C 言語 005 printf 関数

【printf 関数】

printf 関数は標準入出力（Windows ではコマンドプロンプト）に文字列を出力する関数です。
下記は printf 関数の定義です。

定　　義	int printf(const char* format, ...);
ﾍｯﾀﾞｰﾌｧｲﾙ	stdio.h
説　　明	printf 関数は標準入出力（コマンドプロンプト）に文字列を出力する関数。 第 2 引数は複数指定できる。
戻 り 値	正常：出力した文字列のバイト数／異常：負数
例　　文	int var = 10;　　/* 変数 var に整数 10 を代入 */ printf("変数の値は%d です", var);　　/* 「変数の値は 10 です」と出力 */

【書式指定子】

書式指定子を使用して表示形式を指定できます。
書式指定子の記述は「"%[フラグ][最低表示桁数].[精度][型]"」です。
型の指定子以外は省略できます。

【型の指定子】

"%[フラグ][最低表示桁数].[精度]**[型]**"
変数の値をどの形式で出力するかを指定子で指定します。

指定子	出力したい形式
d、i	符号付き（signed）、10 進数の整数
u	符号無し（unsigned）、10 進数の整数
o（オー）	符号無し、8 進数の整数
x、X	符号無し、16 進数の整数
f、F	小数形式の実数
e、E	指数形式の実数
c	文字（1 文字）
s	文字列（1 文字以上）
p	ポインタ

指定子が「e／E」のように「小文字／大文字」になっているものは、小文字にすると出力するアルファベットが小文字に、大文字にすると出力するアルファベットが大文字になります。

Copyright 2018 DoubleQuotation Limited.

変数のデータ型によっては型の指定子の前に修飾子が付きます。

データ型	修飾子
short 型	h
long 型	l（エル）

＜例文 1＞

```
int var = 12;
printf("%d", var);    /* 10 進整数で「12」と出力 */
printf("%o", var);    /* 8 進整数で「14」と出力 */
printf("%x", var);    /* 16 進整数で「c」と出力 */
```

＜例文 2＞

```
double var = 10.123;
printf("%f", var);    /* 小数形式の実数で「10.123000」と出力 */
printf("%e", var);    /* 指数形式の実数で「1.012300e+001」と出力 */
```

＜例文 3＞

```
unsigned var1 = 1234;
short var2 = 1234;
long var3 = 1234;
printf("%u", var1);     /* unsigned 型なので%u で指定 */
printf("%hd", var2);    /* short 型なので%hd で指定 */
printf("%ld", var3);    /* long 型なので%ld で指定 */
```

＜例文 4＞

```
char var1 = 'a';           /* 文字 */
char var2[] = "abcde";    /* 文字列 */
printf("%c", var1);        /* 文字の出力 */
printf("%s", var2);        /* 文字列の出力 */
```

＜例文 5＞

```
int var1 = 10;
double var2 = 10.5;
printf("var1 は%d、var2 は%f", var1, var2);    /* 2 個の変数の値を出力 */
```

指定子と変数の順番を同じにすること。

Copyright 2018 DoubleQuotation Limited.
20181225

【最低表示桁数】

"%[フラグ]**[最低表示桁数]**.[精度][型]"

最低表示桁数を整数で指定します。

＜例文 1＞	出力結果
printf("%3d", 12345);	12345

「%3d」は「最低表示桁数が 3 桁」です。「表示桁数 3 桁」ではありません。

＜例文 2＞	出力結果
printf("%d", 10);　　/* A */	10
printf("%5d", 10);　　/* B */	10

B の方は、最低表示桁数 5 桁、第 2 引数の値は「10」で 2 桁なので、出力結果にはスペース 3 つを付けて 5 桁で出力されています。

【精度】

"%[フラグ][最低表示桁数].**[精度]**[型]"

実数の場合は「小数点の表示桁数」を、文字列の場合は「表示桁数」を整数で指定します。

＜例文 1＞	出力結果
printf("%f", 10.12345);	10.123450
printf("%.2f", 10.12345);	10.12

実数の場合は「小数の表示桁数」

＜例文 2＞	出力結果
printf("%s", "abcdefg");	abcdefg
printf("%.3s", "abcdefg");	abc

文字列の場合は「文字列の表示桁数」

＜丸めに注意＞

表示桁数が 2 であれば、小数点第 3 位の値で丸められます。

丸め方はコンパイラによって異なり、MinGW では「四捨五入」になります。

Copyright 2018 DoubleQuotation Limited.

【フラグの指定子】

"%[**フラグ**][最低表示桁数].[精度][型]"

フラグは表示形式にさまざまなオプションが付けれます。

指定子	オプション
+	数値の+-を出力する。
（空白）	正数の場合、先頭に空白を入れる。
-	左詰めで出力する。（通常は右詰め）
0	ゼロパディング（0 で埋める）する。
#	表記法に従った 8 進数、16 進数で出力する。

＜例文 1＞

```
printf("%d", -10);     /* 通常の負数 */
printf("%+d", 10);    /* 正数に+を付ける */
printf("% d", 10);     /* 正数に空白を付ける */
```

出力結果

```
-10
+10
 10
```

＜例文 2＞

```
printf("%05d", 10);      /* 出力 5 桁でゼロパディング */
printf("%-5d", 10);      /* 出力 5 桁で左詰め */
printf("%+05d", 10);     /* +マークも含めて出力 5 桁 */

/* ゼロパディング、最低出力桁数は全部で 10 桁、精度は 3 桁 */
printf("%010.3f", 10.12345);
```

出力結果

```
00010
10
+0010


000010.123
```


Copyright 2018 DoubleQuotation Limited.
20181225

【エスケープシーケンス】

「エスケープシーケンス」を使用して、出力する文字列の中に特殊な文字コードを埋め込むことができます。

エスケープシーケンス	コード
¥n	改行
¥t	タブ
¥b	バックスペース
¥r	キャリッジリターン
¥f	ページフィード
¥a	警告音
¥0（ゼロ）	NULL 文字

出力する文字に「"」「'」「¥」「?」がある場合、プログラムの記号と勘違いされて正しく出力できません。これらの文字を出力する場合は文字の頭に¥を付け「¥"」「¥'」「¥¥」「¥?」とします。

例文

```
printf("改行コードを入れる¥n");     /* 文字列の出力後に改行している */
printf("¥¥1000 です");              /* 「¥1000 です」と出力 */
```


Copyright 2018 DoubleQuotation Limited.
20181225