

Основной задачей программирования является обработка информации, поэтому любой язык программирования имеет средства для ввода и вывода информации. В языке Си нет операторов ввода-вывода.

Ввод и вывод информации осуществляется через функции стандартной библиотеки. Прототипы рассматриваемых функций находятся в файле `stdio.h`. Эта библиотека содержит функции

- ▣ `printf()` — для вывода информации
- ▣ `scanf()` — для ввода информации.

### Вывод информации

Функция `printf()` предназначена для форматированного вывода. Она переводит данные в символьное представление и выводит полученные изображения символов на экран. При этом у программиста имеется возможность форматировать данные, то есть влиять на их представление на экране.

Общая форма записи функции `printf()`:

```
printf("СтрокаФорматов", объект1, объект2, ..., объектn);
```

**СтрокаФорматов** состоит из следующих элементов:

- ▣ управляющих символов;
- ▣ текста, представленного для непосредственного вывода;
- ▣ форматов, предназначенных для вывода значений переменных различных типов.

Объекты могут отсутствовать.

**Управляющие символы** не выводятся на экран, а управляют расположением выводимых символов. Отличительной чертой управляющего символа является наличие обратного слэша `'\'` перед ним.

Основные управляющие символы:

- ▣ `'\n'` — перевод строки;
- ▣ `'\t'` — горизонтальная табуляция;
- ▣ `'\v'` — вертикальная табуляция;
- ▣ `'\b'` — возврат на символ;
- ▣ `'\r'` — возврат на начало строки;
- ▣ `'\a'` — звуковой сигнал.

**Форматы** нужны для того, чтобы указывать вид, в котором информация будет выведена на экран. Отличительной чертой формата является наличие символа процент `'%'` перед ним:

- ▣ `%d` — целое число типа `int` со знаком в десятичной системе счисления;
- ▣ `%u` — целое число типа `unsigned int`;
- ▣ `%x` — целое число типа `int` со знаком в шестнадцатеричной системе счисления;

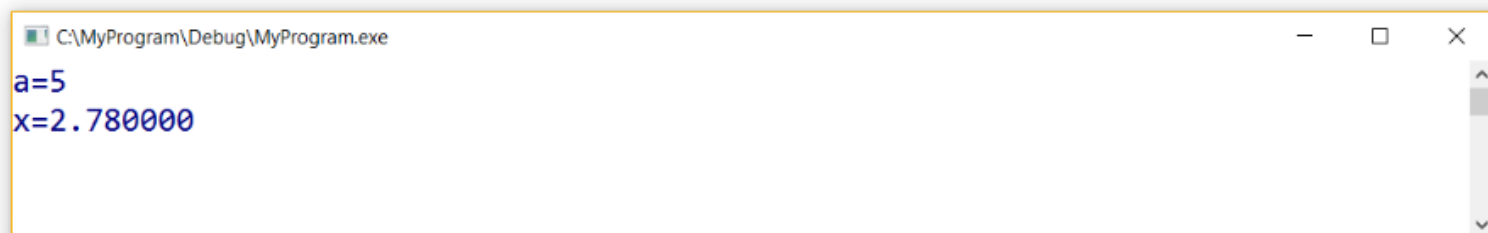
- ❑ `%o` — целое число типа `int` со знаком в восьмеричной системе счисления;
- ❑ `%hd` — целое число типа `short` со знаком в десятичной системе счисления;
- ❑ `%hu` — целое число типа `unsigned short`;
- ❑ `%hx` — целое число типа `short` со знаком в шестнадцатеричной системе счисления;
- ❑ `%ld` — целое число типа `long int` со знаком в десятичной системе счисления;
- ❑ `%lu` — целое число типа `unsigned long int`;
- ❑ `%lx` — целое число типа `long int` со знаком в шестнадцатеричной системе счисления;
- ❑ `%f` — вещественный формат (числа с плавающей точкой типа `float`);
- ❑ `%lf` — вещественный формат двойной точности (числа с плавающей точкой типа `double`);
- ❑ `%e` — вещественный формат в экспоненциальной форме (числа с плавающей точкой типа `float` в экспоненциальной форме);
- ❑ `%c` — символьный формат;
- ❑ `%s` — строковый формат.

Строка форматов содержит форматы для вывода значений. Каждый формат вывода начинается с символа `%`. После строки форматов через запятую указываются имена переменных, которые необходимо вывести. Количество символов `%` в строке формата должно совпадать с количеством переменных для вывода. Тип каждого формата должен совпадать с типом переменной, которая будет выводиться на это место. Замещение форматов вывода значениями переменных происходит в порядке их следования.

