C

初识 C 语言

1972

丹尼斯里奇

（1）优点：

- 设计特性：自顶向下规划、结构化编程、模块化设计
- 高效性
- 可移植性
- 强大而灵活
- 面向程序员

（2）缺点：

- 指针危险！

（3）应用范围：

- 操作系统
- 嵌入式系统

（4）语言标准

C11 标准：

（5）使用 C 语言的 7 个步骤

- 定义程序目标
- 设计程序
- 编写代码
- 编译
- 运行程序
- 测试和调试程序
- 维护和修改代码

本章小结

- C 是强大而简洁的编程语言。很好的控制硬件，容易移植。
- C 是编译型语言。C 编译器和链接器是把 C 语言源代码转换成可执行代码的程序

C 语言概述

简单的 C 程序示例

```c
#include <stdio.h>

int main(void)
{
    int num;
    num = 1;

    printf("I am a simple ");
    printf("computer.\n");
    printf("My favorite number is %d because it is first.\n",num);

    return 0;
}
```

示例解释

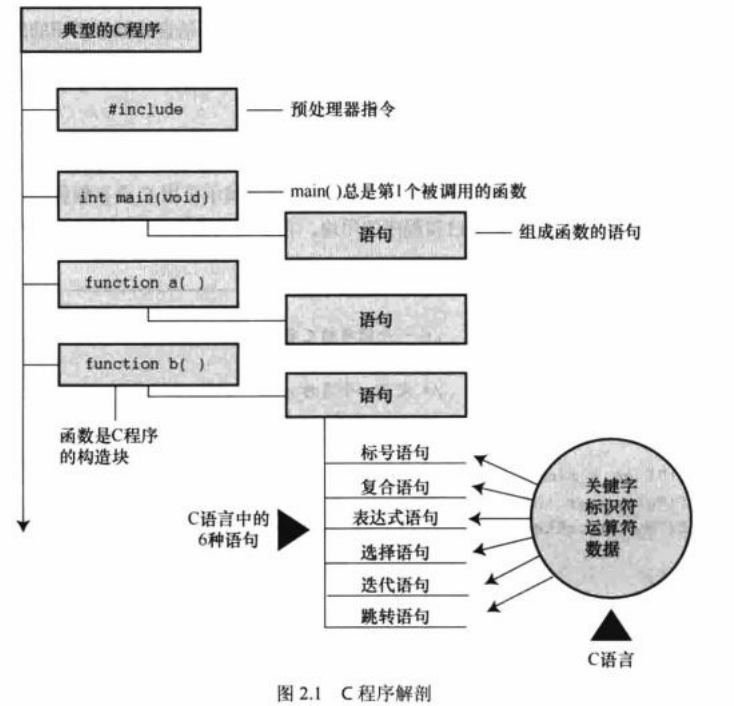

图 2.1　C 程序解剖

程序细节

#include 指令和头文件

main()函数

注释

花括号、函数体和块

声明

声明多个变量用逗号隔开

关键字

标识符（变量名、函数名等）基本要求不报错

1. 见名知意

2. 下划线命名法

3. 驼峰命名法

4. 匈牙利命名法

赋值

printf()函数

return 语句

关键字和保留标识符

表 2.2 ISO C 关键字

auto	extern	short	while
break	float	**signed**	***_Alignas***
case	for	sizeof	***_Alignof***
char	goto	static	***_Atomic***
const	if	struct	***_Bool***
continue	*inline*	switch	***_Complex***
default	int	typedef	***_Generic***
do	long	union	***_Imaginary***
double	register	unsigned	***_Noreturn***
else	restrict	void	***_Static_assert***
enum	return	**volatile**	***_Thread_local***

本章小结

C 程序由一个或多个 C 函数组成。每个 C 程序必须包含一个 `main()` 函数，这是 C 程序要调用的第 1 个函数。简单的函数由函数头和后面的一对花括号组成，花括号中是由声明、语句组成的函数体。

