**printf conversion characters**

|  |  |
| --- | --- |
| **Character** | **Prints expression as** |
| c | single character |
| d | decimal number |
| e | [-]***d.dddddd***E[+-]***dd*** |
| f | [-]***ddd.dddddd*** |
| g | e or f conversion, whichever is shorter, with nonsignificant zeros suppressed |
| o | unsigned octal number |
| s | string |
| x | unsigned hexadecimal number |
| % | print a %; no argument is converted |

Below are some examples of **printf** statements along with the corresponding output:

printf "%d", 99/2 49

printf "%e", 99/2 4.950000e+01

printf "%f", 99/2 49.500000

printf "%6.2f", 99/2 49.50

printf "%g", 99/2 49.5

printf "%o", 99 143

printf "%06o", 99 000143

printf "%x", 99 63

printf "|%s|", "January" |January|

printf "|%10s|", "January" | January|

printf "|%-10s|", "January" |January |

printf "|%.3s|", "January" |Jan|

printf "|%10.3s|", "January" | Jan|

printf "|%-10.3s|", "January" |Jan |

printf "%%" %

Format specifications are strings with % markers indicating format placeholders. Format placeholders consist of: %[flags][width][.precision][type]where the type is interpreted as in the following table:

|  |  |
| --- | --- |
| **Type** | **Description** |
| %b | Formats a **bool**, formatted as **true** or **false**. |
| %c | Formats a character. |
| %s | Formats a **string**, formatted as its contents, without interpreting any escape characters. |
| %d, %i | Formats any basic integer type formatted as a decimal integer, signed if the basic integer type is signed. |
| %u | Formats any basic integer type formatted as an unsigned decimal integer. |
| %x | Formats any basic integer type formatted as an unsigned hexadecimal integer, using lowercase letters a through f. |
| %X | Formats any basic integer type formatted as an unsigned hexadecimal integer, using uppercase letters A through F. |
| %o | Formats any basic integer type formatted as an unsigned octal integer. |
| %e, %E, %f, %F, %g, %G | Formats any basic floating point type (**float**, **float32**) formatted using a C-style floating point format specifications. |
| %e, %E | Formats a signed value having the form [-]d.dddde[sign]ddd where d is a single decimal digit, dddd is one or more decimal digits, ddd is exactly three decimal digits, and sign is + or -. |
| %f | Formats a signed value having the form [-]dddd.dddd, where dddd is one or more decimal digits. The number of digits before the decimal point depends on the magnitude of the number, and the number of digits after the decimal point depends on the requested precision. |
| %g, %G | Formats a signed value printed in f or e format, whichever is more compact for the given value and precision. |
| %M | Formats a [Decimal](http://msdn.microsoft.com/en-us/library/system.decimal.aspx) value. |
| %O | Formats any value, printed by boxing the object and using its **ToString** method. |
| %A, %+A | Formats any value, printed with the default layout settings. Use %+A to print the structure of discriminated unions with internal and private representations. |
| %a | A general format specifier, requires two arguments. The first argument is a function which accepts two arguments: first, a context parameter of the appropriate type for the given formatting function (for example, a [TextWriter](http://msdn.microsoft.com/en-us/library/system.io.textwriter.aspx)), and second, a value to print and which either outputs or returns appropriate text.  The second argument is the particular value to print. |
| %t | A general format specifier, requires one argument: a function which accepts a context parameter of the appropriate type for the given formatting function (a [TextWriter](http://msdn.microsoft.com/en-us/library/system.io.textwriter.aspx))and which either outputs or returns appropriate text. Basic integer types are **byte**, **sbyte**, **int16**, **uint16**,**int32**, **uint32**, **int64**, **uint64**, **nativeint**, and **unativeint**. Basic floating point types are **float** and **float32**. |

The optional *width* is an integer indicating the minimal width of the result. For instance, %6d prints an integer, prefixing it with spaces to fill at least 6 characters. If width is \*, then an extra integer argument is taken to specify the corresponding width.

Valid flags are described in the following table.

|  |  |
| --- | --- |
|  |  |
| 0 | Specifies to add zeros instead of spaces to make up the required width. |
| - | Specifies to left-justify the result within the width specified. |
| + | Specifies to add a + character if the number is positive (to match a - sign for negative numbers). |
| ' ' (space) | Specifies to add an extra space if the number is positive (to match a - sign for negative numbers). |
| # | Invalid. |