МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №4

по дисциплине: Основы программирования тема: «Преобразование типов»

Выполнил: ст. группы ПВ-201 Машуров Дмитрий Русланович

Проверил:

Притчин Иван Сергеевич Брусенцева Валентина

Станиславовна

Белгород 2021 г.

**Лабораторная работа № 4**

**«Преобразование типов»**

**Цель работы**: получение навыков преобразования последовательности символов в числовое значение и наоборот.

**Задания для подготовки к работе:**

1. Изучить, в каких случаях и по каким правилам в языке Си преобразования типов выполняются автоматически.
2. Изучить возможности для явного преобразования типов в языке Си.
3. Разработать алгоритм и составить программу для решения задачи соответствующего варианта. Необходимые преобразования описать функциями. Библиотечные функции для ввода и вывода числовых значений и функции преобразования типов не использовать.
4. Подобрать наборы тестовых данных.

**Задание варианта №17**

С клавиатуры вводится последовательность вещественных чисел, числа разделены запятыми. Конец ввода – конец файла. Записать введенную последовательность в строку в форме с фиксированной точкой, сохранив три знака после точки.

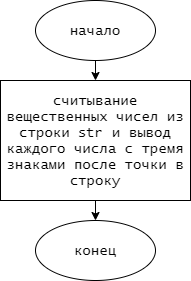
1. **Описание алгоритма и выделение подзадач**

Будем считывать введённую строку в переменную, затем считывать числа.

Можно считывать число целиком и запоминать количество знаков после точки. Далее считанное число будем делить на , чтобы перенести запятую на необходимое количество знаков.

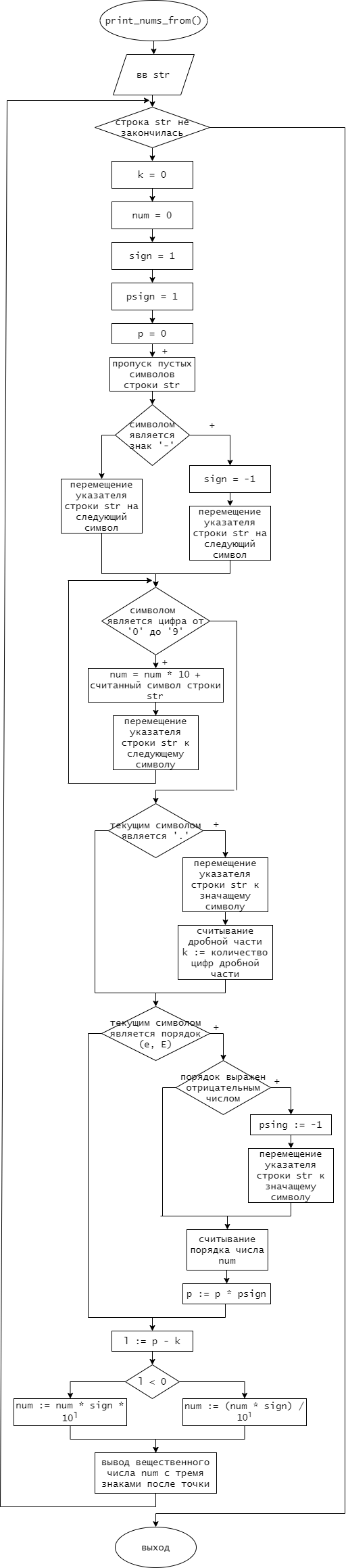
Выделим следующие подзадачи:

