

# 明解C语言

TURING

图灵程序  
设计丛书

第3版

## 入门篇

[日] 柴田望洋 / 著 管杰 罗勇 杜晓静 / 译

**比课本更易懂** | 205段代码+220幅图表

荣获日本工学教育协会著作奖

双色印刷 → 技术书也能赏心悦目



中国工信出版集团



人民邮电出版社  
POSTS & TELECOM PRESS

# 明解C语言

TURING  
图灵程序  
设计丛书

第3版

## 入门篇

[日] 柴田望洋 / 著 管杰 罗勇 杜晓静 / 译



人民邮电出版社  
北京

## 图书在版编目(CIP)数据

明解C语言:第3版.入门篇/(日)柴田望洋著;  
管杰,罗勇,杜晓静译.--2版.--北京:人民邮电出  
版社,2015.11

(图灵程序设计丛书)

ISBN 978-7-115-40482-4

I. ①明… II. ①柴… ②管… ③罗… ④杜… III.

①C语言—程序设计—教材 IV. ①TP312

中国版本图书馆CIP数据核字(2015)第226046号

## 内 容 提 要

本书是日本的C语言经典教材,自出版以来不断重印、修订,被誉为“C语言圣经”。本书图文并茂,示例丰富,设有205段代码和220幅图表,对C语言的基础知识进行了彻底剖析,内容涉及数组、函数、指针、文件操作等。对于C语言语法以及一些难以理解的概念,均以精心绘制的示意图,清晰、通俗地进行讲解。

本书适合C语言初学者阅读。

- 
- ◆ 著 [日]柴田望洋  
译 管杰 罗勇 杜晓静  
责任编辑 乐馨  
执行编辑 杜晓静  
责任印制 杨林杰
- ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号  
邮编 100164 电子邮件 315@ptpress.com.cn  
网址 <http://www.ptpress.com.cn>  
北京天宇星印刷厂印刷
- ◆ 开本:800×1000 1/16  
印张:26  
字数:407千字 2015年11月第2版  
印数:7 001-1 2000册 2015年11月北京第1次印刷  
著作权合同登记号 图字:01-2015-3955号
- 

定价:79.00元

读者服务热线:(010)51095186转600 印装质量热线:(010)81055316

反盗版热线:(010)81055315

广告经营许可证:京崇工商广字第0021号

# 前言

---

大家好！

本书是讲解 C 语言基础知识的教材。为帮助大家理解，书中使用了大量的代码和图表。

请大家回忆一下学习英文时的情形。除了单词和语法之外，是不是还学习了很多在具体对话和文章中的应用示例呢？

学习编程语言时也有着类似的情况。首先，对关键字和库函数等语句和语法规则的学习至关重要。我们知道，仅仅了解单词和语法，并不能写出文章或者与人对话；同样，如果只有一些知识碎片，是不能编写程序的。

为了帮助大家学习真正的 C 语言程序，本书中提供了 205 段完整的代码。另外，通过 220 幅图表，对语法和难懂的概念进行了详细的讲解。

示例程序较多，就相当于外语教材中表示单词和语法的用法的对话和例句较多。请大家通过这为数众多的程序和帮助加深理解的图表，开启你的 C 语言编程之路吧！

笔者在编写本书时使用了口语化的语言。如果读者在阅读时能感觉到像是在听笔者讲课，那笔者将倍感荣幸。

2014 年 7 月  
柴田望洋

## 第1章 初识C语言

1-1	显示计算结果	2
	计算整数的和并显示结果	2
	程序和编译	2
	注释	3
	固定代码	4
	printf函数：格式化输出函数	4
	语句	5
	计算并显示整数的差	5
	格式化字符串和转换说明	6
	符号的称呼	8
	无格式化输出	8
	字符串常量	10
	转义字符	10

2-1	运算	24
	运算符和操作数	24
	乘除运算符和加减运算符	25
	除法运算的商和余数	25
	使用printf函数打印%	26
	获取整数的最后一位数字	26
	多个转换说明	27
	单目运算符	28
	赋值运算符	29
	表达式和赋值表达式	30
	表达式语句	30

1-2	变量	11
	变量和声明	11
	赋值	12
	初始化	13
	声明时初始化	14
	初始化和赋值	14
1-3	输入和显示	16
	通过键盘进行输入	16
	格式化输入函数scanf	16
	乘法运算	17
	输出函数puts	18
	总结	21

## 第2章 运算和数据类型

2-2	数据类型	31
	求平均值	31
	数据类型	32
	数据类型和对象	33
	整型常量和浮点型常量	34
	double类型的运算	34
	数据类型和运算	35
	类型转换	37
	转换说明	39
	总结	42

