

# C言語プログラミング 基本例題88+88

富永和人【編著】

生野壮一郎・菊池眞之【共著】  
黒川弘章・関口暁宣

コロナ社

# まえがき

本書はC言語プログラミングの基本例題集です。C言語の基本的な機能について、88問の例題と解答例を示し、解答例として与えたプログラムを解説しました。例題とプログラムの実例を通してC言語を自学したいとお考えの方のため、またC言語プログラミングの入門から中級レベルの講座での利用に向けて、C言語の基本機能を網羅した分かりやすい例題集をと考えて作成しました。各例題の解説の後には、扱ったトピックに関する注意点やテクニックを「ポイント」としてまとめて示し、さらに「発展」として、少し進んだ問題や、関連する問題を88問設けています。発展問題には解答例を示していないので、自学の読者には力試しとして、また講座で使われる先生には出題の参考として利用いただけると幸いです。

章の並びは、C言語の機能に対する理解を基本から積み上げられるような順序としてあります。このため、1章から順に例題を解いていくのが最も効率的でしょう。他のテキストなどと並行して利用される場合のために、各章で扱っているトピック間の順序関係を下図に示します。矢印に沿った順ならば無理なく例題に取り組めると幸いです。なお個々の例題の中には、前にある他章のトピックにあまり依存していないものもありますので(特に8章の外部変数と静的変数についての例題や、11章のアルゴリズムに関する例題など)、早めに挑戦していただくことも可能です。

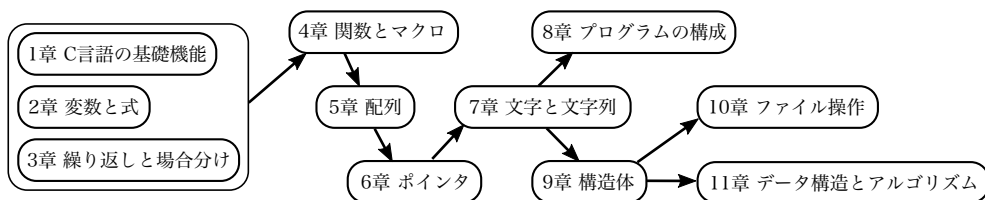


図 各章で扱っているトピック間の順序関係

本書の例題および内容は、東京工科大学における20年以上のC言語教育の経験に基づくものです。大学関係者の皆様、また授業について有益なフィードバックを下さった学生、卒業生、教育助手の皆様に深く感謝します。今も実用的な言語であるC言語を学ぶ方々の知識と技術の向上に、本書が役立つことを切に願います。

2017年1月

著者一同

## 本書の使い方

本書の各章は基本的に、C言語が持つ1つの大きな機能を扱います。本書はコンパクトな例題集とするために、読者が各機能について講座や本書以外のテキストなどである程度学んだ上で、本書の該当する章の例題に取り組みられることを想定しています。章のはじめには各機能の概略をまとめてあります。例題に入る前の知識の確認に役立ててください。

それぞれの章の各節は、**例題**、**考え方**、**解答例**、**解説**、**ポイント**、**発展**という構成になっており、必要に応じて**実行例**も示しました。自学される読者は、はじめに**例題**を見て解き方を考え、もしも難しければ**考え方**を参考にして、プログラムを自分で作ってください。その後で**解答例**を見つ**解説**を読んで、どのような実装が可能なのか学んでください<sup>†</sup>。解答例として与えたプログラムには説明のために注釈（コメント）を多めに与えています。自分で打ち込んで動作させる際にはもちろん注釈は適当に省いて結構です。その例題で重要な点を**ポイント**としてまとめてあるので、見て理解を確認してください。ポイントはまた、後で復習する際の参考にもなるでしょう。より実用的なプログラミングに近い問題などを**発展**に配してあります。プラスアルファの理解のために挑戦してみてください。

講座で利用される先生には、受講者向けに例題そのものを出題して解かせたり、例題に対する解答例をプログラミングの実例として受講者に学習させ、発展を参考に課題を出すという方法などでご活用いただけたらと思います。

巻末には**引用・参考文献**として、C言語そのものや、C言語を用いたプログラミングの学習に役立つと思われる書籍を紹介しました。参考にしてもらえれば幸いです。

### 記 法

本書で用いる記法は次の通りです。プログラムコードはタイプライタ体で記します（例：`int i;`）。コード中に何かを入れるべき場所は《》で示します（例：`int a[《要素数》]`）。

