Лабораторная работа №1
по дисциплине «Типы и структуры данных»
Тема : Длинная арифметика.

Условие задачи:

Составить программу умножения двух чисел - целого, количество разрядов которого не больше 30, и вещественного, порядок которого находится в диапазоне от –99999 до +99999 (т.е. имеет не более 5 разрядов), а длина мантиссы не превышает 30 разрядов.

Программа должна осуществлять ввод чисел в указанном диапазоне значений и выдавать результат в нормализованной форме ±0.m1 E ±K1, где число m1 определено до 30 значащих цифр, число K1 – до 5 цифр. При невозможности произвести вычисления должно выдаваться соответствующее сообщение.

Исходные данные:

целое число:

знак:

тип *char*

представление *“0” или “1”*

размер *1 символ*

число:

тип *array of integer*

представление *массив целых чисел*

размер *1-30 цифр*

вещественное число:

знак:

тип *char*

представление *“0” или “1”*

размер *1 символ*

мантисса:

тип *array of integer*

представление *массив целых чисел*

размер *1-30 цифр*

порядок:

тип *integer*

представление *число от -99999 до 99999*

размер *5 разрядов*

Выходные данные:

знак:

тип *char*

представление *“-” или “+”*

размер *1 символ*

мантисса:

тип array of integer

представление *массив целых чисел*
размер *1-30 цифр*

порядок:

тип *integer*
представление *число от -99999 до 99999*
размер *5 разрядов*

Ввод:

При вводе пользователем целое число может иметь вид:

а) +1234
б) 1234
в) -1234

При вводе пользователем вещественное число может иметь вид:

а) 12.3E123 (или -12.3E-123 или +12.3E+123 -> любые комбинации знаков)
б) 123E123 // без “.”
в) 123.E123
г) .123E123
д) 0.123E123
е) 12.3 // без “E”

Примечание : символ “E” вводится в верхнем регистре транслитом

Вывод:

Вывод имеет формат:

знак_мантиссы**0.**мантисса**E**порядок

Примеры:

+0.123E-15
-0.123E+15

Возможные ошибки пользователя:

Ввод вещественного числа с порядком, при приведении которого к виду мантисса**E**порядок, порядок выходит за границы его представления в программе (< -99999 || >99999)

Реакция: сообщение “Ошибка вычисления”

Ввод таких данных, при которых порядок результата умножения выйдет за границы его представления в программе (< -99999 || >99999)

Реакция: сообщение “Ошибка вычисления”

Некорректный ввод целого числа (наличие букв и символов, проверка на несколько введенных знаков + или -)

Реакция: сообщение “Целое число введено неверно”

Некорректный ввод вещественного числа (наличие букв и символов, проверка лишние знаки +, -, E и точку)

Реакция: сообщение “Вещественное число введено неверно”

Алгоритм

1. Считывание целого числа
2. Обработка строки ввода целого числа
    a. Запись знака в char
    b. Запись самого числа в array of integer
3. Считывание вещественного числа
4. Обработка строки ввода вещественного числа
    a. Запись знака в char
    b. Запись мантиссы в array of integer
    c. Запись порядка в integer
5. Выполнение умножения
6. Запись знака результата в char
7. Обработка мантиссы результата
8. Печать результата

Функции

Функции чтения числа

*void input_int_numbers(char *number, char *znak)*
*void input_float_numbers(char *number, char *znak)*
Параметры:
number *строка из ввода*
znak *знак числа*

Функция обработки целого числа

*void integer_array_generate(const char *array_char, int *array_int, int *counter)*
Параметры:
array_char *строка, содержащая целое число без знака*
array_int *массив цифр целого числа*
counter *количество цифр в целом числа*

Функция обработки вещественного числа

void float_array_generate(const char *array_char, int *array_int, int *exponent, int *counter, int *flag_point, int *flag_e)
Параметры:
array_char строка с вещественным числом без знака
array_int мантисса вещественного числа
exponent порядок вещественного числа
counter длина мантиссы
flag_point количество встречающихся в числе “.”
flag_e количество встречающихся в числе “E”