## 第3章 分支结构程序

3-1	if 语句	46	计算三个数的最大值	59
	if语句·其1	46	条件运算符	60
	奇数的判定	47	差值计算	61
	if语句·其2	48	复合语句(程序块)	62
	奇数·偶数的判断	49	逻辑运算符	65
	非0的判断	49	短路求值	67
	if语句的结构图	50		
	相等运算符	52	3-2 switch语句	70
	余数的判断	53	switch语句和break语句	70
	关系运算符	54	复杂的switch语句	72
	嵌套的if语句	55	switch语句和if语句	73
	判断	57	选择语句	74
	计算较大值	58	总结	75
4-1	do语句	78	限定次数的循环操作	91
	do语句	78	字符常量和putchar函数	92
	复合语句(程序块)中的声明	79	do语句和while语句	93
	读取一定范围内的值	80	前置递增运算符和	
	逻辑非运算符·德摩根定律	81	前置递减运算符	93
	德摩根定律	81	do语句的显示	95
	求多个整数的和及平均值	82	逆向显示整数	96
	复合赋值运算符	84		
	后置递增运算符和		4-3 for语句	99
	后置递减运算符	85	for语句	99
			使用for语句实现固定次数的	
4-2	while语句	87	循环	101
	while语句	87	偶数的枚举	103
	用递减运算符简化程序代码	88	约数的枚举	104
	数据递增	90	表达式语句和空语句	104

## 第4章 程序的循环控制

	循环语句	105		运算符	114
4-4	<b>多重循环</b>	107		标识符	114
	二重循环	107		分隔符	115
	用break语句强制结束循环	108		常量和字符串常量	115
	显示图形	109		自由的书写格式	116
	多重循环	111		连接相邻的字符串常量	117
4-5	<b>程序的组成元素和格式</b>	114		缩进	117
	关键字	114	<b>总结</b>		118

## 第5章 数组

5-1	<b>数组</b>	122		使用数组进行成绩处理	131
	数组	122		对象式宏	131
	数组的声明（使用数组前的准备）	123		数组元素的最大值和最小值	133
	访问数组（数组的使用方法）	123		赋值表达式的判断	134
	数组的遍历	124		数组的元素个数	135
	数组初始化	126	5-2	<b>多维数组</b>	138
	数组的复制	127		多维数组	138
	输入数组元素的值	129	<b>总结</b>		142
	对数组的元素进行倒序排列	129			

## 第6章 函数

6-1	<b>什么是函数</b>	146		三个数中的最大值	151
	main函数和库函数	146		将函数的返回值作为参数	
	什么是函数	146		传递给函数	152
	函数定义	147		调用其他函数	154
	函数调用	148		值传递	155

6-2	函数设计 .....	158
	没有返回值的函数 .....	158
	通用性 .....	159
	不含形参的函数 .....	160
	函数返回值的初始化 .....	161
	作用域 .....	161
	文件作用域 .....	162
	声明和定义 .....	163
	函数原型声明 .....	163
	头文件和文件包含指令 .....	164
	函数的通用性 .....	165
	数组的传递 .....	166

函数的传递和const类型的 修饰符 .....	168
线性查找（顺序查找） .....	170
哨兵查找法 .....	172
多维数组的传递 .....	175

6-3	作用域和存储期 .....	178
	作用域和标识符的可见性 .....	178
	存储期 .....	180

总结 .....	185
----------	-----

7-1	基本数据类型和数 .....	190
	算数类型和基本数据类型 .....	190
	基数 .....	191
	基数转换 .....	192
7-2	整型和字符型 .....	195
	整型和字符型 .....	195
	<limits.h>头文件 .....	196
	字符型 .....	199
	位和CHAR_BIT .....	200
	sizeof运算符 .....	200
	size_t型和typedef声明 .....	202
	整型的灵活运用 .....	202
	整型的内部表示 .....	204
	无符号整数的内部表示 .....	205
	有符号整数的内部表示 .....	207
	按位操作的逻辑运算 .....	209
	位移运算符 .....	212

## 第7章 基本数据类型

整型常量 .....	216
整型常量的数据类型 .....	218
整数的显示 .....	218
数据溢出和异常 .....	220

7-3	浮点型 .....	221
	浮点型 .....	221
	浮点型常量 .....	223
	<math.h>头文件 .....	224
	循环的控制 .....	225

7-4	运算和运算符 .....	228
	运算符的优先级和结合性 .....	228
	优先级 .....	228
	结合性 .....	228
	数据类型转换 .....	230