以下のような角のある矩形の囲みはプログラムやファイルを示します。

```
#include <stdio.h>

int main(void) {
    printf("こんにちは!\n");
    return 0;
}
```

<sup>†</sup> 解答例として示したプログラムは、すべて本書のホームページからダウンロードできます。アドレスは <http://www.coronasha.co.jp/np/isbn/9784339028737/> です（2017年1月現在）。

実行例は以下のように四隅が丸い囲みで示します。コマンドラインのプロンプトは \$ とし、ユーザの入力は下線で示します。↵ はエンターキーを押すことを表します。

```
$ ./a.out ↵
  こんにちは!
$
```

### ユーザの入力について

ユーザの入力を扱うような例題を解く場合には、特に指定がない限り、その入力に妥当な仮定を置いて、想定内の入力だけがくるとして構いません。例えば氏名をキーボードから入力するような例題に対してプログラムを作るときには、その上限を適当に数十文字として、それより長い氏名をユーザは打ち込まないと仮定して結構です。解答例として与えたプログラムもそのようにしてあります。なお、実用的なプログラムを作る際には一般に、ユーザのどのような入力も適切に扱う必要があるので注意して下さい。

### 文字について

C 言語で文字を扱う基本機能は 1 バイト文字 (いわゆる半角英数記号) を対象としており、ひらがなや漢字などの (いわゆる全角) 文字の取り扱いの基本の範囲を超えます。しかしながら、設問や解答例にある文などがローマ字では分かりにくいので、本書のプログラムでは出力する文字列と注釈に限ってかな漢字を使っています。もしもかな漢字が原因で、使っている環境でプログラムが正しくコンパイルできなかつたり、出力が正しく表示されないようであれば、それらを 1 バイト文字で置き換えて、`printf("こんにちは!\n")` を `printf("Hello!\n")` とするなどして下さい。ただし、キーボードや外部ファイルからの入力はすべて 1 バイト文字を想定しています。特に断りのない限り、「文字」は 1 バイト文字、「文字列」は 1 バイト文字からなる文字列と考えて下さい。

扱う 1 バイト文字の文字コードとしては、ASCII という規格から派生したものを仮定します。現在パソコンなどで広く用いられている文字コードです。ただ、そのような文字コードの間にわずかに違いがある場合があります。本書のプログラムで使っているバックスラッシュ (\) という文字が入力できない場合には、円記号 (¥) を代わりに使って下さい。同じくチルダ (~) が入力できなければ代わりにオーバーライン (-) を使って下さい。

### 処理系と環境

C 言語プログラムのコンパイルと実行を行うためのソフトウェアを C 言語の**処理系**といい、処理系や作成したプログラムを動かす場となるシステムを**環境**といいます。読者が

パソコンをお使いであれば、環境とはパソコンと OS のこと<sup>†1</sup>、処理系とはコンパイラやライブラリのことだと思っていただけであればおよそ間違いはないでしょう。

本書の例題では、コマンドラインインタフェース (CLI) で実行するようなプログラムを作ります。CLI での入出力は最も簡単で、学習に適しているためです。Linux や macOS などの UNIX 系の環境ならターミナルで、Windows 環境ならコマンドプロンプトでプログラムを実行します。実行例としては UNIX 系環境での様子を示しました。プログラムの実行方法、実行中断の方法、キーボード入力の終了を知らせる方法などは環境によって違うので、UNIX 以外の環境を利用する場合には使うシステムの説明を参照して下さい。

C 言語のプログラムを動かすには、まずプログラム (ソースファイル) を作成し、次にそれをコンパイルして実行可能プログラムに翻訳し、実行します。テキストエディタでソースファイルを作成し、CLI でコマンド (cc や gcc や cl など、処理系による) を入力してコンパイルして実行するのが最も容易でしょう。コマンドが cc でソースファイルが hello.c の場合、UNIX 系環境では以下のようにコンパイルします。

```
$ cc hello.c
```

統合開発環境 (IDE) を利用して作成とコンパイル (ビルド) を行ってから、CLI で実行するというのも可能でしょう。IDE に CLI のエミュレーション機能があれば、IDE 内で実行もできます。簡単なプログラムの作成から実行までをウェブ上で行うサービスもあります。ただし IDE やウェブの場合、本書の一部の例題で用いている実行コマンドへの引数の指定や入力の切り換えなどに制約がある場合もあります。