在 C 语言中，大部分语句都以分号结尾。声明为变量创建变量名和标识该变量中储存的数据类型。变量名是一种标识符。赋值表达式语句把值赋给变量，或者更一般地说，把值赋给存储空间。函数表达式语句用于调用指定的已命名函数。调用函数执行完毕后，程序会返回到函数调用后面的语句继续执行。

`printf()` 函数用于输出想要表达的内容和变量的值。

一门语言的**语法**是一套规则，用于管理语言中各有效语句组合在一起的方式。语句的**语义**是语句要表达的意思。编译器可以检测出语法错误，但是程序里的语义错误只有在编译完之后才能从程序的行为中表现出来。检查程序是否有语义错误要跟踪程序的状态，即程序每执行一步后所有变量的值。

最后，关键字是 C 语言的词汇。

数据和 C

数据：数据类型关键字

表 3.1 C 语言的数据类型关键字

最初 K&R 给出的关键字	C90 标准添加的关键字	C99 标准添加的关键字
int	signed	_Bool
long	void	_Complex
short		_Imaginary
unsigned		
char		
float		
double		

在 C 语言中，用 int 关键字来表示基本的整数类型。后 3 个关键字（long、short 和 unsigned）和 C90 新增的 signed 用于提供基本整数类型的变式，例如 unsigned short int 和 long long int。char 关键字用于指定字母和其他字符（如，#、$、%和*）。另外，char 类型也可以表示较小的整数。float、double 和 long double 表示带小数点的数。_Bool 类型表示布尔值（true 或 false），_complex 和 _Imaginary 分别表示复数和虚数。

通过这些关键字创建的类型，按计算机的储存方式可分为两大基本类型：整数类型和浮点数类型。

位、字节和字

1 位（开和关）

1 字节均为 8 位

1 字（8 位-16 位-32 位-64 位）

C 语言基本数据类型

转义序列

表 3.2 转义序列

转义序列	含义
\a	警报（ANSI C）
\b	退格
\f	换页
\n	换行
\r	回车
\t	水平制表符
\v	垂直制表符
\\	反斜杠（\）
\'	单引号
\"	双引号
\?	问号
\0oo	八进制值（oo 必须是有效的八进制数，即每个 o 可表示 0～7 中的一个数）
\xhh	十六进制值（hh 必须是有效的十六进制数，即每个 h 可表示 0～f 中的一个数）

基本数据类型

小结：基本数据类型

关键字：

基本数据类型由 11 个关键字组成：int、long、short、unsigned、char、float、double、signed、_Bool、_Complex 和_Imaginary。

有符号整型：

有符号整型可用于表示正整数和负整数。

- int——系统给定的基本整数类型。C 语言规定 int 类型不小于 16 位。

- short 或 short int——最大的 short 类型整数小于或等于最大的 int 类型整数。C 语言规定 short 类型至少占 16 位。
- long 或 long int——该类型可表示的整数大于或等于最大的 int 类型整数。C 语言规定 long 类型至少占 32 位。
- long long 或 long long int——该类型可表示的整数大于或等于最大的 long 类型整数。Long long 类型至少占 64 位。

一般而言，long 类型占用的内存比 short 类型大，int 类型的宽度要么和 long 类型相同，要么和 short 类型相同。例如，旧 DOS 系统的 PC 提供 16 位的 short 和 int，以及 32 位的 long；Windows 95 系统提供 16 位的 short 以及 32 位的 int 和 long。

无符号整型：

无符号整型只能用于表示零和正整数，因此无符号整型可表示的正整数比有符号整型的大。在整型类型前加上关键字 unsigned 表明该类型是无符号整型：unsignedint、unsigned long、unsigned short。单独的 unsigned 相当于 unsignedint。

字符类型：

可打印出来的符号（如 A、&和+）都是字符。根据定义，char 类型表示一个字符要占用 1 字节内存。出于历史原因，1 字节通常是 8 位，但是如果要表示基本字符集，也可以是 16 位或更大。

- char——字符类型的关键字。有些编译器使用有符号的 char，而有些则使用无符号的 char。在需要时，可在 char 前面加上关键字 signed 或 unsigned 来指明具体使用哪一种类型。