#### Пример на Си

```
1  #include <stdio.h>
2  int main()
3  {
4      int a = 5;
5      float x = 2.78;
6      printf("a=%d\n", a);
7      printf("x=%f\n", x);
8      getchar();
9      return 0;
10 }
```

Результат работы программы



Тот же самый код может быть представлен с использованием одного вызова `printf`:

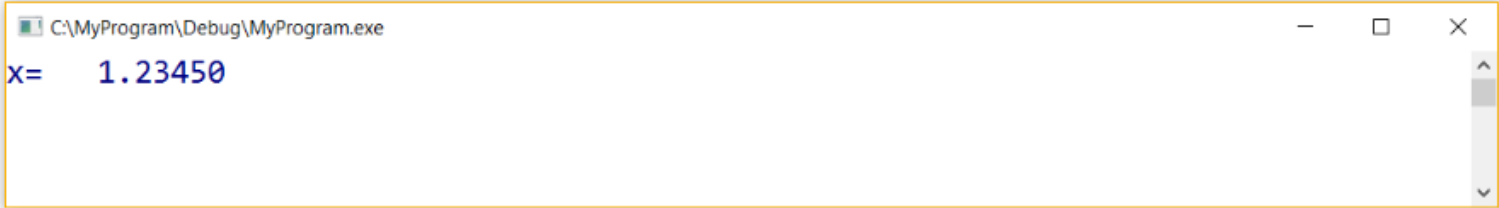
```
1  #include <stdio.h>
2  int main()
3  {
4      int a = 5;
5      float x = 2.78;
6      printf("a=%d\nx=%f\n", a, x);
7      getchar();
8      return 0;
9  }
```

### Табличный вывод

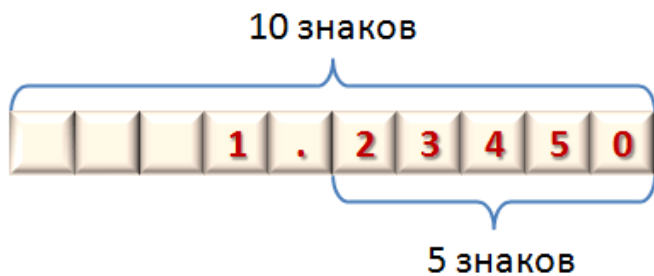
При указании формата можно явным образом указать общее количество знакомест и количество знакомест, занимаемых дробной частью:

```
1  #include <stdio.h>
2  int main()
3  {
4      float x = 1.2345;
5      printf("x=%10.5f\n", x);
6      getchar();
7      return 0;
8  }
```

Результат выполнения



В приведенном примере 10 — общее количество знакомест, отводимое под значение переменной; 5 — количество позиций после разделителя целой и дробной части (после десятичной точки). В указанном примере количество знакомест в выводимом числе меньше 10, поэтому свободные знакоместа слева от числа заполняются пробелами. Такой способ форматирования часто используется для построения таблиц.



### Ввод информации

Функция форматированного ввода данных с клавиатуры scanf() выполняет чтение данных, вводимых с клавиатуры, преобразует их во внутренний формат и передает вызывающей функции. При этом программист задает правила интерпретации входных данных с помощью спецификаций форматной строки. Общая форма записи функции scanf( ):

```
scanf ("СтрокаФорматов", адрес1, адрес2,...);
```

Строка форматов аналогична функции printf().

Для формирования адреса переменной используется символ амперсанд '&':

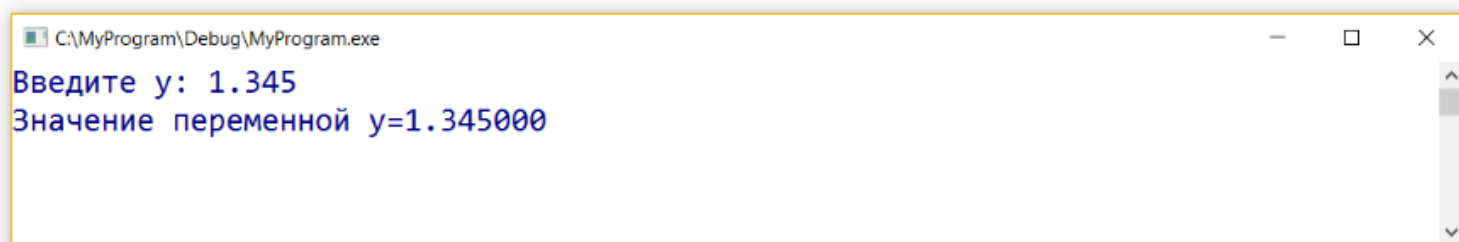
адрес = &объект

Строка форматов и список аргументов для функции обязательны.

### Пример на Си

```
1  #define _CRT_SECURE_NO_WARNINGS // для возможности использования scanf
2  #include <stdio.h>
3  #include <stdlib.h> // для перехода на русский язык
4  int main()
5  {
6      float y;
7      system("chcp 1251"); // переходим в консоли на русский язык
8      system("cls");       // очищаем окно консоли
9      printf("Введите y: "); // выводим сообщение
10     scanf("%f", &y);      // вводим значения переменной y
11     printf("Значение переменной y=%f", y); // выводим значение переменной y
12     getchar(); getchar();
13     return 0;
14 }
```

Результат работы программы:



Функция `scanf( )` является функцией незащищенного ввода, т.к. появилась она в ранних версиях языка Си. Поэтому чтобы разрешить работу данной функции в современных компиляторах необходимо в начало программы добавить строчку

```
#define _CRT_SECURE_NO_WARNINGS
```

Другой вариант — воспользоваться функцией защищенного ввода `scanf_s( )`, которая появилась несколько позже, но содержит тот же самый список параметров.

```
1  #include <stdio.h>
2  int main()
3  {
4      int a;
5      printf("a = ");
6      scanf_s("%d", &a);
7      printf("a = %d", a);
8      getchar(); getchar();
9      return 0;
10 }
```