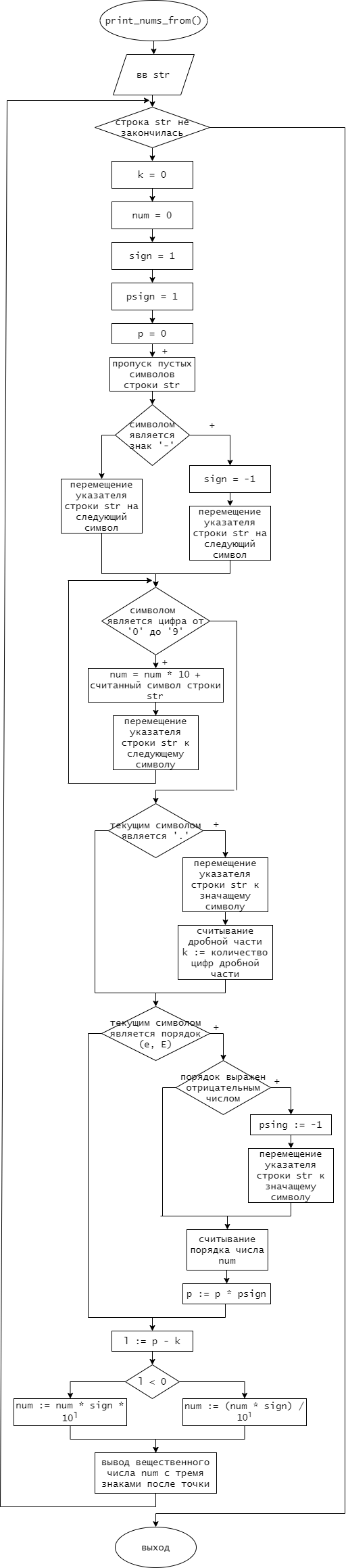
1. Считывание вещественных чисел и вывод их с тремя знаками после точки
2. **Блок-схема с укрупнёнными блоками**

****

1. **Описание подпрограмм**
2. Считывание вещественных чисел и вывод их с тремя знаками после точки
3. Заголовок: void print\_nums\_from()
4. Назначение: считывает вещественные числа из потока ввода и выводит их с тремя знаками после точки

Блок-схема:

****

****

1. **Тестовые данные**

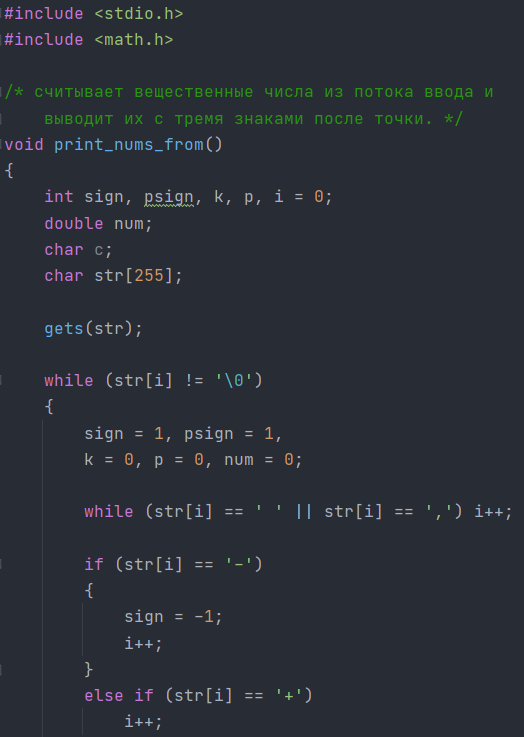
|  |  |  |
| --- | --- | --- |
| **№** | **Вход** | **Выход** |
| **1** | «123.2314, 343.22, 1.1» | 123.231 343.22 1.100 |
| **2** | «-8.56, -3333.3, -123.43123» | -8.560 -3333.300 -123.431 |