总结 .....	232
----------	-----



## 第8章 动手编写各种程序吧

8-1	函数式宏 .....	236		
	函数和数据类型 .....	236		
	函数式宏 .....	237		
	函数和函数式宏 .....	238		
	不带参数的函数式宏 .....	239		
	函数式宏和逗号运算符 .....	240		
8-2	排序 .....	243		
	冒泡排序法 .....	243		
8-3	枚举类型 .....	246		
	枚举类型 .....	246		
	枚举常量 .....	248		
				命名空间 .....
				250
8-4	递归函数 .....	251		
	函数和类型 .....	251		
	阶乘 .....	252		
8-5	输入输出和字符 .....	255		
	getchar函数和EOF .....	255		
	从输入复制到输出 .....	256		
	数字字符计数 .....	256		
	字符 .....	258		
	转义字符 .....	261		
	总结 .....	263		

## 第9章 字符串的基本知识

9-1	什么是字符串 .....	266		
	字符串字面量 .....	266		
	字符串字面量的长度 .....	266		
	字符串 .....	268		
	字符数组的初始化赋值 .....	269		
	空字符串 .....	270		
	字符串的读取 .....	270		
	格式化显示字符串 .....	271		
				读取字符串数组中的字符串 .....
				274
9-3	字符串处理 .....	275		
	字符串长度 .....	275		
	显示字符串 .....	277		
	数字字符的出现次数 .....	278		
	大小写字符转换 .....	279		
	字符串数组的参数传递 .....	280		
	总结 .....	283		
9-2	字符串数组 .....	273		
	字符串数组 .....	273		

# 第10章 指针

10-1 指针	286
函数的参数	286
对象和地址	287
取址运算符	288
指针	289
指针运算符	291
10-2 指针和函数	293
作为函数参数的指针	293
计算和与差	294
二值互换	296
将两个值排序	297

11-1 字符串和指针	316
用数组实现的字符串和用 指针实现的字符串	316
用数组实现的字符串和用 指针实现的字符串的不同点	318
字符串数组	320
11-2 通过指针操作字符串	323
判断字符串长度	323
字符串的复制	325
不正确的字符串复制	328
返回指针的函数	329

# 第12章 结构体

12-1 结构体	340
数据关联性	340

scanf函数和指针	298
指针的类型	299
空指针	300
标量型	301

10-3 指针和数组	302
指针和数组	302
指针运算符和下标运算符	304
数组和指针的不同点	306
数组的传递	308
总结	311

# 第11章 字符串和指针

11-3 字符串处理库函数	330
strlen函数：求字符串的长度	330
strcpy函数、strncpy函数： 复制字符串	331
strcat函数、strncat函数： 连接字符串	332
strcmp函数、strncmp函数： 比较字符串的大小关系	332
atoi函数、atol函数、atof函数： 转换字符串	333
总结	336

结构体	342
结构体成员和运算符	344

成员的初始化 .....	345
结构体成员和->运算符 .....	346
结构体和typedef .....	348
结构体和程序 .....	350
聚合类型 .....	351
返回结构体的函数 .....	351
命名空间 .....	353

结构体数组 .....	353
派生类型 .....	355

<b>12-2 作为成员的结构体</b> .....	356
表示坐标的结构体 .....	356
具有结构体成员的结构体 .....	357
<b>总结</b> .....	361

## 第 13 章 文件处理

<b>13-1 文件与流</b> .....	364
文件与流 .....	364
标准流 .....	365
FILE型 .....	365
打开文件 .....	365
关闭文件 .....	368
打开与关闭文件示例 .....	369
文件数据汇总 .....	370
写入日期和时间 .....	372
获取上一次运行时的信息 .....	376
显示文件内容 .....	378
文件的复制 .....	380

<b>13-2 文本和二进制</b> .....	382
在文本文件中保存实数 .....	382
文本文件和二进制文件 .....	383
在二进制文件中保存实数 .....	384
显示文件自身 .....	386
<b>13-3 printf函数与scanf函数</b> .....	389
printf函数：带格式输出 .....	389
scanf函数：带格式的输入 .....	393
<b>总结</b> .....	397

C语言的历史 .....	400
K&R——C语言的圣经 .....	400

## 附录 C语言简介

C语言标准规范 .....	400
---------------	-----

<b>结语</b> .....	402
<b>参考文献</b> .....	405

# 第1章

## 初识C语言

如果说熟悉了一件事就能大幅进步的话，那么长期从事这件事并已完全熟练的人就应该是高手了。但现实并非如此，就拿体育训练来说，假如训练的方式是错误的，只会越练习越差。编程也是如此，仅仅熟练是不够的。

