# **Стандартные функции форматированного ввода и вывода**

Для использования этих функций в начало программы нужно поставить директиву препроцессора **#include <stdio.h>.** Она включает содержимое файла в текст программы. Этот файл содержит описание стандартных типов и функций для работы со стандартными файлами.

Форматный вывод – функция **PRINTF().**

Эта функция предназначена для форматного вывода данных в стандартный файл вывода. Обычный стандартный файл вывода связан с экраном дисплея, но этот файл можно связать с любым другим устройством вывода. В результате вывод стандартного файла означает вывод данных на соответствующее устройство. Стандартный файл вывода по структуре является текстовым файлом, т.е. представляет собой последовательность строк символов. Функция **printf** имеет вид:

**Int printf(форматная строка,[список элементов ввода])**

Первый параметр этой функции имеет тип: **char \*форматная строка**. Она выводит значения аргументов в соответствии с заданной форматной строкой.

Строка формата определяет, какой вид должны иметь выводимые данные. Строка содержит следующие компоненты:

* Текстовые строки (отображаются в таком виде, в каком стоят в строке формата);
* Управляющие последовательности(специальные символы, предназначенные для управления выводом);
* Спецификации вывода состоят из знака % и символа.

## Управляющие последовательности

|  |  |
| --- | --- |
| **Последовательность** | **Значение** |
| \a | Звуковой сигнал |
| \b | Возврат на 1 символ назад |
| \f | Прогон страницы |
| \n | Перевод строки |
| \r | Возврат каретки |
| \t | Горизонтальная табуляция |
| \v | Вертикальная табуляция |
| \\ | Обратная косая черта |
| \? | Знак вопроса |
| \’ | Одинарная кавычка |
| \” | Двойная кавычка |

Строка формата должна содержать по одной спецификации вывода для каждой выводимой переменной. Функция **printf()** отображает значение каждой переменной в таком виде, какой указан соответствующей спецификацией.

## Спецификации

|  |  |
| --- | --- |
| **Спецификация** | **Тип переменной** |
| %c | Char |
| %d | Int, Short |
| %ld | Long |
| %f | Float, Double |
| %s | Массивы типа Char(строки символов) |
| %u | Unsigned Int, Unsigned Short |
| %lu | Unsigned Long |

Все элементы строки формата, не являющиеся спецификациями или управляющими последовательностями, воспроизводятся так, как текстовые строки и отображаются так, как написаны, включая пробелы.

В одной строке можно вывести любое количество значений переменных, но строка формата обязательно должна содержать по одной спецификации для каждого значения.

**Printf(“a=%f, b=%d”,a,b);**

**Printf(“%d”, x+y);**

Между аргументами и спецификациями устанавливается взаимнооднозначное соответствие в порядке их следования слева на право.

Любая спецификация в форматной строке имеет следующий общий вид:

**% [выравнивание] [ширина (или \* )] [ дополнительные признаки] символ преобразования**

Выравнивание при выводе значения осуществляется в право по умолчанию, т.е. если под выводимое значение отводится большее число позиций чем нужно, то лишние позиции слева заполняются пробелами. Если место символа выравнивания поставить **« - »** то выравнивание будет осуществляться влево (т.е. значения будут прижиматься к левой границе).

Ширина определяет количество позиций, необходимых при выводе значения, если она отсутствует, то под значение будет отводиться стандартное число позиций. Если \*, то количество позиций будет соответствовать выводимому значению.

**Примеры**

1. Вывод символа

**% [ - ] [ ширина] c**

d = ‘A’; printf(“%-4c”,d); результат А\_\_\_

1. Вывод строки

**% [ - ] [ ширина] [. точность]s**

s1=’123456789’; printf(“%9.3s”,s1); результат \_\_\_\_\_\_123

здесь параметр точность определяет число печатаемых символов строки.

1. Вывод целого числа со знаком

**% [ - ] [±][ ширина] [l] d**

printf(“ I = %5ld\n”,i);

Если **+** то значение будет снабжаться знаком, даже если оно положительное, **l** в случае типа **Long**

1. Вывод целого числа без знака

**% [ - ] [#][ ширина] [l] {u, o, x, X}**

# определяет вывод начального нуля в восьмеричном формате и 0х или 0Х в шеснадцатиричном.

U – означает вывод в 10-й системе счисления без знака

О - означает вывод в 8-й системе счисления

Х, х - означает вывод в 16-й системе счисления (х тогда a-f, X тогда A-F)

1. Вывод чисел с плавающей точкой

**% [ - ] [±] [#][ ширина] [. Точность] {f, e, E, g, G}**

# значения выводятся с завершающими не имеющими значения нулями

точность – количество цифр после точки

Если ширина и точность не заданы, то вывод осуществляется со стандартными значениями этих параметров.

