 или >+99999, то работа программы завершается и выводится сообщение о переполнении порядка.
2. Далее также происходит проверка длинны мантиссы результата, если она = 31, то производится соответствующее округление (если 31-й разряд больше или равен 5, то к 30-му разряду добавляется единица, если меньше 5, то 31-й разряд отбрасывается).

**5. Тесты.**

**Негативные тесты:**

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Результат** | **Условие проверки** |
| divisible: erre | Incorrect input. | Ввод посторонних символов для делимого |
| divisible: 123451  diviser: 545rtr | Incorrect input. | Ввод посторонних символов для делителя |
| divisible: 134.67 | Incorrect input. | Ввод не целого делимого |
| divisible: -13-45 | Incorrect input. | Некорректный ввод делимого. |
| divisible: 123451  diviser: -134.-45 | Incorrect input. | Некорректный ввод делителя. |
| divisible: | Incorrect input. | Пустой ввод |
| divisible: 12345123451234  51234512345123453 | Incorrect input. | Количество символов в делимом выходит за пределы допустимых. |
| divisible: 123451  diviser:  -1234567891234567891234567891234 | Incorrect input. | Количество символов в мантиссе делителя выходит за пределы допустимого. |
| divisible: 123451  diviser: -1345.90e73236 | Incorrect input. | Количество символов в порядке делителя выходит за пределы допустимого. |
| divisible: 123451  diviser: 0 | Division cannot be made. | Деление на 0. |
| divisible: 125757  diviser: 454.4647e-99999 | Overflow. | Переполнение порядка результата. |
|  |  |  |
| divisible: 999999999999999999999999999998  diviser: 999999999999999999999999999999 | Overflow. | Переполнение мантиссы результата. |

**Положительные тесты:**

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Результат** | **Условие проверки** |
| divisible: 34534536  diviser: -1456.65e-26 | **-**0.237081907115642055401091545669e+31 | Обычный тест. |
| divisible: 2  diviser: 999999999999999999999999999999 | +0.1e-29 | Самое короткое делить на самое длинное |
| divisible: 9999  diviser: 9999 | +0.1e+1 | Деление двух равных чисел |
| divisible: 1  diviser: 5678.7868e-34 | +0.176093950207815514398251401161e+31 | Деление 1 на число |
| divisible: 19  diviser: 1000000000 | +0.19e-7 | В результата деления получаются лидирующие нули |
| divisible: 2  diviser: 103 | +0.194174757281553398058252427184e-1 | В результате деления 31 цифра мантиссы <5 |
| divisible: 2  diviser: 106 | +0.18867924528301886792452830189e-1 | В результате деления 31 цифра мантиссы >=5 |

**6. Выводы по проделанной работе.**

В ходе выполнения работы мною были сделаны следующие выводы:

* Процессоры не могут обрабатывать и производить математические операции с длинными числами, реализация данных действий ложится на программиста.
* Операции умножения и деления длинных чисел выполняются с помощью стандартных школьных алгоритмов умножения и деления в столбик.

**Ответы на контрольные вопросы.**

1. Каков возможный диапазон чисел, представляемых в ПК?

**Для целых чисел** зависит от количества выделенных разрядов. Для 64-разрядного процессора невозможно использовать больше 20 десятичных разрядов для представления числа, так как - 1=18 446 744 073 709 551 615

**Для вещественных чисел.**

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.

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

Точность представления чисел определяет длина мантиссы, а порядок в свою очередь ограничивает диапазон допустимых значений.

Под представление мантиссы отводится 52 разряда, под порядок – 11. Таким образом, возможные значение чисел находятся в диапазоне от 3.6 E –4951 до 1.1 E +4932

3. Какие стандартные операции возможны над числами?

Сложение, вычитание, умножение и деление чисел.

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

* Массив (массив символов или чисел – для ввода и вывода, числовой массив – для обработки)
* Структуру (число делится на знак мантиссы, мантиссу и порядок)
* Можно разбить мантиссу на несколько частей, далее обработать, а в конце “склеить” обратно.

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

Так как процессоры не могут обрабатывать длинные числа, то операции с ними следует выполнять с помощью алгоритмов умножения, деления, вычитания и сложения в столбик.