布尔类型：

布尔值表示 true 和 false。C 语言用 1 表示 true，0 表示 false。

- _Bool——布尔类型的关键字。布尔类型是无符号 int 类型，所占用的空间只要能储存 0 或 1 即可。

实浮点类型：

实浮点类型可表示正浮点数和负浮点数。

- float——系统的基本浮点类型，可精确表示至少 6 位有效数字。
- double——储存浮点数的范围（可能）更大，能表示比 float 类型更多的有效数字（至少 10 位，通常会更多）和更大的指数。
- long double——储存浮点数的范围（可能）比 double 更大，能表示比 double 更多的有效数字和更大的指数。

复数和虚数浮点数：

虚数类型是可选的类型。复数的实部和虚部类型都基于实浮点类型来构成：

- float _Complex
- double _Complex
- long double _Complex
- float _Imaginary
- double _Imaginary
- long long _Imaginary

声明简单变量

小结：如何声明简单变量

1. 选择需要的类型。
2. 使用有效的字符给变量起一个变量名。
3. 按以下格式进行声明：

```
类型说明符 变量名;
```

类型说明符由一个或多个关键字组成。下面是一些示例：

```
int erest;
unsigned short cash;
```

4. 可以同时声明相同类型的多个变量，用逗号分隔各变量名，如下所示：

```
char ch, init, ans;
```

5. 在声明的同时还可以初始化变量：

```
float mass = 6.0E24;
```

声明多个变量用逗号隔开

关键字

标识符（变量名、函数名等）基本要求不报错

1. 见名知意

2. 下划线命名法

3. 驼峰命名法

4. 匈牙利命名法

使用数据类型

编写程序时，应注意合理选择所需的变量及其类型。通常，用 int 或 float 类型表示数字，char 类型表示字符。在使用变量之前必须先声明，并选择有意义的变量名。初始化变量应使用与变量类型匹配的常数类型。例如：

```
int apples = 3;         /* 正确 */
int oranges = 3.0;      /* 不好的形式 */
```

许多程序员和公司内部都有系统化的命名约定，在变量名中体现其类型。例如，用 i_前缀表示 int 类型，us_前缀表示 unsigned short 类型。这样，一眼就能看出来 i_smart 是 int 类型的变量，us_versmart 是 unsigned short 类型的变量。

本章小结

C 有多种的数据类型。基本数据类型分为两大类：整数类型和浮点数类型。通过为类型分配的储存量以及是有符号还是无符号，区分不同的整数类型。最小的整数类型是 char，因实现不同，可以是有符号的 char 或无符号的 char，即 unsigned char 或 signed char。但是，通常用 char 类型表示小整数时才这样显示说明。其他整数类型有 short、int、long 和 long long 类型。C 规定，后面的类型不能小于前面的类型。上述都是有符号类型，但也可以使用 unsigned 关键字创建相应的无符号类型：unsigned short、unsigned int、unsigned long 和 unsigned long long。或者，在类型名前加上 signed 修饰符显式表明该类型是有符号类型。最后，_Bool 类型是一种无符号类型，可储存 0 或 1，分别代表 false 和 true。

浮点类型有 3 种：float、double 和 C90 新增的 long double。后面的类型应大于或等于前面的类型。有些实现可选择支持复数类型和虚数类型，通过关键字 _Complex 和 _Imaginary 与浮点类型的关键字组合（如，double _Complex 类型和 float _Imaginary 类型）来表示这些类型。

整数可以表示为十进制、八进制或十六进制。0 前缀表示八进制数，0x 或 0X 前缀表示十六进制数。例如，32、040、0x20 分别以十进制、八进制、十六进制表示同一个值。l 或 L 前缀表明该值是 long 类型，ll 或 LL 前缀表明该值是 long long 类型。

在 C 语言中，直接表示一个字符常量的方法是：把该字符用单引号括起来，如'Q'、'8'和'$'。C 语言的转义序列（如，'\n'）表示某些非打印字符。另外，还可以在八进制或十六进制数前加上一个反斜杠（如，'\007'），表示 ASCII 码中的一个字符。