## C 言語の規格

C 言語にはいくつかの規格があります。本書で示すプログラムの例は原則として、2011 年に作られた C11 と呼ばれる規格に定められたうちの基本的な機能を使って作られています<sup>†2</sup>。そのため現在利用されている多くの処理系でコンパイルでき、動作するでしょう。

規格に従うと煩雑になるような用語については、より分かりやすい一般的に受け入れられていると思われる語を使いました。例えば、外部定義されたオブジェクトは「外部変数」、static を使って宣言されたオブジェクトは「静的変数」などとしました。

<sup>†1</sup> 統合開発環境とは別のものなので注意して下さい。統合開発環境はむしろ処理系にあたります。

<sup>†2</sup> ただし、プログラムの動作を面白くするために、指定された秒数だけ停止する sleep という UNIX 系の処理系の機能を使う部分が 2 箇所ほどあります (4.5 節, 8.2 節の発展)。これについてはその箇所でも明示しており、この機能を使わなくても問題なく学習できます。

# 目 次

## 1. C 言語の基礎機能

1.1	文字列の出力 — メッセージを表示する	1
1.2	数値の出力 — 数値データを表示する	2
1.3	数値の入力 — 10 進数を 16 進数に変換する	4
1.4	場合分けの基本 (1) — BMI で体型を判定する	5
1.5	場合分けの基本 (2) — 整数を分類する	7
1.6	繰り返しの基本 — 温度計の目盛りを表示する	8
1.7	ライブラリ関数を使う — 擬似乱数	10

## 2. 変数と式

2.1	型と変数 — 領域の大きさを表示する	12
2.2	数値の計算 — 商品ポイント計算	14
2.3	式の評価 — 球の体積を求める	15
2.4	演算子の優先順位 — 文字を暗号化する	17
2.5	演算子の結合性 — 人口密度を求める	19
2.6	真偽値 — 比較演算や論理演算の値	21
2.7	キャストによる型変換 — 小数点以下第 3 位を切り捨てる	23
2.8	副作用のある演算子 — 順列の総数を求める	24
2.9	代入式の値 — くじ引きプログラム	26
2.10	オペランドの評価順序 — お菓子を配る	27
2.11	型が表現できる範囲 — オーバフローとアンダフロー	29
2.12	浮動小数点数の誤差 — $1/N$ を $N$ 回足す	31

## 3. 繰り返しと場合分け

3.1	回数の決まっている繰り返し — 平均点を求める	33
3.2	繰り返しと場合分けの組合せ — 成績評価点平均を求める	34

3.3	switch 文 — 電卓プログラム	37
3.4	乱数を使う — じゃんけんプログラム	38
3.5	二重ループ (1) — 九九の表を表示する	41
3.6	二重ループ (2) — 完全数を求める	43
3.7	無限ループ — あいこなら続けるじゃんけんプログラム	45
3.8	繰り返しの使い分け — 賭けじゃんけんプログラム	47

## 4. 関数とマクロ

4.1	関数の引数 — お茶をどうぞ	50
4.2	関数の戻り値 — 指定桁で四捨五入する	53
4.3	引数のない関数 — サイコロゲーム	55
4.4	関数から関数を呼び出す — 複利計算プログラム	57
4.5	処理を関数にまとめる — 数遊び Fizz Buzz	59
4.6	オブジェクト形式マクロ — 単位換算プログラム	61
4.7	関数形式マクロ — 変数の値を入れ替える	63

## 5. 配 列

5.1	配列の基本 — 山を登って下りる	66
5.2	配列の初期化 — 月の末日の一覧を表示する	68
5.3	配列の走査 — 重いりんごを選ぶ	70
5.4	ソート — りんごを軽い順に並べる	72
5.5	配列を使ったアルゴリズム — エラトステネスのふるい	74
5.6	2次元配列 (1) — 乱数表を作る	75
5.7	2次元配列 (2) — 魔方陣を作る	77

## 6. ポ イ ン タ

6.1	ポインタの基本 — ポインタの値と基本的な演算	80
6.2	ポインタを関数に渡す — 分数を約分する	82
6.3	ポインタと配列 (1) — ポインタと配列の関係	84

