普通高等教育精品教材 / 国家精品课程主讲教材

C语言程序设计

（第2版）

□ 何钦铭 颜晖 主编

高等教育出版社
HIGHER EDUCATION PRESS

普通高等教育精品教材
国家精品课程主讲教材

C 语言程序设计

C Yuyan Chengxu Sheji

（第2版）

何钦铭　颜　晖　主编

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容提要

本书是为将C语言作为入门语言的程序设计课程编写的以培养学生程序设计基本能力为目标的教材。

教材以程序设计为主线，以编程应用为驱动，通过案例和问题引入内容，重点讲解程序设计的思想和方法，并结合相关的语言知识的介绍。全书主要包括3方面的内容：基本内容（数据表达、数据处理和流程控制）、常用算法和程序设计风格，以及C语言应用中的一些处理机制（编译预处理和命令行参数等）。涉及数据类型、表达式、分支、循环、函数、数组、指针、结构、文件的概念和应用，以及指针和各种构造类型的混合运用，基本算法等内容。

本书可以作为高等学校相关课程和计算机等级考试的教学用书，也可作为对C语言程序设计感兴趣的读者的自学用书。

图书在版编目（CIP）数据

C语言程序设计 / 何钦铭，颜晖主编. -- 2版. -- 北京 : 高等教育出版社, 2012.3
ISBN 978-7-04-034672-5

Ⅰ.①C… Ⅱ.①何… ②颜… Ⅲ.①C语言-程序设计-高等学校-教材 Ⅳ.①TP312

中国版本图书馆CIP数据核字(2012)第023767号

策划编辑 张 龙　　责任编辑 张 龙　　封面设计 张申申　　版式设计 杜微言
插图绘制 尹文军　　责任校对 杨凤玲　　责任印制 韩 刚

出版发行 高等教育出版社
社　　址 北京市西城区德外大街4号
邮政编码 100120
印　　刷 高等教育出版社印刷厂
开　　本 787mm × 1092mm 1/16
印　　张 21.75
字　　数 490千字
购书热线 010-58581118
咨询电话 400-810-0598

网　　址 http://www.hep.edu.cn
　　http://www.hep.com.cn
网上订购 http://www.landraco.com
　　http://www.landraco.com.cn
版　　次 2008年1月第1版
　　2012年3月第2版
印　　次 2012年3月第1次印刷
定　　价 31.80元

物 料 号 34672-00

前　言

程序设计是高校重要的计算机基础课程，它以编程语言为平台，介绍程序设计的思想和方法。通过该课程的学习，学生不仅要掌握高级程序设计语言的知识，更重要的是在实践中逐步掌握程序设计的思想和方法，培养问题求解和语言的应用能力。因此，这是一门以培养学生程序设计基本方法和技能为目标，以实践能力为重点的特色鲜明的课程。

C 语言是得到广泛使用的程序设计语言之一，它既具备高级语言的特性，又具有直接操纵计算机硬件的能力，并以其丰富灵活的控制和数据结构、简洁而高效的语句表达、清晰的程序结构和良好的可移植性而拥有大量的使用者。目前，C 语言被许多高校列为程序设计课程的首选语言。

C 语言程序设计是一门实践性很强的课程，该课程的学习有其自身的特点，听不会，也看不会，只能练会。学习者必须通过大量的编程训练，在实践中掌握语言知识，培养程序设计的基本能力，并逐步理解和掌握程序设计的思想和方法。因此，C 语言程序设计课程的教学重点应该是培养学生的实践编程能力，教材也要以程序设计为中心来组织内容。

虽然目前介绍 C 语言的教材很多，但在多年教学实践中，发现较适合大学程序设计入门课程教学要求的书并不多。现有的很多教材一般围绕语言本身的体系展开内容，以讲解语言知识特别是语法知识为主，辅以一些编程技巧的介绍，不利于培养学生的程序设计能力和语言应用能力。