浮点数可写成固定小数点的形式（如，9393.912）或指数形式（如，7.38E10）。C99 和 C11 提供了第 3 种指数表示法，即用十六进制数和 2 的幂来表示（如，0xa.1fp10）。

printf() 函数根据转换说明打印各种类型的值。转换说明最简单的形式由一个百分号（%）和一个转换字符组成，如%d 或%f。

字符串和格式化输入/输出

字符串简介

字符串（*character string*）是一个或多个字符的序列，如下所示：

```
"Zing went the strings of my heart!"
```

双引号不是字符串的一部分。双引号仅告知编译器它括起来的是字符串，正如单引号用于标识单个字符一样。

C 语言没有专门用于储存字符串的变量类型，字符串都被储存在 char 类型的数组中。数组由连续的存储单元组成，字符串中的字符被储存在相邻的存储单元中，每个单元储存一个字符（见图 4.1）。

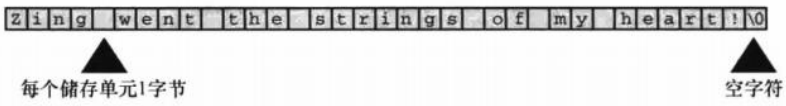

图 4.1 数组中的字符串

注意图 4.1 中数组末尾位置的字符\0。这是空字符（*null character*），C 语言用它标记字符串的结束。空字符不是数字 0，它是非打印字符，其 ASCII 码值是（或等价于）0。C 中的字符串一定以空字符结束，这意味着数组的容量必须至少比待存储字符串中的字符数多 1。因此，程序清单 4.1 中有 40 个存储单元的字符串，只能储存 39 个字符，剩下一个字节留给空字符。

一般而言，C 把函数库中相关的函数归为一类，并为每类函数提供一个头文件。例如，printf() 和 scanf() 都隶属标准输入和输出函数，使用 stdio.h 头文件。string.h 头文件中包含了 strlen() 函数和其他一些与字符串相关的函数（如拷贝字符串的函数和字符串查找函数）。

常量和 C 预处理器

请注意格式，首先是#define，接着是符号常量名（TAXRATE），然后是符号常量的值（0.015）（注意，其中并没有=符号）。所以，其通用格式如下：

```
#define NAME value
```

C90 标准新增了 const 关键字，用于限定一个变量为只读[1]。其声明如下：

```
const int MONTHS = 12; // MONTHS 在程序中不可更改，值为 12
```

这使得 MONTHS 成为一个只读值。也就是说，可以在计算中使用 MONTHS，可以打印 MONTHS，但是不能更改 MONTHS 的值。const 用起来比#define 更灵活，第 12 章将讨论与 const 相关的内容。

printf() 和 scanf()

printf() 函数和 scanf() 函数能让用户可以与程序交流，它们是输入/输出函数，或简称为 *I/O 函数*。它们不仅是 C 语言中的 I/O 函数，而且是最多才多艺的函数。过去，这些函数和 C 库的一些其他函数一样，并不是 C 语言定义的一部分。最初，C 把输入/输出的实现留给了编译器的作者，这样可以针对特殊的机器更好地匹配输入/输出。后来，考虑到兼容性的问题，各编译器都提供不同版本的 printf() 和 scanf()。尽管如此，各版本之间偶尔有一些差异。C90 和 C99 标准规定了这些函数的标准版本，本书亦遵循这一标准。

虽然 printf() 是输出函数，scanf() 是输入函数，但是它们的工作原理几乎相同。两个函数都使用格式字符串和参数列表。我们先介绍 printf()，再介绍 scanf()。

转换说明

表 4.3　转换说明及其打印的输出结果

转换说明	输出
%a	浮点数、十六进制数和 p 记数法（C99/C11）
%A	浮点数、十六进制数和 p 记数法（C99/C11）
%c	单个字符
%d	有符号十进制整数
%e	浮点数，e 记数法
%E	浮点数，e 记数法
%f	浮点数，十进制记数法
%g	根据值的不同，自动选择%f 或%e。%e 格式用于指数小于-4 或者大于或等于精度时
%G	根据值的不同，自动选择%f 或%E。%E 格式用于指数小于-4 或者大于或等于精度时
%i	有符号十进制整数（与%d 相同）
%o	无符号八进制整数
%p	指针
%s	字符串
%u	无符号十进制整数
%x	无符号十六进制整数，使用十六进制数 0f
%X	无符号十六进制整数，使用十六进制数 0F
%%	打印一个百分号