6.4	ポインタと配列 (2) — 順位の前後を表示する	86
6.5	配列を関数に渡す — 配列の内容を逆順にする	88
6.6	配列とポインタの応用 — 1次元セルオートマトン	90
6.7	ポインタの配列 — 所要時間で経路をソートする	93
6.8	動的メモリ確保 — レースのタイムを格納する	96

## 7. 文字と文字列

7.1	文字の基本 — 文字と文字コード	99
7.2	1文字入力 — 大文字を数える	101
7.3	1文字ずつの入出力 — 伏せ字にする	102
7.4	行の処理 — 行頭の文字を大文字にする	104
7.5	文字を扱う型 — 文字種を関数で判定する	106
7.6	文字列の基本 — 文字列を反転する	108
7.7	文字列とポインタ — 回文かどうか判定する	110
7.8	文字列を扱う関数 — 改行文字を取り除く	112
7.9	文字コードを使った計算 — パスワード	113
7.10	文字列を返す関数 — 2進数表現を求める	116
7.11	領域を確保して返す関数 — 無制限の文字列入力	118
7.12	文字列へのポインタの配列 (1) — データを作る	120
7.13	文字列へのポインタの配列 (2) — 要素を挿入する	122
7.14	文字列へのポインタの配列 (3) — 要素を削除する	125
7.15	2次元文字配列 — データを作る	127
7.16	コマンド行引数 (1) — メッセージを繰り返し表示する	128
7.17	コマンド行引数 (2) — 引数を連結して表示する	130

## 8. プログラムの構成

8.1	外部変数 — 外貨両替プログラム	132
8.2	ブロック有効範囲の静的変数 — 貯金箱プログラム	134
8.3	ソースファイルの分割と外部参照 — 消費税計算プログラム	136
8.4	ファイル有効範囲の静的変数 — 領収書を状差しに刺す	138



8.5	ヘッダファイルを作る — 蔵書管理プログラム	140
-----	------------------------	-----

## 9. 構造体

9.1	構造体の宣言 — 従業員情報を格納する	144
9.2	構造体の初期化と代入 — プリペイドカードを発行する	146
9.3	構造体の入れ子 — 日付を構造体で表現する	148
9.4	構造体の配列 — 複数のプリペイドカードを発行する	150
9.5	構造体へのポインタ (1) — 関数に構造体のデータを渡す	152
9.6	構造体へのポインタ (2) — 関数から構造体にデータを返す	153
9.7	構造体の値と関数 — 複素数の計算	155

## 10. ファイル操作

10.1	文字単位のファイル操作 — ファイルをコピーする	158
10.2	行単位のファイル入力 — 名簿を読み込む	160
10.3	行単位のファイル出力 — 名簿をソートして書き出す	163

## 11. データ構造とアルゴリズム

11.1	線形リスト (1) — 駅一覧を作る	166
11.2	線形リスト (2) — 駅を探索して情報を表示する	169
11.3	線形リスト (3) — 駅を削除する	172
11.4	線形リスト (4) — 駅を追加する	174
11.5	数値計算 — ルートを求める	177
11.6	再帰を使う (1) — ハノイの塔を解く	179
11.7	再帰を使う (2) — フィボナッチ数を求める	182

引用・参考文献	184
---------	-----

索引	185
----	-----

# 1

## C 言語の基礎機能

C 言語はさまざまな機能を持っています。それらを使ってプログラムを作成する際に共通に必要なとなる基礎的な機能には以下のものがあります。

- 画面出力、キーボード入力
- 変数を使った計算
- プログラムの動作の制御（場合分けや繰り返し）
- 標準的なライブラリ機能の利用

本章では、これらを使った例題を示します。

### ▶ 1.1 文字列の出力 — メッセージを表示する ◀

**例題 1.1** 以下のように画面にメッセージを表示するプログラムを作成せよ。

```
$ ./a.out ↵  
Hello, world  
Bye!  
$
```

#### 考 え 方

画面に文字列を出力するには `printf` を使います。

#### 解 答 例

プログラム 1-1

```
1 #include <stdio.h>  
2  
3 int main(void) {  
4     printf("Hello, world\n");  
5     printf("Bye!\n");  
6     return 0;  
7 }
```

