МГТУ им. Н. Э. Баумана

Отчет по лабораторной работе №1 по предмету «Типы и структуры данных»
«Длинная арифметика: обработка больших чисел»

Выполнила Параскун София,
ИУ7-34Б
Проверила
Силантьева А. В.

Москва, 2020 г.

1. Описание условия задачи

Составить программу умножения целого числа длиной до 30 десятичных цифр на действительное число в форме +-m.n E +-K, где суммарная длина мантиссы (m+n) - до 30 значащих цифр, а величина порядка K - до 5 цифр. Результат выдать в форме +-0.m1 E +-K1, где m1 - до 30 значащих цифр, а K1 - до 5 цифр. Программа должна осуществлять ввод чисел и выдавать либо верный результат в указанном формате (при корректных данных), либо сообщение о невозможности произвести счет.

2. Описание ТЗ, включающего внешнюю спецификацию

a) описание исходных данных и результатов

На вход мы получаем два числа — целое в формате «знак» (не обязателен) и «число» (не более 30 цифр) и вещественное число в формате +-m.n E +-K, где суммарная длина мантиссы (m+n) не более 30 значащих цифр, величина порядка K не более 5 цифр. Причем для вещественного числа знак не обязателен, порядок вводится с помощью «E» (английская) до которой и после может быть не больше одного пробела.

Результат представляет собой число в форме +-0.m1 E +-K1, где m1 - до 30 значащих цифр, а K1 - до 5 цифр.

Пользователь вводит числа через консоль, после вывода соответствующей подсказки.

В случае ввода некорректных данных или невозможности вычислений, об этом сообщается пользователю через консоль.

b) Способ обращения к программе

Программа запускается через консоль, после чего можно увидеть приглашения ввода для первого числа — целого. После успешного ввода появится второе приглашение на ввод вещественного числа. Если и это число было введено корректно, программа выведет результат или сообщение об ошибке, если это невозможно.

c) Описание возможных аварийных ситуаций и ошибок пользователя

№ ситуации	Аварийная ситуация	Реакция программы
1	Ввод некорректного по подсказке целого числа	Прекращение работы и сообщение «Incorrect integer number!»
2	Ввод некорректного по подсказке вещественного числа	Прекращение работы и сообщение «Incorrect float number!»
3	Ввод данных, приведших к переполнению порядка	Прекращение работы и сообщение «Order overflow!»

3. Описание внутренних СД

Необходимо обеспечить хранение данных, превышающих по значению стандартные типы. Для целых чисел создается отдельная структура, содержащая знак и сами значащие цифры, для вещественных — знак мантиссы, значащие цифры мантиссы, порядок.

Значение константы MAX_LEN равно 30.

Первой структурой является целое число integer_num, содержащее поля sign, mant_len и mantissa.

```
struct integer_num
{
    short sign;
    short mant_len;
    short mantissa[MAX_LEN];
};
```

Sign отвечает за знак числа, mant_len - за кол-во значащих цифр числа, а mantissa - за сами значащие цифры.

Второй структурой является вещественное число float_num, содержащее поля sign, mant_len, mantissa и order.

Sign отвечает за знак числа, mant_len - за кол-во значащих цифр мантиссы, mantissa – за сами значащие цифры мантиссы, а order – за порядок вещественного числа (в том числе знак порядка).

```
struct float_num
{
    short sign;
    short mant_len;
    short mantissa[MAX_LEN];
    int order;
};
```

4. Описание алгоритма

Внутри основной функции происходит считывание сначала целого числа, при его успешном вводе вещественное, если же и оно было введено верно, то происходит умножение двух чисел, после чего вывод результата, если в ходе вычислений не было получено ошибки.

Для считывания и записи целого числа функция input_int вызывает вспомогательную функцию int_num_parser, записывающей значащие цифры в поле mantissa структуры first_num.
Для считывания и записи вещественного числа функция input_float вызывает функцию float_num_parser, которая в свою очередь производит запись в структуру second_num. Также внутри этой функции происходит вызов order_parser, которая работает с порядком вещественного числа (если он задан в формате «E k1», где k1 – порядок числа).

Умножение чисел производится функцией multi. С помощью функции zxz первое число умножается на каждую цифру второго (таким образом получается arr_of_mult), после чего функция sum_of_mult складывает полученные результаты (result_of_mult), и полученное число записывается в third_num, которое является структурой float_num.

После всего этого, если не произошло ошибки вычисления, с помощью функции output число third_num выводится в консоль в требуемом формате.

5. Набор тестов, с указанием, что проверяется

Так как корректная работа программы зависит от введенных пользователем данных, сначала рассмотрим классы эквивалентности, связанные с входными данными.

