付録２
printf 関数と scanf 関数

ここでは、***printf*** 関数と ***scanf*** 関数を詳細に解説します。

2

printf 関数

■ 形式

```
#include  <stdio.h>
int printf(const char *format, ...);
```

▶ ... は、**省略記号**（*ellipsis*）と呼ばれ、任意の個数の仮引数を受け取ることを示す。

■ 機能

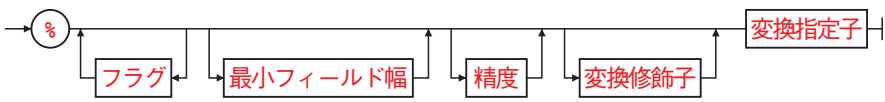
printf 関数は、*format* に続く実引数を、文字の並びである出力形式に変換して、標準出力ストリームへ出力する。その変換は、*format* が指す書式文字列内の指令にしたがう。書式文字列内には、指令が含まれていなくともよいし、複数含まれていてもよい。

▶ 書式に対して実引数が不足しているときの動作は、未定義とする。実引数が残っているにもかかわらず書式がつきてしまう場合、余分な実引数は評価するだけで無視する。

指令は、以下のいずれかである。

◆ 変換されずにそのまま出力ストリームに複写される % 以外の文字。

◆ 後続の実引数を 0 個以上取り出す変換指定（その構文は下図）。



フラグ

-, +, 空白, #, 0 によって、変換指定の意味を修飾する。0 個以上を指示でき、その順序は任意。

-	変換結果をフィールド内に**左詰め**にする。指定がなければ、右詰めとなる。
+	符号付き変換をされる数値の前に、**プラス符号**または**マイナス符号**を付ける。指定がなければ、負の値のみにマイナス符号が付けられる。
空白	符号付き変換の結果が符号で始まらない場合、または符号付き変換の結果の文字数が 0 の場合、数値の前に**空白**を付ける。 ▶ 空白フラグと + フラグの両方を指定すると、空白フラグは無効となる。
#	数値の表記形式（基数など）がわかるような変換を行う。 • o 変換 … 最初の数字を 0 にする（精度が増やされる）。 • x, X 変換 … 数値の前に 0x（あるいは 0X）を付加する（0 のときは付加しない）。 • e, E, f, g, G 変換 … 小数点文字の後ろに数字が続かなくても、常に小数点文字を付ける（通常は、小数点文字は数字が後続する場合に限り付けられる）。 • g, G 変換 … 後ろに続く 0 を結果から取り除かない。 • その他の変換での動作は未定義とする。

0	• d, i, o, u, x, X, e, E, f, g, G 変換 … フィールド幅の左側を、空白でなく 0 で詰める（ただし、符号や基数は、0 の前におかれる）。 • その他の変換での動作は未定義とする。 ▶ 0 フラグと - フラグの両方を与えた場合、0 フラグは無効となる。 ▶ d, i, o, u, x, X 変換において精度が指定された場合、0 フラグは無効となる。

2

最小フィールド幅

アステリスク * または 10 進整数。

値を変換した結果の文字数がこのフィールド幅より少ない場合、このフィールド幅を満たすまで左側（- フラグを指定した場合は右側）に、（0 フラグを指定しなければ）空白を詰める。

精度

ピリオド . にアステリスク * または 10 進整数が続く。10 進整数を省略した場合、0 が指定されたものとみなされる。各変換に対して、以下の指定を行う。

- d, i, o, u, x, X 変換 … 出力する最小の桁数。
- e, E, f 変換 … 小数点文字の後ろに出力する桁数。
- g, G 変換 … 最大の有効桁数。
- s 変換 … 最大の文字数。
- その他の変換での動作は未定義とする。

▶ フィールド幅、精度をアステリスクで指定したときは、それに対応する int 型の実引数が必要である（それは、変換される実引数の前に与えられなければならない）。フィールド幅を指定する実引数が負の場合、- フラグが前置された正のフィールド幅として解釈され、精度を指定する実引数が負の場合、精度を省略したものとして解釈される。

変換修飾子

h, l, L のいずれかの文字。省略可能。

h	• d, i, o, u, x, X 変換 … 実引数の型が **short** 型または **unsigned short** 型であることを指定する（実引数は汎整数拡張にしたがって拡張されている。その値を表示する前に **short** 型または **unsigned short** 型に戻す変換を行う）。 • n 変換 … 実引数の型が **short** 型へのポインタであることを指定する。
l	• d, i, o, u, x, X 変換 … 実引数の型が **long** 型または **unsigned long** 型であることを指定する。 • n 変換 … 実引数の型が **long** 型へのポインタであることを指定する。
L	• e, E, f, g, G 変換 … 実引数の型が **long double** 型であることを指定する。