好的教材源于教学改革和教学实践，能体现教学改革的成果。浙江大学从 1997 年起，就开始实施全方位的程序设计课程的教学改革，目的就是培养学生的程序设计能力，以适应新世纪人才培养的需求。经过多年的建设，在教学内容、教学方法、教学手段和考核方式上，已经基本形成一套比较完整的体系。“C 程序设计基础及实验”课程 2004 年被评为国家精品课程；“以强化实践教学和激发自主学习为手段，提高大学生程序设计能力”的教学改革成果获得浙江省 2005 年教学成果一等奖；主持的教育部高等学校计算机基础课程教学指导委员会课题“程序设计基础课程教学实施方案”的研究成果已经正式出版。《C 语言程序设计》充分展示了浙江大学在程序设计课程教学改革中取得的以上成果，并在 60 多所高校得到广泛采用，2008 年被教育部评为普通高等教育精品教材。

在不断深入的课程改革基础上，结合读者反馈意见，我们对《C 语言程序设计》

进行了修订并推出了第2版。第2版保持了第1版的内容组织结构，重点修订了教材中的引例和示例，除了引言和回顾两章，各章节都设置了引例，引例和示例力求典型、简洁，前后呼应，进一步强化了以程序设计为主线，以案例和问题引入内容的教学设计理念。

本书以程序设计为主线，从编程应用为驱动，通过案例和问题引入内容，重点讲解程序设计的思想和方法，并穿插介绍相关的语言知识。全书共12章，主要包括3方面的内容：基本内容（数据表达、数据处理和流程控制）、常用算法和程序设计风格以及C语言应用中的一些处理机制（编译预处理和命令行参数等）。其中第1～8章侧重基本知识和基本编程能力，包括数据表达中的基本数据类型、简单构造类型和指针，数据处理中的表达式，以及流程控制中的顺序、分支、循环三种语句级控制方式和函数的使用这一单位级控制手段。第9～12章包括指针和各种构造类型的混合运用、文件的使用、用结构化程序设计思想实现复杂问题的编程和基本算法等内容。

在教材的结构设计上，强调实践，使学生从第1周起就练习编程，并贯穿始终。在前两章中，简单介绍一些背景知识和利用计算机求解问题的过程，然后从实例出发，介绍顺序、分支和循环3种控制结构以及函数的使用，使学生对C语言有一个总体的了解，并学习编写简单的程序，培养学习兴趣。从第3章开始，逐步深入讲解程序设计的思想和方法，说明如何应用语言解决问题。

为了提高读者的学习兴趣，对语言知识的介绍一般通过实例程序引入，还将程序设计的技巧、方法以及编程中的常见错误分散在每节的内容中，以“✓”（编程风格）和“☞”（提示）的形式给出。为了鼓励学生多思考、多练习，提高综合能力，本书设计了多种形式的练习题目，节后有练习，章后有习题。节后的练习针对本节涉及的概念和编程，题型多样，难度较低，学生可以即学即练，加深理解，提高兴趣；章后的习题主要是综合性的题目，包括本章的综合，以及从第1章到本章的综合，以程序设计题为主。

本书第1章介绍程序与程序设计语言的知识以及利用计算机求解问题的过程；第2章从实例出发，简单介绍顺序、分支和循环三种控制结构及函数的使用，以及在实例程序中用到的语言知识；第3～5章通过大量的例题，分别讲解分支、循环结构以及函数程序设计的思路和方法；第6章介绍数据类型和表达式的作用；第7章通过3个典型示例介绍一维数组、二维数组和字符串的应用；第8章介绍指针的基本概念；第9章用案例说明结构类型在编程中的应用；第10章讲解函数和程序结构方面有一定深度的内容；第11章介绍指针和数组、指针与结构以及其他构造类型的概念及其在编程中的应用；第12章介绍文件的使用。附录将散布在全书各个章节中的数据类型、表达式和控制结构等内容作了归纳性的汇总，使读者对C语言的数据表达、数据处理和流程控制有一个清楚的认识。本书目录中打星号的部分是扩展知识，读者可酌情有选择性地学习。

在课程教学中，建议将内容分为16个教学单元：第1章和第2章为3～4个单元，第3～12章一般每章为1～2个单元。

我们还编写了配套的《C语言程序设计实验与习题指导（第2版）》一书，其中，实验指导

部分有13个实验，包括20个实验项目和一个综合实验，每个实验都提供精心设计的编程示例或调试示例，以及实验题（编程题和改错题）。读者可以先模仿示例操作，然后再做实验题，通过“模仿—改写—编写”的上机实践过程，在循序渐进的引导中逐步熟悉编程环境，理解和掌握程序设计的思想、方法和技巧，以及基本的程序调试方法。习题部分则包括与教材配套的选择题、填空题及参考答案，以帮助读者巩固各章节知识点。