表 4.5　printf()中的标记

标记	含义
-	待打印项左对齐。即，从字段的左侧开始打印该项项 示例："%-20s"
+	有符号值若为正，则在值前面显示加号；若为负，则在值前面显示减号 示例："%+6.2f"
空格	有符号值若为正，则在值前面显示前导空格（不显示任何符号）；若为负，则在值前面显示减号 +标记覆盖一个空格 示例："%6.2f"
#	把结果转换为另一种形式。如果是%o 格式，则以 0 开始；如果是%x 或%X 格式，则以 0x 或 0X 开始；对于所有的浮点格式，#保证了即使后面没有任何数字，也打印一个小数点字符。对于%g 和%G 格式，#防止结果后面的 0 被删除 示例："%#o"、"%#8.0f"、"%+#10.3e"
0	对于数值格式，用前导 0 代替空格填充字段宽度。对于整数格式，如果出现-标记或指定精度，则忽略该标记

!!! 数组与指针!!!

- 如果用 scanf() 读取基本变量类型的值，在变量名前加上一个&；
- 如果用 scanf() 把字符串读入字符数组中，不要使用&。

程序清单 4.15 中的小程序演示了这两条规则。

程序清单 4.15 input.c 程序

```
// input.c -- 何时使用&
#include <stdio.h>
int main(void)
{
    int age;            // 变量
    float assets;       // 变量
    char pet[30];       // 字符数组，用于储存字符串

    printf("Enter your age, assets, and favorite pet.\n");
    scanf("%d %f", &age, &assets);  // 这里要使用&
    scanf("%s", pet);               // 字符数组不使用&
    printf("%d $%.2f %s\n", age, assets, pet);

    return 0;
}
```

表 4.6 ANSI C 中 scanf() 的转换说明

转换说明	含义
%c	把输入解释成字符
%d	把输入解释成有符号十进制整数
%e、%f、%g、%a	把输入解释成浮点数（C99 标准新增了%a）
%E、%F、%G、%A	把输入解释成浮点数（C99 标准新增了%A）
%i	把输入解释成有符号十进制整数
%o	把输入解释成有符号八进制整数
%p	把输入解释成指针（地址）
%s	把输入解释成字符串。从第 1 个非空白字符开始，到下一个空白字符之前的所有字符都是输入
%u	把输入解释成无符号十进制整数
%x、%X	把输入解释成有符号十六进制整数

表 4.7 scanf()转换说明中的修饰符

转换说明	含义
*	抑制赋值（详见后面解释） 示例："%*d"
数字	最大字段宽度。输入达到最大字段宽度处，或第 1 次遇到空白字符时停止 示例："%10s"
hh	把整数作为 signed char 或 unsigned char 类型读取 示例："%hhd"、"%hhu"
ll	把整数作为 long long 或 unsigned long long 类型读取（C99） 示例："%lld"、"%llu"

续表

转换说明	含义
h、l 或 L	"%hd"和"%hi"表明把对应的值储存为 short int 类型 "%ho"、"%hx"和"%hu"表明把对应的值储存为 unsigned shot int 类型 "%ld"和"%li"表明把对应的值储存为 long 类型 "%lo"、"%lx"和"%lu"表明把对应的值储存为 unsigned long 类型 "%le"、"%lf"和"%lg"表明把对应的值储存为 double 类型 在 e、f 和 g 前面使用 L 而不是 l，表明把对应的值被储存为 long double 类型。如果没有修饰符，d、i、o 和 x 表明对应的值被储存为 int 类型，f 和 g 表明把对应的值储存为 float 类型
j	在整型转换说明后面时，表明使用 intmax_t 或 uintmax_t 类型（C99） 示例："%zd"、"%zo"
z	在整型转换说明后面时，表明使用 sizeof 的返回类型（C99）
t	在整型转换说明后面时，表明使用表示两个指针差值的类型（C99） 示例："%td"、"%tx"

sacnf()函数示例

```c
#include <stdio.h>

int main(void)
{
    int num;

    scanf("%d",&num);//里面啥也没有的！！！
    printf("您输入的数字为%d\n",num);

    return 0;
}
```