変換修飾子を、上記以外の変換指定子とともに指定した場合の動作は未定義とする。

変換指定子

変換を指定する **d**, **i**, **o**, **u**, **x**, **X**, **f**, **e**, **E**, **g**, **G**, **c**, **s**, **p**, **n**, **%** の各文字。

d, **i**　**int** 型の実引数を [-]*dddd* 形式の符号付き 10 進表記に変換する。

精度は出力すべき数字の最少個数を指定する。値を変換した結果の数字の個数(桁数)が指定された精度より少ない場合は、その精度になるまで前に 0 を付ける。省略時の精度は 1 となる。値 0 を精度 0 で変換した結果の桁数は 0 となる。

o, **u**, **x**, **X**　**unsigned** 型の実引数を *dddd* 形式の符号無し 8 進表記 (**o**)、符号無し 10 進表記 (**u**)、符号無し 16 進表記（**x** または **X**）に変換する。**x** 変換では、文字 abcdef を用い、**X** 変換では、文字 ABCDEF を用いる。

精度は出力すべき最少の桁数を指定する。値を変換した結果の桁数が指定された精度より少ない場合は、その精度になるまで先行する 0 で拡張する。省略時の精度は 1 となる。値 0 を精度 0 で変換した結果の桁数は 0 となる。

f　**double** 型の実引数を [-]*ddd.ddd* 形式の 10 進表記に変換する。

このとき、小数点以下の桁数は、精度の指定に等しい。省略時の精度は 6 となる。精度に 0 が指定され、かつ **#** フラグが指定されていない場合は、小数点文字を出力しない。小数点文字を出力するときには、その前に必ず 1 個以上の数字を出力する。

この変換は、適切な桁数へ値の丸めも行う。

e, **E**　**double** 型の実引数を [-]*d.ddd***e** ± *dd* 形式の 10 進表記に変換する。

このとき、小数点文字の前に 1 桁の（実引数が 0 の場合を除いて、0 以外の）数字を出力し、小数点以下には精度と同じ桁数の数字を出力する。省略時の精度は 6 となる。精度に 0 が指定され、かつ **#** フラグが指定されていない場合、小数点文字を出力しない。この変換は、適切な桁数への値の丸めも行う。**E** 変換指定子の場合、指数に先行する文字は、**e** ではなく **E** となる。

指数は、常に 2 桁以上となる。値が 0 の場合、指数の値は 0 となる。

g, **G**　**double** 型の実引数を、有効桁数を指定する精度にしたがって、**f** 形式、あるいは **e** 形式（**G** 変換指定子の場合は **E** 形式）のいずれかに変換する。

精度が 0 の場合は、1 と解釈される。

使用される形式は、変換される値に依存する。変換の結果から得られる指数が -4 より小さいか、精度以上であれば、**e** 形式（または **E** 形式）を用いる。いずれの形式でも、後続する 0 を結果の小数部から取り除く。小数点以下に数字が続くときだけ、小数点文字を出力する。

c　**int** 型の実引数を **unsigned char** 型に型変換し、その結果の文字を出力する。

s　実引数は、文字型の配列へのポインタでなければならない。配列内の文字を、終端ナル文字の直前まで出力する。精度が指定された場合、精度を超える文字数は出力しない。精度が指定されないか、精度が配列の大きさよりも大きい場合、配列はナル文字を含まなければならない。

p　実引数は、**void** へのポインタでなければならない。そのポインタの値を処理系定義の方法で、表示可能な文字の並びに変換する。

n　実引数は、整数へのポインタでなければならない。この ***printf*** の呼出しでその時点までに出力ストリームに出力された文字数を、その整数に格納する。実引数の変換は行わない。

%　**%** を出力する。実引数はない。変換指定全体は、**%%** でなければならない。

無効な変換指定子に対する動作は、未定義とする。

実引数が、共用体もしくは集成体であるか、またはそれらを指すポインタである場合（**%s** 変換のときの文字型配列または **%p** 変換のときのポインタを除く）、動作は未定義とする。

フィールド幅が存在しないとき、または小さいときでも、変換結果を切り捨てることはない。すなわち、変換結果の文字数がフィールド幅より大きい場合、その変換結果を含む幅までフィールドを拡張する。