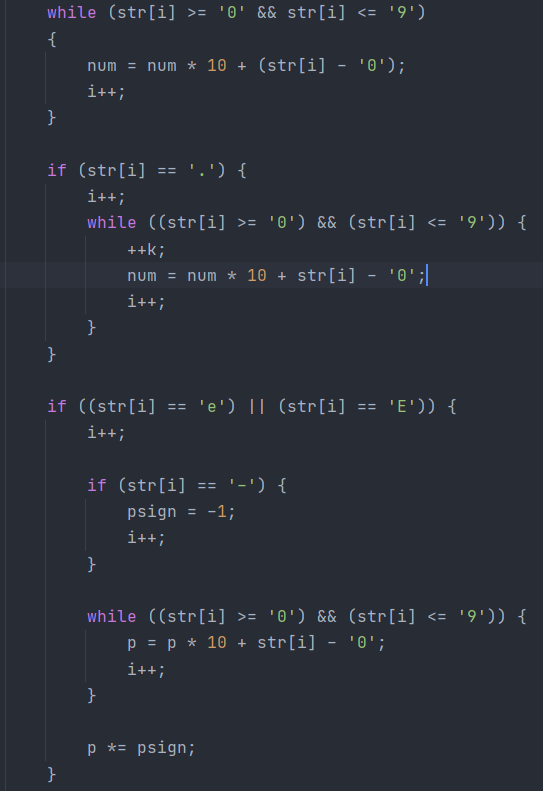
1. **Текст программы**

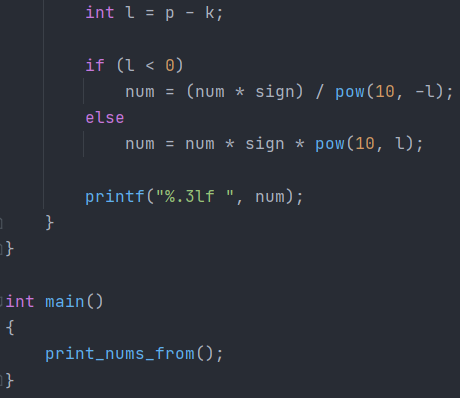
/\*  
 \* С клавиатуры вводится последовательность  
 \* вещественных чисел, числа разделены запятыми.  
 \* Конец ввода – конец файла.  
 \* Записать введенную последовательность в строку в форме  
 \* с фиксированной точкой, сохранив три знака после точки  
 \*/  
  
#include <stdio.h>  
#include <math.h>  
  
/\* считывает вещественные числа из потока ввода и  
 выводит их с тремя знаками после точки. \*/  
void print\_nums\_from()  
{  
 int sign, psign, k, p, i = 0;  
 double num;  
 char c;  
 char str[255];  
  
 gets(str);  
  
 while (str[i] != '\0')  
 {  
 sign = 1, psign = 1,  
 k = 0, p = 0, num = 0;  
  
 while (str[i] == ' ' || str[i] == ',') i++;  
  
 if (str[i] == '-')  
 {  
 sign = -1;  
 i++;  
 }  
 else if (str[i] == '+')  
 i++;  
  
 while (str[i] >= '0' && str[i] <= '9')  
 {  
 num = num \* 10 + (str[i] - '0');  
 i++;  
 }  
  
 if (str[i] == '.') {  
 i++;  
 while ((str[i] >= '0') && (str[i] <= '9')) {  
 ++k;  
 num = num \* 10 + str[i] - '0';  
 i++;  
 }  
 }  
  
 if ((str[i] == 'e') || (str[i] == 'E')) {  
 i++;  
  
 if (str[i] == '-') {  
 psign = -1;  
 i++;  
 }  
  
 while ((str[i] >= '0') && (str[i] <= '9')) {  
 p = p \* 10 + str[i] - '0';  
 i++;  
 }  
  
 p \*= psign;  
 }  
  
 int l = p - k;  
  
 if (l < 0)  
 num = (num \* sign) / pow(10, -l);  
 else  
 num = num \* sign \* pow(10, l);  
  
 printf("%.3lf ", num);  
 }  
}  
  
int main()  
{  
 print\_nums\_from();  
}

1. **Результаты работы и скрины программы**

*Программа:*







*Пример №1:*



*Пример №2:*



1. **Анализ допущенных ошибок**

* нет