关键概念

C 语言用 char 类型表示单个字符，用字符串表示字符序列。字符常量是一种字符串形式，即用双引号把字符括起来："Good luck, my friend"。可以把字符串储存在字符数组（由内存中相邻的字节组成）中。字符串，无论是表示成字符常量还是储存在字符数组中，都以一个叫做空字符的隐藏字符结尾。

在程序中，最好用#define 定义数值常量，用 const 关键字声明的变量为只读变量。在程序中使用符号常量（明示常量），提高了程序的可读性和可维护性。

C 语言的标准输入函数（scanf()）和标准输出函数（printf()）都使用一种系统。在该系统中，第 1 个参数中的转换说明必须与后续参数中的值相匹配。例如，int 转换说明%d 与一个浮点值匹配会产生奇怪的结果。必须格外小心，确保转换说明的数量和类型与函数的其余参数相匹配。对于 scanf()，一定要记得在变量名前加上地址运算符（&）。

空白字符（制表符、空格和换行符）在 scanf()处理输入时起着至关重要的作用。除了%c 模式（读取下一个字符），scanf()在读取输入时会跳过非空白字符前的所有空白字符，然后一直读取字符，直至遇到空白字符或与正在读取字符不匹配的字符。考虑一下，如果 scanf()根据不同的转换说明读取相同的输入行，会发生什么情况。假设有如下输入行：

```
-13.45e12# 0
```

如果其对应的转换说明是%d，scanf()会读取 3 个字符（-13）并停在小数点处，小数点将被留在输入中作为下一次输入的首字符。如果其对应的转换说明是%f，scanf()会读取-13.45e12，并停在#符号处，而#将被留在输入中作为下一次输入的首字符；然后，scanf()把读取的字符序列-13.45e12 转换成相应的浮点值，并储存在 float 类型的目标变量中。如果其对应的转换说明是%s，scanf()会读取-13.45e12#，并停在空格处，空格将被留在输入中作为下一次输入的首字符；然后，scanf()把这 10 个字符的字符码储存在目标字符数组中，并在末尾加上一个空字符。如果其对应的转换说明是%c，scanf()只会读取并储存第 1 个字符，该例中是一个空格[1]。

本章小结

字符串是一系列被视为一个处理单元的字符。在 C 语言中，字符串是以空字符（ASCII 码是 0）结尾的一系列字符。可以把字符串储存在字符数组中。数组是一系列同类型的项或元素。下面声明了一个名为 name、有 30 个 char 类型元素的数组：

```
char name[30];
```

要确保有足够多的元素来储存整个字符串（包括空字符）。

字符串常量是用双引号括起来的字符序列，如："This is an example of a string"。

scanf()函数（声明在 string.h 头文件中）可用于获得字符串的长度（末尾的空字符不计算在内）。scanf()函数中的转换说明是%s 时，可读取一个单词。

C 预处理器为预处理器指令（以#符号开始）查找源代码程序，并在开始编译程序之前处理它们。处理器根据#include 指令把另一个文件中的内容添加到该指令所在的位置。#define 指令可以创建明示常量（符号常量），即代表常量的符号。limits.h 和 float.h 头文件用#define 定义了一组表示整型和浮点型不同属性的符号常量。另外，还可以使用 const 限定符创建定义后就不能修改的变量。

printf()和 scanf()函数对输入和输出提供多种支持。两个函数都使用格式字符串，其中包含的转换说明表明待读取或待打印数据项的数量和类型。另外，可以使用转换说明控制输出的外观：字段宽度、小数位和字段内的布局。

运算符、表达式和语句

C 控制语句：循环

C 控制语句：分支和跳转

字符输入/输出和输入验证

函数

数组和指针

字符串和字符串函数