F – вывод значения в виде целой и дробной частей.[-]d…d.d…d

e – вывод в формате [-]d.d…de[-]dd

E - вывод в формате [-]d.d…dE[-]dd

g – вывод осуществляется соответственно с f или e в соответствии с наиболее короткой формой.

G – вывод осуществляется соответственно с f или E в соответствии с наиболее короткой формой.

Вывод сообщений с помощью функции **PUTS()**

Она используется для вывода текстовых сообщений, но её нельзя использовать для отображения значений переменных. При её выводе на экран в конце автоматически выполняется переход на новую строку.

**Puts(“Hello, word!”);** эквивалентна **printf (“Hello, word!\n”);**

Форматный ввод - функция **SCANF()**

Для вывода из стандартного файла используется функция scanf(). Функция имеет вид:

**Int scanf(форматная строка, список указателей аргументов)**

Форматная строка имеет ту же структуру, что и форматная строка функции printf. За форматной строкой через запятые перечисляются указатели на переменные, значения которых будут вводиться в стандартный файл. Форматная строка может содержать пустые символы, литерные и спецификации преобразования формата. К пустым относятся: пробел, знаки табуляции, символы перевода формата и конца строки. Пустые символы не учитываются и не оказывают влияния. Пробелы во входном потоке данных (т.е. во вводимых записях) воспринимаются как разделители отдельных полей. Если форматная строка содержит литерные символы, не входящие в число пустых и не %(спецификации) то при вводе предполагается, что соответствующие символы должны появляться во входном потоке данных в соответствующих позициях. Спецификации преобразования формата имеют следующий общий вид:

**% [\*] [ширина ] [ дополнительные признаки] символ преобразования**

\* обозначает пропуск при вводе поля, определенного данной спецификацией, т.е. информация введенная по соответствующей спецификации не будет присваиваться никакой переменной. Ширина определяет максимальное число символов вводимых по данной спецификации. В число спецификаций входят:

1. **%[\*][ширина]c** – осуществляет ввод символов, ширина определяет количество символов, которые будут прочитаны из входного потока и будут присвоены соответствующему массиву символов. Если ширина опущена, то вводится 1 символ. Можно вводить и пустые символы.
2. **%[\*][ширина]s –** для ввода строк символов. Ширина задает максимальную длину вводимой строки. Строки во входном потоке должны разделяться пустыми символами.
3. **%[\*][ширина][l, h][d,u,o,x] –** предназначены для ввода целых чисел.

l – ввод целого типа long

h - ввод целого типа short

по умолчанию предполагается ввод типа int

d – ввод десятичного целого

u – ввод беззнакового целого

o – ввод восьмеричного целого без знака

x – ввод целого без знака в 16- ой системе счисления.

1. **%[\*][ширина][l][f,e,g] –** ввод значений с плавающей запятой. L задает ввод типа double, если не задан, то тип float. F, e, g – синонимы, т.е. предназначены для ввода значения с плавующей точкой.

Спецификация ввода по образцу.

**%[\*][ширина][образец]**

Образец определяет множество символов, из которых может состоять вводимая строка. Он задается строкой символов, заключенной в [ ].

**[a, b, c, d]** или **[a – z]** или **[a – f, 0 - 9]**

если на первом месте образца стоит « ^ » вводятся все символы из входного потока кроме перечисленных [^a-d].

По спецификациям преобразования задаваемых образцом вводится строка символов, разрешенных этим образцом, до тех пор, пока не встретятся другой символ, ведущие пустые символы пропускаются.

Функция scanf возвращает количество прочитанных и присвоенных полей значений.

### Примеры

**1. Int I;**

**Foat x;**

**Char name[50];**

**Scanf(“%d %f %s”, &i, &x, name);**

Вводим строку = 25 54.32е-1 Thompson

В результате получим I=25; x=54.32\*10-1; name= Thompson

(к строке Thompson автоматически будет добавлен признак конца строки)

**2. Int I;**

**Foat x;**

**Char name[50];**

**Scanf(“%2d %f %\*d %2s”,&i, &x, name);**

Вводим строку = 56789 0123 45a72

В результате получим I=56; x=789.0; name= 45;

#### Замечание

*При разработке программ рекомендуется по возможности реже использовать функцию scanf, а пользоваться getc, gets и другими. Они более просты и выполняются с более быстрой скоростью*.