

# 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  $\pm 0.m_1 E \pm K_1$ , где число  $m_1$  определено до 30 значащих цифр, число  $K_1$  – 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\_мантиисы**0**.мантииса**E**порядок

*Примеры:*

+0.123E-15  
-0.123E+15

### **Возможные ошибки пользователя:**

Ввод вещественного числа с порядком, при приведении которого к виду мантииса**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”

### **Алгоритм**

1. Считывание целого числа
2. Обработка строки ввода целого числа
  - а. Запись знака в char
  - б. Запись самого числа в array of integer
3. Считывание вещественного числа
4. Обработка строки ввода вещественного числа
  - а. Запись знака в char
  - б. Запись мантиисы в array of integer

- с. Запись порядка в integer
5. Выполнение умножения
6. Запись знака результата в char
7. Обработка мантиссы результата
8. Печать результата

## Функции

### Функция чтения числа

void input\_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 числе

### Функция обработки вещественного числа

void float\_array\_generate(const char \*array\_char, int \*array\_int, int \*exponent, int \*counter)

Параметры:

*array\_char* строка с вещественным числом без знака

*array\_int* мантисса вещественного числа

*exponent* порядок вещественного числа

*counter* длина мантиссы

### 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  $\geq 5$ , 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
123	0	0
0	0	0

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