由于我们参与了教育部高等学校计算机基础课程教学指导委员会“计算机基础实验经典实验案例集”的编制工作，由浙江大学主编的《C语言程序设计经典实验案例集》一书也可以作为本课程学习的辅助资料，特别是对于“课程设计”环节的学习有很好的指导作用。

此外，本书还提供了大量的数字资源。读者可以登录浙江大学的国家精品课程展示网站http://jpkc.zju.edu.cn/共享优质教学资源；访问浙江大学ACM程序设计网站http://acm.zju.edu.cn/，参加具有较高难度的程序设计训练和竞赛，开阔眼界，锻炼能力。

为方便课程教学，我们编写了《C语言程序设计教师用书（第2版）》，包括各章的教学要点、PPT讲稿、练习与习题答案、实验与习题指导教材参考答案等4部分内容。同时也研制了具有在线判题功能的“C语言程序设计上机练习系统”，作为学生上机练习、考核的平台。需要教师用书或练习系统的教师可以直接发送邮件到：jsj@pub.hep.cn。

本书由何钦铭教授和颜晖教授主编并统稿，何钦铭、颜晖、柳俊、杨起帆、张高燕、吴明晖、张泳、陈建海和罗国明共同参加了编写工作。

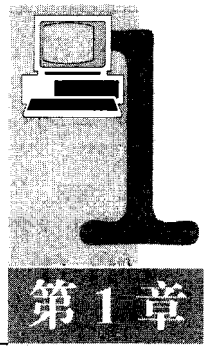
计算机科学和技术在不断发展，计算机教学的研究和改革也从未停顿。希望在从事计算机基础教学的各位同仁的共同努力下，能不断提高我国高等学校C语言程序设计课程的教学质量和水平。

由于作者水平所限，书中难免存在谬误之处，敬请读者指正并与我们联系：yanhui@zju.edu.cn。

编　者

2012年1月

目 录



第1章 引 言

本章要点

- 什么是程序？程序设计语言一般包含哪些功能？
- 程序设计语言在语法上一般包含哪些内容？
- 结构化程序设计有哪些基本的控制结构？
- C 语言有哪些特点？
- C 语言程序的基本框架是怎样的？
- 形成一个可运行的 C 语言程序需要经过哪些步骤？
- 如何应用流程图描述简单的算法？

对于将 C 语言作为第一门编程语言（Programming Language）的读者来说，最关心的问题是如何尽快学会用 C 语言进行程序设计。要做到这一点，对程序设计语言（如 C 语言）要有所了解，更重要的是通过不断的编程实践，逐步领会和掌握程序设计的基本思想和方法。

熟练的编程技能是在知识与经验不断积累的基础上培养出来的。初学者一开始由于缺乏足够的语言知识和编程经验，对于很简单的问题往往也会感到无所适从，不知如何下手编写程序。本书建议读者从一开始学习 C 语言起就要试着编写程序，可以先模仿教材中的程序，试着改写它并循序渐进，直到会独立地编写程序解决比较复杂的问题。

为了使读者能逐步从简单的模仿中体会程序设计的基本思想和方法，而不是拘泥于具体的语法细节，本章作为教材的引言，将简要介绍程序设计语言的功能、语法要素、C 语言的特点以及程序设计求解问题的一般步骤等。

1.1 一个C语言程序

为了让读者对C语言有一个感性认识，首先来看一下用C语言编写的一个程序。

【例1-1】 求阶乘问题。输入一个正整数 n，输出 $n!$。

源程序

```
#include <stdio.h>                    /* 编译预处理命令 */
int main(void)                        /* 主函数 */
{
    int n;                            /* 变量定义 */
    int factorial(int n);             /* 函数声明 */

    scanf("%d", &n);                  /* 输入一个整数 */
    printf("%d\n", factorial(n));     /* 调用函数计算阶乘 */

    return 0;
}

int factorial(int n)                  /* 定义计算 n! 的函数 */
{
    int i, fact = 1;

    for(i = 1; i <= n; i++)
        fact = fact*i;

    return fact;
}
```

