# 数据和C

本章中您将学习下列内容:

* 关键字：

int,short,long,unsigned,char,float,double,\_Bool,\_Complex,

\_Imaginary

* 运算符：

sizeof

* 函数：
* scanf()
* C使用的基本数据类型
* 整数类型和浮点数类型的区别。
* 对上述类型，如何书写常量和声明变量。
* 使用printf()和scanf()函数读写各种数据类型的值。

程序离不开数据。将数字、文字和单词输入计算机，目的是希望计算机能够处理这些数据。例如，要计算机计算利息支付或者显示经过排序的葡萄酒商列表。除了数据读取，本章的内容还包括更有趣的对数据的操作练习。

本章研究数据类型中的两大系列：整数类型和浮点数类型。C语言提供属于这两个系列的多种数据类型。本章介绍这些数据类型的名称、如何声明它们、如何以及何时使用它们。您还将发现常量和变量的区别，并且作为奖励，您将很快看到第一个交互式的程序。

## 3.1 示例程序

这里仍以一个示例程序作为开端。正如同前面章节中那样，我们将解释您感到不熟悉的地方。该程序的大致意图应该是很清晰的，因此请试着编译并运行程序清单3.1中的源代码。为节省时间，输入源代码时，可略去注释。

程序清单 3.1 rhodium.c程序

/\* rhodium.c --用金属铑衡量您的体重 \*/

#include <stdio.h>

int main(void)

{

float weight; /\* 用户体重 \*/

float value; /\* 相等重量的铑的价值 \*/

printf("Are you worth your weight in rhodium?\n");

printf("Let’s check it out.\n");

printf("Please enter your weight in pounds: ");

/\* 从用户处获取输入 \*/

scanf("%f",&weight);

/\* 假设铑为每盎司770美元 \*/

/\* 14.5833把常衡制的英磅转为金衡制的盎司 \*/

value = 770 \* weight \* 14.5833;

printf("Your weight in rhodium is worth $%.2f.\n",value);

printf("You are easily worth that! If rhodium prices

drop.\n");

printf("eat more to maintain your value.\n");

return 0;

}

**错误和警告**

如果输入这个程序的过程中出现错误（error），比如少了一个分号，编辑器会给出语法错误消息。即使输入正确，编译器还可能发出像这样的警告(warning): “警告——从double类型转换为float类型时有可能丢失数据。”错误信息表明程序中存在错误，不能对其编译。警告则表明尽管代码正确但有可能不是程序员所要的。警告不终止编译。这里的警告和C语言怎样处理770这样的值有关。本例不必理会此问题，本章稍后将对这个警告消息进行说明。

输入此程序时您可以把770改为贵金属铑的现价，但是不要改动14.5833，这是相当于1英镑的盎司数。盎司金衡制用于衡量贵金属，而英镑常衡制用于衡量人（无论贵贱）的体重。

注意，“enter your weight”的意思是用来通知计算机输入您的体重值，然后按Enter或Return键（不要只是键入体重后就一直等着）。按回车键是用来通知计算机已经完成了键入回答的工作。程序希望您输入一个数字（比如150），而不是单词（比如 too much）。如果键入字母而非数字，程序运行将产生问题，这个问题需要使用if语句解决（请参见第7章“C控制语句：分支和跳转”）。所以这里先请输入数字。下面是程序的示例输出结果：

Are you worth your weight in rhodium?

Let’s check it out.

Please enter your weight in pounds: **150**

Your weight in rhodium is worth $1684371.12.

You are easily worth that! If rhodium prices drop.

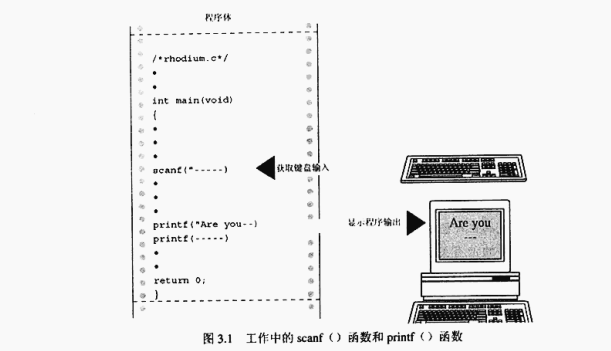
eat more to maintain your value.