■ 返却値

printf 関数は、出力された文字数を返す。出力エラーが発生したときは、負の値を返す。

scanf 関数

■ 形式

```
#include  <stdio.h>
int scanf(const char *format, ...);
```

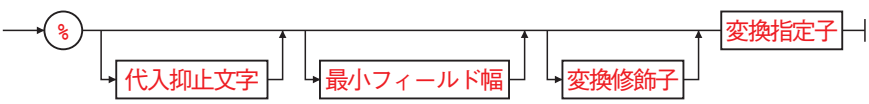
■ 機能

scanf 関数は、標準入力ストリームからの入力を変換し、*format* に続く実引数が指すオブジェクトに格納する。*format* が指す文字列は、入力として許される文字列とそれらの代入時の変換方法とを指定する書式とする。書式文字列内には、指令が含まれていなくともよいし、複数含まれていてもよい。

▶　書式に対して実引数が不足しているときの動作は、未定義とする。実引数が残っているにもかかわらず書式がつきてしまう場合、余分な実引数は評価するだけで無視する。

指令は、以下のいずれかである。

- 1 個以上の空白類文字。
- （% でも空白類文字でもない）文字。
- 変換指定（その構文は下図）。



代入抑止文字

アステリスク *****。省略可能。

最大フィールド幅

0 以外の 10 進整数。省略可能。

変換修飾子

変換結果を格納するオブジェクトの大きさを示す、**h**, **l**, **L** のいずれかの文字。

h	• **d**, **i**, **n** 変換 … 実引数が、**int** 型へのポインタではなく **short** 型へのポインタであることを指定する。 • **o**, **u**, **x** 変換 … 実引数が、**unsigned** 型へのポインタではなく **unsigned short** 型へのポインタであることを指定する。

l	• d, i, n 変換 … 実引数が、int 型へのポインタではなく long 型へのポインタであることを指定する。 • o, u, x 変換 … 実引数が、unsigned 型へのポインタではなく unsigned long 型へのポインタであることを指定する。 • e, f, g 変換 … 実引数が、float 型へのポインタではなく double 型へのポインタであることを指定する。
L	• e, f, g 変換 … 実引数が、float 型へのポインタではなく long double 型へのポインタであることを指定する。

変換修飾子を、上記以外の変換指定子とともに指定した場合の動作は未定義とする。

scanf 関数は、書式内の各指令を順に実行する。指令の実行が失敗すると、*scanf* 関数は呼び出した関数へと戻る。この失敗には、以下の二つがある。

(a) **入力誤り** … 入力文字が得られないことに起因する。

(b) **照合誤り** … 不適切な入力に起因する。

空白類文字で構成される指令は、最初の非空白類文字の直前まで（この文字は読まずに残す）、またはそれ以上読み取ることができなくなるまで、入力読取りを実行する。通常の指令は、ストリームでの次の文字の読取りを実行する。入力した文字が指令を構成する文字と異なる場合、その指令は失敗し、その入力文字およびそれ以降の文字は読まれないままストリーム上に残る。

変換指定の指令は照合入力列の集合を、各指定子に対する規定に基づき定義する。変換指定の実行は、次の手順にしたがう。

変換指定が [, c, n 指定子を含まない場合、空白類文字列を読み飛ばす。変換指定が n 指定子を含まなければ、ストリームから入力項目を読み取る。入力項目を、入力文字列中の最長の照合列として定義する。ただし、最長の照合列の長さが指定されたフィールド幅を超える場合の入力項目は、照合列の先頭からのフィールド幅の長さの部分とする。入力項目に続く文字が存在する場合、その先頭の文字は読まれずに残る。入力項目の長さがゼロのとき、指令の実行は失敗となる。この状態は、照合誤りとする。ただし、何らかのエラーのために、ストリームからの入力ができないときの失敗は、入力誤りとする。

% 指定子の場合を除いて、変換指定は入力項目を（または、%n 指定のときには入力文字数を）、変換指定子に対応した適切な型に変換する。入力項目が照合列でないとき、指令の実行は失敗となる。この状態は、照合誤りとする。代入抑止文字 * が指定されていなければ、実引数 *format* に続く実引数のうち、変換結果をまだ受け取っていない先頭のものが指すオブジェクトに、変換結果を代入する。このオブジェクトが適切な型をもたない場合、または変換結果が与えられた記憶域内で表現できなかった場合、動作は未定義とする。

