Лабораторная работа №1

Дремин Кирилл, студент ИУ7-36Б

Описание условия задачи

Необходимо реализовать арифметические операции над числами, выходящими за разрядную сетку ПК, выбрав и разработав необходимые типы данных для хранения и обработки данных чисел. Требуется смоделировать операцию деления простого числа на действительное число, где порядок имеет до 5 разрядов (от -99999 до 99999), а мантисса - до 30 знаков.

Описание ТЗ

Описание исходных данных и результатов:

(типы, форматы, точность, способ передачи, ограничения) Программа получает на вход два значения. Первое значение является целым числом, оно может содержать до 30 десятичных цифр и (опционально) знак '+'/'-'.

Второе значение является действительным числом. Оно вводится в формате +/-m.n E +/-K, где суммарная длина мантиссы m+k <= 30, а величина порядка K - до 5 цифр (т.е. порядок принимает значения от -99999 до +99999)

Результат выводится в формате +/-0.m E +/-K, где m - мантисса до 30 значащих цифр, а K - порядок до 5 цифр

Описание задачи, реализуемой программой

Программа производит операцию деления первого введённого (целого) числа на второе (действительное) и выводит результат в нормализованной форме, либо сообщает о невозможности произвести счёт.

Способ обращения к программе

Обращение к программе происходит путём вызова скомплированного исполняемого файла в консольном режиме

Описание возможных аварийных ситуаций и ошибок пользователя

Аварийные ситуации:

1. Результат деления не попадает под ограничения выводимого формата (происходит в случае, если абсолютное значение порядка превышает 99999)
2. Второй введённый параметр равен нулю (ошибка деления на ноль) Ошибки пользователя:
3. Ввод одного из параметров в некорректном формате (для первого параметра - попытка ввода вещественного числа, а также лишние символы в числе)
4. Ввод некорректных данных (нераспознаваемые символы в потоке ввода)

Описание внутренних СД

Основной тип, используемый в программе - полиморфный тип числа длинной арифметики.

```
typedef struct { sign_t sign; int value[len]; } name
```

Данный тип полиморфен относительно своего названия и размера (кол-ва цифр.)

Данный тип также определяет универсальный операции над своими реализациями:

```
// сравнение абсолютного значения
#define abs_compare(type, a, b) abs_compare_##type(a, b)
// вычитание из абсолютного значения первого числа абсолютного значения второго числа
#define abs_sub(type, a, b, res) abs_sub_##type(a, b, res)
// добавление к абсолютному значению первого числа абсолютного значения второго числа
#define abs_add(type, a, b, res) abs_add_##type(a, b, res)
// добавление к первому числу второго числа
#define add(type, a, b, res) add_##type(a, b, res)
// вычитание из первого числа второго числа
#define sub(type, a, b, res) sub_##type(a, b, res)
// присваивание значения первого числа второму числу
#define copy(type, from, to) copy_##type(from, to, true)
```

В программе используются следующие конкретные реализации данного типа:

- **mantissa_t** длиной 30
- **mantissa_large_t** длиной в два раза больше **mantissa_t**
- **exponent_t** длиной 5
- **exponent_large_t** длиной на 1 больше **exponent_t**

Из данных типов составляются структуры для различных форм представления вещественных чисел в программе:

mantissa_t и exponent_t составляют собой тип real_t, который является типом, используемым для предоставления результатов вычисления арифметических операций.

mantissa_t и exponent_large_t составляют собой тип real_input_t, используемый для сохранения результатов ввода (расширенная экспонента требуется для обработки её переполнения в случаях ввода наподобие 10000E99999)

mantissa_large_t и exponent_large_t составляют собой тип real_inner_t, используемый для внутреннего представления результатов вычисления, также позволяет максимально точно сохранить результат, который в дальнейшем может быть преобразован в real_t

Описание алгоритма

Основные алгоритмы в программе - ввод числа и деление.

Ввод числа осуществляется путём посимвольного считывания ввода и дальнейшего его анализа. Используется принцип работы конечного автомата с переходами между состояниями в зависимости от текущего состояния и введённого символа. По мере ввода символов значение типа real_input_t постепенно заполняется, определяются знаки и значения мантиссы и экспоненты.

Деление числа происходит с использованием алгоритма деления в столбик (с учётом особенностей реализации операций над используемами структурами данных).