#### 解 説

1 行目はヘッダ `stdio.h` を取り込む指示です。このヘッダは C 言語の標準的な入出力機能を使うためのもので、このプログラムでは `printf` を使うために取り込みました。

## 2 1. C言語の基礎機能

Hello, world を表示して行を変えるために**改行文字** `\n` を出力しています (4行目)。それぞれの行を出力するため `printf` を2つ書いていますが (4~5行目)、以下のように "" の中に `\n` を2つ書けば、1つの `printf` で2行表示できます。

```
printf("Hello, world\nBye!\n");
```

6行目にある **return 文** 「`return 0;`」はプログラムの実行を終了させる文です。

### ポイント

☞ 画面出力で改行するには改行文字 `\n` を使う。

### 発展

`printf` を一度だけ使って、以下のように画面に表示するプログラムを作成せよ。出力2行目の先頭の空きは空白8個分、3行目は24個分である。

```
$ ./a.out ↵
To be, or not to be:
    that is the question.
                                -- Hamlet
$
```

## ▶ 1.2 数値の出力 — 数値データを表示する ◀

**例題 1.2** 次のような変数 (日付, 降水確率, 最高気温, 最低気温を表す) を用意し

- `int` 型の変数: `y1`, `y2`, `m1`, `m2`, `d1`, `d2`, `prec1`, `prec2`
- `double` 型の変数: `high1`, `high2`, `low1`, `low2`

以下に示す値を代入してから

```
y1 = 2020; m1 = 2; d1 = 4; prec1 = 0; high1 = 9.59; low1 = -10.1;
y2 = 2020; m2 = 10; d2 = 20; prec2 = 30; high2 = 15.3; low2 = 5.51;
```

それらの変数の値を以下のように画面に表示せよ。下の2行については、各欄の桁を示した通りに揃えること。分かりやすいように空白1つを `_` で示してある。

```
$ ./a.out ↵
最高_15.300000, 最低_-10.100000
2020/02/04_9.590000_-10.100000_0%
2020/10/20_15.300000_5.510000_30%
$
```

## 考 え 方

`printf` で変数の値などのデータを表示するには、`printf` に与える **書式** (カッコ内の最初に書く " " でくくられたもの) の中で、適切に **変換** 指定を行います。

## 解 答 例

プログラム 1-2

```

1  #include <stdio.h>
2
3  int main(void) {
4      int y1, m1, d1, prec1;
5      int y2, m2, d2, prec2;
6      double high1, low1, high2, low2;
7
8      y1 = 2020; m1 = 2; d1 = 4; prec1 = 0; high1 = 9.59; low1 = -10.1;
9      y2 = 2020; m2 = 10; d2 = 20; prec2 = 30; high2 = 15.3; low2 = 5.51;
10
11     printf("最高 %f, 最低 %f\n", high2, low1);
12     printf("%4d/%02d/%02d %10f %10f %2d%\n", y1, m1, d1, high1, low1, prec1);
13     printf("%4d/%02d/%02d %10f %10f %2d%\n", y2, m2, d2, high2, low2, prec2);
14     return 0;
15 }
```

## 解 説

データを文字列として出力するための変換を `printf` に指定します。int 型データの出力には **d 変換** を用います (12~13 行目)。表示領域の最小幅を %《幅》d のように指定すると、データはその領域に右詰めで表示されます。データの桁数が少なくて表示領域に満たない場合にできる左側 (上位桁) の空きをゼロ詰めるには %0《幅》d と指定します。double 型データは **f 変換** で出力します (11 行目)。表示領域の最小幅と小数点以下の桁数 (**精度**) を指定するには %《幅》.《精度》f とします (12~13 行目)。幅は小数点を含む全体の文字数で、精度の指定は省略すると 6 桁になります。ゼロ詰めの指定は %d と同様です。

% は変換指定の開始を意味する特殊な文字なので、% という文字そのものを表示するには %% と書きます (12~13 行目)。

## ポ イ ント

☞ `printf` でデータを出力するには適切な変換を用いる。

## 発 展

例題 1.2 のプログラムを改良し、データを以下のような表の形で出力するようにせよ。

```

$ ./a.out ↵
----Date-----H-----L-----P
+-----+-----+-----+-----+
```

```
|2020/02/04|_9.6|-10.1|_0%|
+-----+
|2020/10/20|_15.3|_5.5|30%|
+-----+
$
```