変換指定子

d, i, o, u, x, X, e, E, f, g, G, s, [, c, p, n, % の各文字。

d　符号が省略可能な 10 進整数。実引数は、整数へのポインタでなければならない。

i　符号が省略可能な整数。実引数は、整数へのポインタでなければならない。

o　符号が省略可能な 8 進整数。実引数は、符号無し整数へのポインタでなければならない。

u　符号が省略可能な 10 進整数。実引数は、符号無し整数へのポインタでなければならない。

x, X　符号が省略可能な 16 進整数。実引数は、符号無し整数へのポインタでなければならない。

e, E, f, g, G　符号が省略可能な浮動小数点数。実引数は、浮動小数点数へのポインタでなければならない。

s　非空白類文字の並び。実引数は、すべての文字の並びに加えてナル文字を受け取れる大きさをもつ配列の先頭文字へのポインタでなければならない。この変換は、文字列の終わりを示すナル文字を自動的に付加する。

[　期待される**走査文字集合**（*scanset*）の要素の空でない並び。実引数は、すべての文字の並びに加えて、ナル文字を受け取れる大きさをもつ配列の先頭文字へのポインタでなければならない。この変換は、文字列の終わりを示すナル文字を自動的に付加する。

変換指定子は、この左角括弧の対になる右角括弧] とそこまでの書式文字列中のすべての文字の並びとする。左角括弧の直後がアクサンシルコンフレックス ^ でないときに、左右の角括弧間の**走査文字の並び**（*scanlist*）が走査文字集合を構成する。左角括弧の直後が ^ であるときの走査文字集合は、その ^ と右角括弧の間にある走査文字の並びに現れないすべての文字とする。変換指定子が [ ] または [^] で始まっているときには、右角括弧を走査文字の並びに含め、その次に現れる右角括弧を変換指定の終了とする。変換指定子の開始が [ ] でも [^] でもないときの変換指定の終わりの文字は、最初の右角括弧とする。ハイフン - が走査文字の並びに含まれ、かつ走査文字の先頭の文字（先頭が ^ のときは 2 文字目）でも最後の文字でもない場合の動作は、処理系定義とする。

c　フィールド幅（指令中にフィールド幅がないときは 1 とする）で指定された長さの文字の並び。この指定子に対応する実引数は、文字の並びを受け取れる大きさをもつ配列の先頭文字へのポインタでなければならない。この変換は、ナル文字を付加しない。

p　処理系定義の文字の並びの集合。この集合は、***printf*** 関数での **%p** 変換が生成する文字の並びの集合と同じとする。この指定子に対応する実引数は、**void** へのポインタへのポインタでなければならない。入力項目の解釈は、処理系定義とする。入力項目が、同一プログラム内で以前に変換した値である場合、結果のポインタ値は、その変換前の値と等しくなる。これ以外の場合、**%p** 変換の動作は未定義とする。

n　入力を読み取らない。この指定子に対応する実引数は、整数へのポインタでなければならない。***scanf*** 関数の呼出しでこれまでに入力ストリームから読み取った文字数を、その整数に書き込む。**%n** 指令を実行しても、***scanf*** 関数が終了時に返す入力項目の個数は増加しない。

%　一つの % を照合する。変換も代入も起こらない。変換指定全体は、**%%** でなければならない。

変換指定が無効であるときの動作は、未定義とする。

入力中にファイルの終わりを検出すると、変換は終了する。現在の指令にマッチする文字を 1 文字も（先行して読み飛ばされる空白類文字が許される場合、それらを除く）読み取らないままファイルの終わりを検出すると、その指令の実行は入力誤りで終了する。現在の指令に照合する文字を 1 文字でも読み取った後でファイルの終わりを検出すると、その指令が照合誤りとならない限り、それに続く指令（もし存在すれば）の実行が入力誤りによる終了となる。

指令と矛盾する入力文字によって変換が終了した場合、その矛盾した入力文字は読まれないまま入力ストリーム上に残る。入力中で後続する（改行文字を含む）空白類は、指令に照合しない限り読み込まれないまま残る。通常の文字指令および代入抑止を含む変換指定の成功を、**%n** 指令以外で直接判定することはできない。

■ 返却値

変換が一つも行われないまま入力誤りが発生すると、***scanf*** 関数はマクロ EOF の値を返す。それ以外の場合、***scanf*** 関数は代入された入力項目の個数を返す。この個数は、入力中に照合誤りが発生すると、変換指定子に対応する実引数の個数よりも少なくなることもあり、0 になることもある。