**此程序中的新元素**

此程序中包含C语言如下的一些新元素：

* 请注意代码中使用了一种新的变量说明。前面例子中只有整型变量（int）,而本例中还包含了一个浮点变量（float）类型，以便处理更大范围内的数据。float类型可以处理带有小数点的数字。
* 程序还示范了常量的几种新写法，您现在就可以使用带有小数点的数了。
* 要打印这种新的变量类型，请在printf()代码中使用%f说明符来处理浮点值。对%f说明符使用.2修饰词可以精准控制输出格式，使浮点数显示到小数点后两位。
* 使用scanf()函数为程序提供键盘输入。%f指示scanf()从键盘读取一个浮点数，&weight指定将输入值赋于名为weight的变量中。scanf()函数使用&符号指示weight变量的位置。下一章将进一步讨论&符号，现在请相信此处您需要它。
* 也许本程序最突出的新特点是它的交互性。计算机向您询问信息，并使用您输入的数字。与非交互性程序相比较，交互性程序使用起来更为有趣。更重要的是，交互性方法使程序更加灵活。例如，本示例程序可以用于任何合理的体重，而不是150磅。不必每次重写，程序即可针对不同体重进行计算。scanf()和printf()函数使这种交互性成为可能。scanf()函数从键盘读取数据并将其传递给程序，而printf()函数则从程序读取数据并将其传递给屏幕。两个函数一起使用，就可以建立人机之间的双向通信（请参见图3.1），这使计算机的使用更加饶有趣味。

本章解释上述新特性列表中的前两项：各种数据类型的变量和常量。第4章“字符串和格式化输入/输出”将介绍后3项，本章仍将继续使用scanf()和printf()的有限功能。



## 3.2 变量与常量数据

在程序的指示下，计算机可以做很多事，比如数值计算、名字排序、执行语音或视频命令、计算彗星轨道、准备邮寄列表、拨电话号码、画图、做决策以及任何其他可以想象到的事。要完成这些任务，程序需要使用数据，即承载信息的数字与字符。有些数据可以在程序使用之前预先设定并在整个运行过程中没有变化，这称为常量。另外的数据在程序运行过程中可能变化或被赋值，这称为变量。在示例程序中weight是变量，而14.5833则是常量。770.0呢？虽然铑的价格不是一成不变的，但此程序中把它作为常量对待。变量与常量的区别在于，变量的值可以在程序执行过程中变化与指定，而常量则不可以。

## 3.3 数据：数据类型关键字

除了变量和常量的区别，各种数据类型间也有不同。一些数据类型是数字，而另一些则是字母（更广泛的地说是字符）。计算机需要一种方法来区分和使用这些不同的类型。C通过识别一些基本的数据类型做到这一点。如果是常量数据，编译器一般通过其书写来辨认其类型，比如：42是整数，而42.100是浮点数。变量则需要在声明语句中指定其类型。稍后会介绍有关声明变量的详细内容。我们首先了解C语言的基本数据类型。K&R C给出了7个数据类型相关的关键字。C90标准向其中添加了2个关键字，C99标准又添加了另外3个，如表3.1所示。

表3.1 C的数据关键字

|  |  |  |
| --- | --- | --- |
| 原来的K&R关键字 | C90关键字 | C99关键字 |
| int | signed | \_Bool |
| long | void | \_Complex |
| short |  | \_Imaginary |
| unsigned |  |  |

int 关键字提供C使用的基本的整数类型。下面3个关键字（long、short和unsigned）以及ANSI附加的signed 用于提供基本类型的变种。char关键字用于表示字母以及其他字符（如#、$、%，和\*）。char类型也可以表示小的整数。float、double和组合long double 表示带有小数

# 结构和其他数据形式

在本章中您将学习下列内容：

* 关键字：
* struct, union, typedef
* 运算符：

. ->

* 什么是C语言中的结构？如何创建结构模板和结构变量？
* 如何访问结构成员？如何编写处理结构的函数？
* C的typedef工具。
* 联合及指向函数的指针

设计程序最重要的一个步骤就是选择一个表示数据的好方法。在多数情况下，使用简单的变量甚至数组都是不够的。C使用结构变量（structure variable）进一步增强了表示数据的能力。C的结构的基本形式就足以灵活地表示多种数据，并且它还使您能够创建新的形式。如果您熟悉Pascal语言的“记录”的话，您对C语言的结构也不会感到陌生。如果您不熟悉，本章将使您了解C语言中的结构。我们来研究一个具体的例子，看看为什么需要使用结构以及如何创建和使用结构。

## 14.1 示例问题：创建图书目录

Gwen Glenn想要打印她的图书的详细目录。她希望打印出出关于每本图书的各种信息：书名、作者、出版商、版权日期、页数、册数及价格。其中的一些项目（如书名）可以用字符串数组存储，其他的项目需要一个int数组或float数组。使用7个不同的数组来保存所有的信息将是比较复杂的；尤其是如果Gwen还想创建几个完整的列表，一个按书名排序，一个按作者排序，一个按价格排序，等等。一个好的解决方法是使用一个数组，该数组的每个成员包含了一本书的所有信息。

于是，Gwen 需要一种数据形式，其中既可包括字符串又可包括数字，还能分别保存这些信息。C的结构就满足了这种需要。要了解如何建立一个结构以及它如何工作，我们从一个较为简单的示例程序开始学习。为了简化问题，我们试加两个限制条件。首先，每本书的信息只包括书名、作者和当前的市场价格。其次，限制目录中只有一本书。如果您有更多的书，别着急，很快我们就会扩展这个程序。