Основной принцип работы алгоритма: Имеем на входе вещественные числа A и B, необходимо получить R = A / B Создадим переменную R и присвоим ей значение 0 Далее пока числа A и B оба не равны нулю, производим следующие действия: Пока A >= B, вычитаем B из A и прибавляем единицу к R. Когда A становится меньше B, умножаем R на 10 и делим B на 10 (в представлении в виде цифр - сдвигаем вправо на 1), контролируя выход за пределы представления вещественного числа.

Когда A или B станет равно нулю, завершаем. Итоговый результат, сохранённый в R будет мантиссой результата. Порядок же результата высчитывается как разность порядка A и порядка B.

Набор тестов с указанием проверямого параметра

Ввод	Вывод	Что проверяется
100 и -0.1	- 0.1 E + 4	деление чисел, отличающихся множителем на степень десяти + деление положительного на отрицательное
0 и +1.5 E -415	+ 0.0 E + 0	числитель равный нулю => результат равен нулю
-150 и -0.3	+ 0.5 E + 3	деление отрицательного числа на отрицательное
1 и 60	0.1666...6667 E - 1	обработка округления
99999999999999999999999999999 и 11111111111111111111111111111	+0.9 E +1	деление с использованием предельно большого целого числа
999999999999999999999999999999 и 2	+0.5 E +30	циклическое округление
10 и 100 E 99999	+0.1 E -99999	наименьший возможный результат
10000 и +0.1 E -99993	+0.1 E +99999	наибольший возможный результат
Пустой ввод	Error: empty input!	обработка пустого ввода
100 и 0	Error: zero division!	обработка деления на ноль
1000 и 1 E -99998	Error: division leads to overflow!	проверка обработки результата, не попадающего под ограничения обрабатываемого типа
10 и ffad	Error: parsing error!	Проверка ввода некорректных данных
10.5 и 1.5 E +3	Error: integer number parsing error	Проверка ввода первого параметра не целым числом, а вещественным
9999999999999999999999999999999 (31 цифра '9')	Error: overflow of format in input!	слишком большое значение введённого числа

Выводы

В случаях, когда необходимо проводить арифметические операции над числами повышенной точности и/или размера целесообразно использовать длинную арифметику.

Длинная арифметика может быть смоделирована путём представления чисел в виде массива цифр, что позволит легко реализовать различные операции, такие как сложение, вычитание, сравнение, учёт переноса значений между разрядами.

Для операций над длинной арифметикой можно использовать классические алгоритмы, аналогичные таковым действиями с обычными числами при счёте, это обусловлено тем, что представление в виде массива цифр равнозначно представлению числа в десятичной системе счисления, в отличие от основания 2 в обычных числах.

Ответы на вопросы

1. Каков возможный диапазон чисел, представляемых в ПК?

Целые числа (беззнаковые):

Выделенные разряды	Диапазон
16	0...65 535
32	0...4 294 967 295
64	0...18 446 744 073 709 551 616

Целые числа (со знаком):

Выделенные разряды	Диапазон
16	-32768...32767
32	-2 147 483 648...2 147 483 647
64	-9 223 372 036 854 775 808...9 223 372 036 854 775 807

Вещественные числа:

Выделенные разряды	Диапазон
32 (single precision)	3.4E-38...3.4E+38
64 (double precision)	1.7E-308...1.7E+308
80 (extended precision)	3.4E-4932...3.4E+4932

Диапазон значений вещественного числа ограничен длиной порядка мантисы

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

Возможная точность представления чисел в случае вещественных чисел определяется длиной мантиссы числа, для 32-битного вещественного числа с мантиссой длиной 23 бита, точность составляет 7-8 десятичных цифр, для 64-битного вещественного числа с мантиссой длиной 52 бита, точность составляет 15-16 десятичных цифр.

3. Какие стандартные операции возможны над числами?

Над числами возможны арифметические и логические операции Арифметические: Сложение, вычитание, унарный плюс и минус, инкремент и декремент, умножение и деление, для целых - деление по модулю Логические: Сравнение, для целых: исключающее ИЛИ, логическое И, ИЛИ, побитовое отрицание

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

Для обработки таких чисел возможно использовать особым образом устроенный массив цифр, используемый для представления вещественного чисал (с мантиссой, порядком) с основанием системы счисления 10

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

При использовании выше описанного представления чисел можно реализовать основные арифметические операции, смоделировав их наивным образом (соответствующе ручному счёту), что будет довольно просто вследствие использования системы счисления с основанием 10