Функция умножения двух чисел

void counting(const int *array_first, int first_len, const int *array_second, int second_len, int *result)
Параметры:
array_first массив символов первого множителя
first_len длина первого множителя
array_second массив символов второго множителя
second_len длина второго множителя
result массив символов результата умножения

Псевдокод функции:

```
pointer целое число
for i = second_len-1 to 0 do
    pointer = 0;
    for j = first_len-1 to 0 do
        result[i+j+1] += pointer + array_second[i]*
                                                array_first[j];
        pointer = result[i+j+1] // 10;
        result[i+j+1] %= pointer;
        j -= 1;
    result[i] += pointer;
    i -= 1;
```

Функция обработки результата

void normalize(int *array, int *result, int *exponent, int count)

Параметры:

array массив символов результата умножения
result обработанный массив символов результата
exponent порядок результата
count максимальное число символов в результате

Назначение :

Функция редактирует часть вещественного числа, выделенную жирным +0.**123**E+123 и изменяет порядок числа, увеличив его на количество чисел в этой части для дальнейшей передачи в функцию печати.

Функция округления мантиссы

void rounding(int *array)

Параметры:

array массив символов мантиссы

Псевдокод:

```
pointer = 1;
for i = 30 to 0 do
    result[i] = (pointer + array[i-1]) % 10
    pointer = (pointer + array[i-1]) // 10
result[0] = pointer;
```

Назначение:

Функция округляет мантиссу до тридцати разрядов. Если 31-ый элемент >= 5, то округление происходит в большую сторону, затрагивая предыдущие разряды.

Функция печати результата

void print_float(char znak, int *mantissa, int exponent, int len_mantissa)

Параметры:

znak знак результата **+**0.123E+123
mantissa мантисса результата +0.**123**E+123
exponent порядок результата +0.123E**+123**
len_mantissa длина мантиссы

Тестовые данные

1. *Одно или оба числа равны 0*

Целое число	Вещественное число	Результат
0	1.23E+123	0.0
123	0.0	0.0
0	0	0.0

2. *Стандартные входные данные*

Целое число	Вещественное число	Результат
100	10.1E-2	+0.101E+2
20	-.202	-0.404E+1
50	-100.E10	-0.5E+14

3. *Округление*

Целое число	*Вещественное число*	*Результат*
99999999999999999999999999 (30 девяток)	*2.E0*	*+0.2E+30*
2	*-777777777777777777777777777. (30 семерок)*	*-0.1555555555555555555555555555E+30*

4. *Границы порядка*

Целое число	*Вещественное число*	*Результат*
10	*0.1E99998*	*+0.1E+99999*
10	*0.1E99999*	*Ошибка вычисления*

5. *Нормализация*

Целое число	*Вещественное число*	*Результат*
1	*0012.34*	*+0.1234E+2*
1	*12.3400*	*+0.1234E+2*
1	*0012.3400*	*+0.1234E+2*

6.Некорректный ввод

Целое число	*Вещественное число*	*Результат*
++1	*+-1.1E1*	*Целое число введено неверно / Вещественное число введено неверно*
1-	*+1.1E1+abc*	
1abc	*+1.e1*	
+1.1E+1	*1.1.1.1E+1E*	

Ответы на вопросы к лабораторной работе

Каков возможный диапазон чисел, представляемых в ПК?

Диапазон чисел, представляемых в ПК зависит от разрядности процессора. Если процессор имеет 32 разряда, то максимальное значение составит 2^{32}-1 = 4 294 967 295. Для 64 разрядов максимально возможное значение числа равно 2^{64}-1 = 18 446 744 073 709 551 615.

Какова возможная точность представления чисел?

Для 64-разрядного процессора принципиально невозможно использовать больше 20 десятичных разрядов для представления целого числа или больше 20 знаков после точки в мантиссе для вещественного.

Какие стандартные операции возможны над числами?

Над числами возможны стандартные арифметические операции: сложение, вычитание, умножение, деление.

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

Наиболее предпочтительный тип данных - массив: массив символов – для ввода числа, массив чисел – для обработки.

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

Руководствуясь стандартными алгоритмами арифметических операций, таких как умножение и деление в столбик, где числа рассматриваются поразрядно.