请看程序清单14.1所示的程序及其输出结果，然后阅读要点解释。

程序清单 14.1 book.c程序

/\*book.c --仅包含一本书的图书目录\*/

#include <stdio.h>

#define MAXTITL 41 /\*书名的最大长度+1\*/

#define MAXAUTL 31 /\*结构模板：标记为book\*/

struct book {

char title[MAXTITL];

char author[MAXAUTL];

float value;

}; /\*结构模板结束\*/

int main(void)

{

struct book library;/\* 把library声明为book类型的变量 \*/

printf("Please enter the book title.\n");

gets(library.title); /\*访问title部分\*/

printf("Now enter the author.\n");

gets(library.author);

printf("Now enter the value.\n");

scanf("%f", &library.value);

printf("%s by %s: $%.2f\n", library.title, library.author,

library.value);

printf("%s:\"%s\" ($%.2f)\n", library.author, library.title,

library.value);

printf("Done.\n");

return 0;

}

下面是一个运行示例：

Please enter the book title.

Chicken of the Alps

Now enter the author.

Bismo Lapoult

Now enter the value.

14.95

Chicken of the Alps Bismo Lapoult:$14.95

Bismo Lapoult:”Chicken of the Alps”($14.95)

程序清单14.1中创建的结构由3部分组成，每个部分称为成员（member）或字段（field）。这3个部分中一个存储书名，一个存储作者名，一个存储价格。下面是必须掌握的3个重要技巧：

* 建立结构的格式或布局。
* 声明遵循该布局的变量。
* 获取对一个结构变量的各个部件的访问。

## 14.2 建立结构声明

结构声明（structure declaration）是描述结构如何组合的主要方法。声明就像下面这样:

struct book{

char title[MAXTITL];

char author[MAXAUTL];

float value;

}

该声明描述了一个由两个字符数组和一个float变量组成的结构。它并没有创建一个实际的数据对象，而是描述了组成这类对象的元素（有时候，我们也把结构声明叫做模板，因为它勾勒出数据该如何存储。如果您已经听过C++模板，那只是这个词的不同用法）。我们来看看细节。首先使用关键字struct，它表示接下来是一个结构。后面是一个可选的标记（单词book）,它是用来引用该结构的快速标记。因此以后我们就可以这样声明：

struct book library;

它把library声明为一个使用book结构设计的结构变量。

在结构声明中，接下来是用一对花括号括起来的结构成员列表。每个成员变量都用它自己的声明来描述，用一个分号来结束描述。例如，title是一个拥有MAXTITL个元素的char数组。每个成员可以是任何一种C的数据类型，甚至可以是其他结构。

结束花括号后的分号表示结构设计定义的结束。可以把这个声明放在任何函数的外面（就像我们已经做的那样），也可以放在一个函数定义内部。如果这个结构声明置于一个函数内部，它的标记只能在函数内部使用。如果是外部声明，它可以被本文件中该声明之后的所有函数使用。例如，若再有一个函数可以这样定义：

struct book dickens;

这样，这个函数中将含有一个book结构的变量dicken。

标记名是可选的。但是再用我们所使用的那种方式建立结构（在一个地方定义结构设计，而在其他地方定义实际的结构变量）时，必须使用标记。我们学习完结构变量的定义之后，再回头来看看这一点。

## 14.3 定义结构变量

词语“结构”（structure）有两个意思。一个意思是“结构设计”，这个我们刚刚已经讨论过了。结构设计告诉编译器如何表示数据，但是它没有让计算机为数据分配空间。下一步是创建一个“结构变量”，即这个词的第二个意思。程序中创建结构变量的那一行如下：

struct book library;

看到这条指令，编译器会创建一个变量library。编译器使用book模板为该变量分配空间：一个具有MAXTTTL个元素的char数组，一个具有MAXAUTL个元素的char数组和一个float变量。这些存储空间是以一个名字library被结合在一起的（请参见图14.1）（下一节将说明在需要的时候如何分解这个存储空间）。

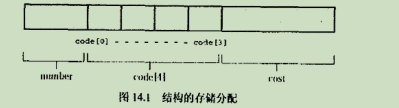
struct stuff{

int number;

char code[4];

float cost;

};



在结构变量的声明中，struct book所起的作用就像int或float在较简单的声明中的作用一样。例如，可以定义两个struct book类型的变量，甚或可以定义一个指向该结构的指针:

struct book doyle, panshin, \* ptbook;

结构变量doyle和panshin均包含title、author和value部分。指针prbook可以指向doyle、panshin,或任何其他book结构变量。实际上，book结构的声明创建了一个名为struct book的新类型。

就计算机而言，下面的声明：

struct book library;

是以下声明的简化：

struct book {

char title[MAXTITL];

char author[AXAUTL];

float value;

}library; /\*在定义之后跟变量名\*/

换句话说，声明结构的过程和定义结构变量的过程可以被合并成一步。如下所示，将声明和变量定义合并在一起，是不需要使用标记的一种情况。

struct book { /\*无标记\*/

char title[MAXTITL];

char author[AXAUTL];

float value;

} library;

然而，如果您想多次使用一个结构模板，就需要使用带有标记的形式：或者您也可以