不过，任何事情在开始的时候，都需要先试试水。本章就带领大家尝试一下简单的C语言编程。

## 1-1 显示计算结果

即使是计算机进行计算，计算结果如果不显示在画面上，我们也无法知晓。本节就来学习将计算结果显示在画面上的方法。

### 计算整数的和并显示结果

电脑也称为电子计算机，对它来说，任何任务都是通过计算来完成的。那么就让我们使用C语言来进行下面的计算吧。

计算整数 15 和 37 的和，并显示结果。

在编辑器中键入如代码清单 1-1 所示的程序代码。C 语言程序是区分大小写和全半角字符的，请大家在书写的时候特别注意。

代码清单 1-1
chap01/list0101.c

```

/*
  显示整数 15 和 37 的和
*/
#include <stdio.h>

int main(void)
{
    printf("%d", 15 + 37); /* 用十进制数显示整数 15 和 37 的和 */
    return 0;
}

```

注意不要和 studio 混淆

空白部分通过 Tab 键或空格键输入（注意不可用全角空格）。

**运行结果**  
52

► 程序中的空白和引号（"）等符号不可用全角输入。空白部分应通过空格键或 Tab 键输入（详见 4-5 节）。

另外，本书中的示例代码都可以从图灵社区的支持页面下载<sup>①</sup>。各代码清单右上角显示的是包括文件夹名在内的文件名。

### 程序和编译

如代码清单 1-1 所示，人们通过字符序列创建出的程序称为源程序（source program），用来保存源程序的文件称为源文件（source file）。

<sup>①</sup> 打开<http://www.it-ebooks.com.cn/book/1671>，点击“随书下载”。

► source 是“原始”的意思，因此源程序也叫作原始程序。

习惯上我们把 C 语言源文件的扩展名定为“.c”，例如我们可以把源文件命名为 list0101.c，并保存在 chap01 这个文件夹中。

★

通过字符序列创建出的程序，需要转换为计算机能够理解的位序列，也就是 0 和 1 的序列。源程序通常需要进行如图 1-1 所示的翻译操作之后才能执行（关于位的介绍请参考第 7 章）。

完成这些翻译工作之后运行程序，屏幕上就能显示出结果 52 了。

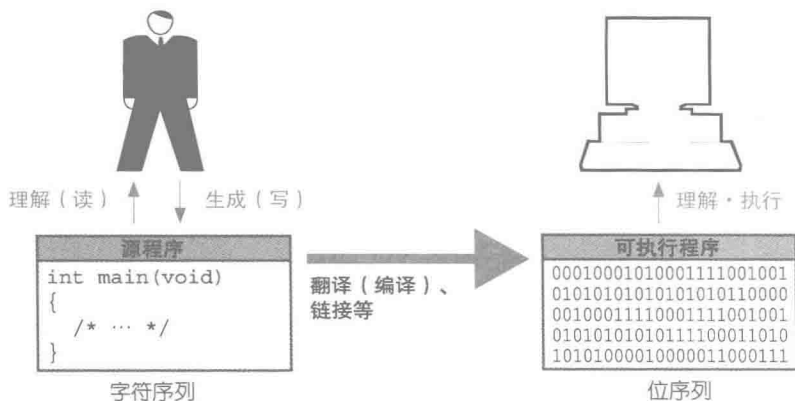


图 1-1 源程序和可执行程序

► 编译器和运行环境不同时，翻译的步骤和程序的执行方法也不同，请大家参考各自编译器的说明书。在后面的专题 1-1 中会对翻译和编译器等术语进行说明。

源程序中如果有拼写错误，翻译的时候就会发生错误，并显示出相应的**诊断消息**（diagnostic message）。出现这种情况时请仔细检查键入的程序代码，纠正错误之后再行编译。

程序中有着大量 # 和 { 等符号，大家可能不理解它们的意思。不过没关系，我们慢慢来，一点一点学习。

► 稍后我们还会对符号的称呼进行总结。

## 注释

源程序中 /\* 和 \*/ 之间的部分，称为**注释**<sup>①</sup>（comment）。有没有注释以及注释的内容如何，

① C99 支持单行注释，即“//……”这种形式，“//”之后直到行尾的内容为注释。（本书脚注均为译者注。）

其实对程序的运行并没有什么影响。编程者用简洁明了的语言将程序想要表达的意思标注在程序旁，这样能提高程序的可读性。

### ■ 注意 ■

请大家在源程序中，用简洁的语言把想要表达的意思以**注释**的形式记录下来。

从程序中可以看出，注释也可以是多行的。但是请大家注意不要把结束注释用的符号误写成 `/*`，否则后面的程序都会被解释为注释。

## 固定代码

删除程序中注释后的状态如图 1-2 所示。白底以外的部分是一段固定代码，它的含义之后会详细介绍，请大家牢记这段代码。

现阶段我们暂时先照搬这段代码，其余的部分由自己编写。

```
#include <stdio.h>

int main(void)
{
    printf("%d", 15 + 37);
    return 0;
}
```