### ▶ 1.3 数値の入力 — 10 進数を 16 進数に変換する ◀

**例題 1.3** 以下のように、入力した整数を 16 進数で表示するプログラムを作成せよ。

```
$ ./a.out ↵
decimal number? 255 ↵
hexadecimal = ff
$
```

#### 考 え 方

`scanf` で入力を読み取ります。16 進表示には `printf` の **x 変換** (`%x`) を使いましょう。

#### 解 答 例

##### プログラム 1-3

```
1 #include <stdio.h>
2
3 int main(void) {
4     int n;
5
6     printf("decimal number? ");
7     scanf("%d", &n);           // 10進で入力
8     printf("hexadecimal = %x\n", n); // 16進で出力
9     return 0;
10 }
```

#### 解 説

`int` 型の変数 `n` を用意して (4 行目), `scanf` でキーボードからの入力を読み取ります (7 行目)。画面に何も表示せずに入力待ちになると何が起きているかユーザに分からないので、入力を促すメッセージ (**プロンプト**) を `printf` で表示します (6 行目)。

入力された文字列から変数にデータを得るために変換を指定します。10 進符号付き整数を表す文字列から `int` 型の数値を得るには `d` 変換を用います。書式に `%d` を指定して、コンマの後に `&(int 型の変数)` と書きます (7 行目)。これによって変数 `n` に数値が得られます。その数値を 16 進数として表示するために、`printf` の `x` 変換を使用しました (8 行

目)。printf や scanf には他にも 8 進数を扱う **o 変換** など、さまざまな変換があります。

// から行末までは**注釈**で、プログラムの動作には関係ありません。この形式の注釈に対応していない処理系の場合には、`/* … */` という注釈の形式を使って下さい。

## ポイント

☞ scanf で変数に数値を読み取るときには、変数名に `&` をつける。

☞ printf や scanf にはさまざまな変換があるので適切に使い分けよう。

## 発展

入力された 2 つの 16 進数の積を求めて結果を 16 進数で表示するプログラムを作成せよ。

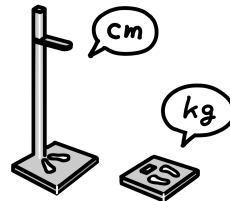
```
$ ./a.out ↵
16進数を1つ入力して下さい: 100 ↵
16進数をもう1つ入力して下さい: dead ↵
16進で 100 * dead = dead00 です
$
```

## ▶ 1.4 場合分けの基本 (1) — BMI で体型を判定する ◀

**例題 1.4** 身長と体重を入力すると、BMI と、肥満かどうかを表示するプログラムを作成せよ。BMI は次の式で求め、25 以上なら肥満、25 未満なら肥満でないとする。

$$\text{BMI} = \text{体重 [kg]} / (\text{身長 [m]} \times \text{身長 [m]})$$

```
$ ./a.out ↵
身長[cm]? 153 ↵
体重[kg]? 51 ↵
BMIは21.8, 肥満ではありません
$ ./a.out ↵
身長[cm]? 165 ↵
体重[kg]? 69 ↵
BMIは25.3, 肥満です
$
```



## 考え方

BMI は実数値なので `double` 型で扱きましょう。合わせて身長と体重も `double` 型とします。肥満かどうかは、BMI の値を使って **if 文** で場合分けをして表示します。