№ теста	Входные данные	Что проверяется?	Реакция программы
1	1difw	Наличие недопустимых символов в целом числе	Incorrect integer number!
2	1.258	Ввод вещественного числа под видом целого	Incorrect integer number!
3	-13284	Ввод целого числа со знаком минус	*продолжение ввода*
4	+13284	Ввод целого числа со знаком плюс	*продолжение ввода*
5	12345678901234567890123456789	Ввод целого длиной 30 цифр	*продолжение ввода*
6	123456789012345678901234567890	Ввод целого длиной 31 цифра	Number so long. Incorrect integer number!
7	-12345678901234567890123456789	Ввод целого длиной 30 со знаком	*продолжение ввода*
8	-123456789012345678901234567890	Ввод целого длиной 31 со знаком	Number so long. Incorrect integer number!
9		Пустое целое число	Empty number. Incorrect integer number!
9	003	Целое число с ведущими нулями	Number must be without head-0. Incorrect integer number!
10	12	Пустое вещественное число	Empty number. Incorrect float number!
11	12 -1kvkls E-10	Наличие недопустимых символов в мантиссе	Incorrect float number!
12	12 -1.422 E1jf	Наличие недопустимых символов в порядке	Incorrect float number!
13	12 .00025	Вещественное число, начинающееся с точки	0.3 E -2
14	12 123001.	Вещественное число, заканчивающееся точкой	0.1476012 E 7
15	12 123.456	Вещественное число без знака и без порядка	0.1481472 E 4

16	12 -123 E 10	Вещественное число со знаком и пробелами до и после экспоненты	-0.1476 E 14
17	12 123E -5	Вещественное число без знака, без пробела перед экспонентой и со знаком порядка после пробела	0.1476 E -1
18	12 -123 E+6	Вещественное число со знаком, пробелом перед экспонентой, без пробела после экспоненты, со знаком порядка	-0.1476 E 10
19	12 123.829E61	Вещественное число с точкой, без пробелов перед и после экспоненты	0.1485848 E 65
20	12 1.23456789012345678901234567890	Мантисса вещественного числа 30 цифр	0.14814813681481481 3681481481368 E 2
21	12 1.234567890123456789012345678901	Мантисса вещественного числа 31 цифра	So long mantissa. Incorrect float number!
22	12 - 1.23456789012345678901234567890	Вещественное число со знаком, мантисса 30 цифр	0.14814813681481481 3681481481368 E 2
23	12 - 1.234567890123456789012345678901	Вещественное число со знаком, мантисса 31 цифра	So long mantissa. Incorrect float number!
24	12 1.3 E99999	Порядок вещественного числа содержит 5 цифр	Order overflow!
25	12 1.3 E100000	Порядок вещественного числа содержит 6 цифр	So long order. Incorrect float number!
26	12 1.3 E -99999	Порядок вещественного числа записан через пробел, со знаком и 5 цифрами	0.156 E -99997
27	12 - 1.23456789012345678901234567890 E +99999	Вещественное число со знаком, мантиссой длиной 30 цифр, пробел до и после экспоненты, порядком со знаком из 5 цифр	Order overflow!

Теперь рассмотрим случаи корректного ввода данных и возможные исходы работы программы.

№ теста	Входные данные	Что проверяется?	Реакция программы
1	12 -123 E+6	Числа с маленьким кол-вом цифр и порядком (меньшим максимального)	-0.1476 E 10
2	12 - 1.23456789012345678901234567890 E +99999	Максимально возможный порядок	Order overflow!
3	100 0.0001E-99999	Минимально возможный порядок	Order overflow!
4	100 0.001E-99999	При нормализации порядок переполняется, но результат является корректным	0.1 E -99999
5	0 1.2	Целое число 0	0
6	12 0.0	Вещественное число 0	0
7	77777777777777777777777777777 5	Длина значащей части целого числа 30 (ожидается округление)	0.388888888888888888888888889 E 31
8	22222222222222222222222222222 7	Длина значащей части целого числа 30 (ожидается отбрасывание)	0.155555555555555555555555555 E 31

6. Выводы по проделанной работе

В ходе этой лабораторной удалось поработать с форматами данных, выходящих за пределы обычного использования (целые числа с 30 значащими цифрами, вещественные с мантиссой 30 цифр). Также были написаны особые алгоритмы под используемые структуры данных, что дало понять как важно выбрать подходящий тип данных для исполнения реализуемого алгоритма.

Ответы на контрольные вопросы

1. Какой возможный диапазон чисел, представляемых в ПК?

Для int отводится 4 байта, для short – 2 байта. У обоих типов первый бит слева отводится под знак числа. Для типа int диапазон значений от -2^31 до 2^31 – 1 (от -2 147 483 648 до 2 147 483 647), для типа short от -2^15 до 2^15 – 1 (от -32 768 до 32 767).

Для float с плавающей точкой - 4 байта, для double с плавающей точкой - 8 байт, для мантиссы float - 23 бита, для экспоненты float - 8 бит, для мантиссы double - 52 бита, для экспоненты double - 11 бит.

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

Возможная точность представления числа зависит от разрядности используемой системы. Диапазон изменения чисел определяется количеством разрядов, отведенных для хранения порядка числа, а точность (кол-во значащих цифр) определяется количеством разрядов, отведенных для хранения мантиссы.

3. Какие стандартные операции возможны над числами?

Числа можно складывать, вычитать, умножать, делить, сравнивать (>, <), округление.

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

Можно выбрать массив типа short/int, чтобы не потерять преимуществ работы с числами.

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

Написать функции, которые будут производить необходимые операции для вышеприведенных массивов типа short/int.