图 1-2 程序和固定代码

► `stdio` 是 standard I/O (标准输入输出) 的缩写。请注意不要与 `studio` 混淆。

## printf 函数：格式化输出函数

`printf` 函数可以在显示器上进行输出操作（末尾的 `f` 源自 `format`（格式化）这个单词）。

如果想要使用某个函数的功能，就必须通过**函数调用**（function call）来实现。调用 `printf` 函数显示 15 和 37 的和的过程如图 1-3 所示。

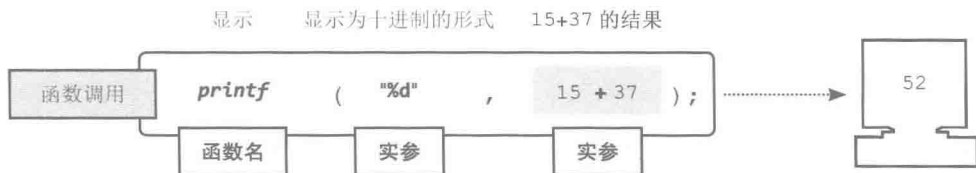


图 1-3 调用 `printf` 函数在画面上显示结果

调用此函数即发出了“显示这些内容”的请求，然后通过括号中的**实参**（argument）来传递想要显示的内容。另外，如本例所示，当实参超过两个时，需要用逗号隔开。

`printf` 函数的第一个实参 `"%d"` 指定了输出格式，它告诉程序：以十进制数的形式显示后面的实参。因此，通过调用 `printf` 函数显示出了第二个实参 `15+37` 的值，即 15 与 37 的和 52。

▶ `"%d"` 的 `d` 源自 `decimal`（十进制数）。关于十进制以外的数和显示等，我们将在第 7 章详细讲述。另外，关于 `printf` 函数的详情，请参考 13-3 节。

### ■ 注意 ■

函数调用是申请进行处理请求，而调用函数时的一些辅助指示则通过**实参**来发出。

## 语句

请大家仔细观察之前的程序代码，调用 `printf` 函数的时候使用了分号，那段固定代码（`return 0;`）中也使用了分号。这里的分号就相当于中文里的句号。

正如在句子末尾加上句号才能构成完整的一句话，C 语言中也需要在末尾加上分号来构成正确的**语句**（statement）。

### ■ 注意 ■

原则上语句必须以**分号**结尾。

开始执行程序后，固定代码中 `{` 和 `}` 之间的语句会被按顺序执行（详情请参考第 6 章）。

## 计算并显示整数的差

代码清单 1-2 所示程序的功能是计算并显示 15 减去 37 的差。

将加法运算的程序变为减法运算是很容易的。比如计算 15 减去 37 的差并显示结果的程序如代码清单 1-2 所示。

▶ 只需复制代码清单 1-1，并改变不同的地方，即可快速地生成程序。



chap01/list0102.c

## 代码清单 1-2

```
/*  
 * 显示整数 15 减去 37 的差  
 */  
  
#include <stdio.h>  
  
int main(void)  
{  
    printf("%d", 15 - 37);    /* 用十进制数显示整数 15 减去 37 的值 */  
  
    return 0;  
}
```

运行结果

-22

运行程序就会显示结果 -22。可以看到当计算结果为负数时，数字前面会自动加上负号。

## 专题 1-1 翻译阶段和编译

运行 C 语言之前，理论上要经过 8 个翻译阶段（translation phase）。另外，运行源代码还需要安装必要的软件环境，也就是编译器<sup>①</sup>。

大多数 C 语言编译器都是通过编译方式（如本文中描述的方式）把源代码翻译成计算机能够直接理解执行的形式。但是也存在逐行解释然后执行的解释方式（执行速度比较缓慢）。

## 格式化字符串和转换说明

程序运行的时候如果只显示和或者差的值，理解上会比较困难，接下来我们让结果显示得更加人性化一些，请看代码清单 1-3 所示程序。

这次我们把 *printf* 函数的第一个实参设置得更长更复杂一些。

<sup>①</sup> 即符合 C 语言规范的实现（implementation）。