# 索引

<b>【あ】</b>		外部変数の——	137	標準出力	103, 158
アドレス演算	80	構造体の——	147	標準入力	100, 158
アルゴリズム	166	静的変数の——	135	ファイル有効範囲	138, 143
アンダフロー	30	配列の——	68	フィボナッチ数	182
入れ子	33	書式	3	副作用	12, 25
インクリメント	12	処理系	iii	符号付き	12, 140
インタフェース	89	水平タブ	107	符号なし	12, 140
エラトステネスのふるい	74	数値計算	166	浮動小数点型	12
演算子	12	スタック	138	部分式	16
オーバフロー	30	ストリーム	158	プロセッサ時間	183
オープン	158	制御構造	33	ブロック	64
オブジェクト形式マクロ	50, 62	整数型	12	ブロック有効範囲	135
オペランド	12	生存期間	132	プロンプト	4
<b>【か】</b>		静的変数	132, 134, 138	ヘッダ	11
改行文字	2, 104	精度	3	変換	
外部変数	132, 137	線形リスト	166, 167	型——	14, 23
型	12	選択ソート	73	printf などの——	3
仮引数	50	添字	66	変数	12
環境	iii	ソースファイル	132	ポインタ	80
関数	50, 132	ソート	72	本体	
関数形式マクロ	50, 64	素数	44, 74	関数の——	50
関数プロトタイプ	50, 52, 132, 137	<b>【た】</b>		マクロの——	50
間接演算	80	代入	12, 27	<b>【ま】</b>	
擬似乱数	10	タグ	144	マクロ	50
キャスト	23, 140	種 (擬似乱数の)	11	魔方陣	77
空白類文字	107	注釈	5	無限ループ	46
空ポインタ	80	定義宣言 (外部変数の)	137	メンバ	144
空文字	99, 109	データ構造	166	文字	99, 100
繰り返し	33	テキストストリーム	158	文字コード	18, 99, 100
クローズ	158, 159	デクリメント	12	文字列へのポインタ	121
結合	143	動的メモリ確保	80, 119	——の配列	99, 121
結合性	12, 20	<b>【な】</b>		文字列リテラル	109
構造体	144	二重ループ	33, 77	戻り値	50
誤差	12, 32	ニュートン法	177	モンテカルロ法	40
コマンド行引数	99, 128	ノード	166, 167	<b>【や】</b>	
<b>【さ】</b>		<b>【は】</b>		ユークリッドの互除法	83
再帰呼び出し	166, 180	場合分け	33	有効範囲	132
参照宣言 (外部変数の)	137	バイナリストリーム	158	優先順位	12, 18
式	12	配列	66	要素	66
——の値	12, 25	ハノイの塔	179	呼び出し	
実引数	50	バブルソート	95, 165	関数の——	50
自動変数	132	評価	12, 16	マクロの——	50
初期化		評価順序	28	<b>【ら】</b>	
		標準エラー出力	158	ライブニッツの公式	179

リンク	143, 166, 167	列挙型	71
ループ	33	列挙定数	71

【A, B】	【L, M】	【V, W, X】	
atof 関数	136	limits.h	29
atoi 関数	129	malloc 関数	97, 119
break 文	38	M_PI	52
		void へのポインタ型	80
		while 文	9
		x 変換	4
【C】	【N, O】	【数字】	
c 変換	18	NULL	80
case ラベル	38	o 変換	5
clock 関数	183		
CLOCKS_PER_SEC	183	【P】	
clock_t 型	183	p 変換	80
		perror 関数	98
		pow 関数	58
		printf 関数	1, 3
		ptrdiff_t 型	95
		putchar 関数	103
【D】	【R】	【記号】	
d 変換	3	! 演算子	22
default ラベル	38	!= 演算子	22
do-while 文	46	#include 指令	11, 141
		%c	18
		%d	3, 4
		%e	30
		%f	3
		%lf	6
		%p	80
		%x	4
		%zu	12, 84
		%%	3
		& 演算子 (単項)	4, 80, 81
		& 演算子 (2 項)	117
		&& 演算子	22
		* 演算子 (単項)	80, 81
		++ 演算子	25
		-- 演算子	25
		-> 演算子	144, 153
		. 演算子	144, 145
		/* */	5
		//	5
		< 演算子	22
		= 演算子	27
		== 演算子	22
		[] 演算子	66
		\n	2, 104, 107
		\t	107
		\0	99, 109
		^D	102
		^Z	102
		演算子	22
【E】	【S】		
e 変換	30	s 変換	110
EOF	99, 101	scanf 関数	4
exit 関数	97, 98	sin 関数	52
EXIT_FAILURE	38, 97	sizeof 演算子	12, 84
extern	132, 137	size_t 型	12, 97
		sleep 関数	60, 135
		sprintf 関数	118
		sqrt 関数	71, 75, 178
		srand 関数	11
		static	134, 143
		stdin	100, 160
		stdio.h	1
		stdout	103, 160
		strchr 関数	154
		strcmp 関数	141
		strcpy 関数	145
		strncpy 関数	162
		switch 文	37
【F】	【T】		
f 変換	3, 6	time 関数	39
fabs 関数	178	toupper 関数	104
fclose 関数	159		
fgetc 関数	158		
fgets 関数	110		
FILE * 型	158		
fopen 関数	158		
for 文	33		
fprintf 関数	164		
fputc 関数	159		
fputs 関数	165		
free 関数	98, 168		
【G, I】			
getchar 関数	99		
if 文	5		
INT_MAX	29		
isdigit 関数	103		
islower 関数	107		
isspace 関数	107		
isupper 関数	102		



—— 編著者・著者略歴 ——

**冨永 和人**（とみなが かずと）

1989年 東京工業大学工学部情報工学科卒業  
 1991年 東京工業大学大学院博士前期課程修了(情報工学専攻)  
 1994年 東京工業大学大学院博士後期課程単位取得退学(情報工学専攻)  
 1994年 東京工科大学講師  
 1996年 博士(工学)(東京工業大学)  
 2002年 東京工科大学助教授  
 2003年 イリノイ大学(米国)客員研究員  
 ~04年  
 2007年 東京工科大学准教授  
 2012年 和情報網設立(代表)  
 現在に至る

**菊池 眞之**（さくち まさゆき）

1994年 早稲田大学理工学部電気工学科卒業  
 1996年 大阪大学大学院基礎工学研究科博士前期課程修了(物理系専攻)  
 1996年 大阪大学大学院基礎工学研究科博士後期課程中退(物理系専攻)  
 1996年 大阪大学助手  
 1997年 大阪大学大学院助手  
 1999年 博士(工学)(大阪大学)  
 1999年 筑波大学助手  
 2003年 東京工科大学講師  
 現在に至る

**関口 暁宣**（せきぐち あきのり）

1996年 東京大学工学部機械情報工学科卒業  
 1998年 東京大学大学院工学系研究科修士課程修了(機械情報工学専攻)  
 2002年 東京大学大学院工学系研究科博士課程修了(機械情報工学専攻)  
 博士(工学)  
 2002年 弘前大学助手  
 2007年 弘前大学大学院助教  
 2008年 東京工科大学講師  
 現在に至る

**生野壮一郎**（いくの そういちろう）

1994年 山形大学工学部電子情報工学科卒業  
 1996年 山形大学大学院工学研究科博士前期課程修了(電子情報工学専攻)  
 1999年 筑波大学大学院工学研究科博士後期課程修了(電子情報工学専攻)  
 博士(工学)  
 1999年 東京工科大学講師  
 2006年 東京工科大学助教授  
 2007年 東京工科大学准教授  
 2016年 東京工科大学教授  
 現在に至る

**黒川 弘章**（くろかわ ひろあき）

1993年 慶應義塾大学理工学部電気工学科卒業  
 1995年 慶應義塾大学大学院理工学研究科修士課程修了(電気工学専攻)  
 1997年 慶應義塾大学大学院理工学研究科博士課程修了(電気工学専攻)  
 博士(工学)  
 1997年 東京工科大学講師  
 2009年 東京工科大学准教授  
 現在に至る

## C 言語プログラミング基本例題 88+88

C Programming : Basic Examples and Problems 88+88

© Tominaga, Ikuno, Kikuchi, Kurokawa, Sekiguchi 2017

2017年3月6日 初版第1刷発行

★

検印省略

編著者 富 永 和 人  
著者 生 野 壮 一 郎  
菊 池 眞 之  
黒 川 弘 章  
関 口 暁 宣  
発行者 株式会社 コロナ社  
代表者 牛来真也  
印刷所 三美印刷株式会社  
製本所 株式会社 グリーン

112-0011 東京都文京区千石 4-46-10

発行所 株式会社 コロナ社  
CORONA PUBLISHING CO., LTD.

Tokyo Japan

振替 00140-8-14844 · 電話 (03) 3941-3131 (代)

ホームページ <http://www.coronasha.co.jp>

ISBN 978-4-339-02873-7 C3055 Printed in Japan

(松岡)



 <出版者著作権管理機構 委託出版物>

本書の無断複製は著作権法上での例外を除き禁じられています。複製される場合は、そのつど事前に、出版者著作権管理機構（電話 03-3513-6969, FAX 03-3513-6979, e-mail: info@jcopy.or.jp）の許諾を得てください。

本書のコピー、スキャン、デジタル化等の無断複製・転載は著作権法上での例外を除き禁じられています。購入者以外の第三者による本書の電子データ化及び電子書籍化は、いかなる場合も認めていません。落丁・乱丁はお取替えいたします。