运行程序，输入4，输出24，即4的阶乘。

上述程序并不要求初学者能完全理解，但希望读者能对C程序有个初步的印象。该程序中的许多内容将会在随后各章中逐步介绍。

C程序由函数（Function，一种子程序）所组成。上述程序涉及4个函数：main()、factorial()、scanf()和printf()。其中，scanf()和printf()是系统事先设计好的函数，分别用于数据的输入和输出；factorial()是程序中定义的函数，主要目的是求 $n!$，并将n作为函数的参数；main()函数是程序的主函数。

所有的C程序都有且只有一个main()函数。C程序从main()函数处开始运行，当main()函数结束时，程序也就结束了。对于上述例子，程序先执行main()函数中的scanf()函数调用，输入数据n，然后调用printf()函数，输出结果。当调用printf()函数时，必须要先知道所要输

出的数据，即 factorial(n)。因此，此时发生了对 factorial( )函数的调用（Call），调用该函数所获得的结果作为 printf( )函数的参数（Argument），由 printf( )函数负责将该值按十进制整数（%d）的形式输出。

程序最根本的功能是对数据的处理，为此，首先要解决待处理数据的表示问题。在 factorial( )函数中，用整数类型变量（Variable）n 表示要求阶乘的整数，同样在 main( )函数中也用 n 来表示（用别的变量名字也可以，如 m）。同时，在 factorial( )函数中，用 fact 存储计算的结果，用 i 表示 1 到 n 之间的某个整数。

程序还需要对数据处理的过程进行控制。上述程序中，最主要的控制是将 1 到 n 的每个整数 i 乘到结果变量 fact 中。这个控制通过 for 循环语句（Loop Statement）来实现。

变量都有类型（如整数类型 int），并在内存中占有一定的空间，例如在 Visual C++ 中，整数变量占用 4 个字节的空间。因此，每个整数都有一定的取值范围。运行上述程序，输入整数 13，其结果（13!）就超出了整数的取值范围，会输出一个错误的结果。

1.2 程序与程序设计语言

计算机程序（Program）是人们为解决某种问题用计算机可以识别的代码编排的一系列加工步骤。计算机能严格按照这些步骤去做，包括计算机对数据的处理。程序的执行过程实际上是对程序所表示的数据进行处理的过程。一方面，程序设计语言提供了一种表示数据与处理数据的功能；另一方面，编程人员必须按照语言所要求的规范（即语法规则）进行编程。

1.2.1 程序与指令

计算机最基本的处理数据的单元应该就是计算机的指令了。单独的一条指令本身只能完成计算机的一个最基本的功能，如实现一次加法运算或实现一次大小的判别。计算机所能实现的指令的集合称为计算机的指令系统。

虽然计算机指令所能完成的功能很基本，并且指令系统中指令的个数也很有限。但一系列指令的组合却能完成一些很复杂的功能，这也就是计算机的奇妙与强大功能所在。一系列计算机指令的有序组合就构成了程序。

假设某台虚拟的计算机指令系统有以下 7 条指令，每条指令的第一部分是指令名（如 Store，Add 等），随后的几个部分是指令处理所涉及的数据（如 X, Y, Z, P 等）。

指令 1：Input X；将当前输入数据存储到内存的 X 单元。

指令 2：Output X；将内存 X 单元的数据输出。

指令 3：Add X Y Z；将内存 X 单元的数据与 Y 单元的数据相加并将结果存储到 Z 单元。

指令 4：Sub X Y Z；将内存 X 单元的数据与 Y 单元的数据相减并将结果存储到 Z 单元。

指令 5：BranchEq X Y P；比较 X 与 Y，若相等则程序跳转到 P 处执行，否则继续执行下一条指令。

指令 6：Jump P；程序跳转到 P 处执行。

指令 7：Set X Y；将内存 Y 单元的值设为 X。

对上述 7 条简单的指令进行不同的组合就可以实现多项功能。

【例 1-2】 编写由上述指令组成的虚拟的程序，实现以下功能：

（1）输入 3 个数 A, B 和 C，求 A+B+C 的结果。

（2）输入 2 个数 A 和 B，求 A*B 的结果。

虚拟程序 1

```
Input A;          输入第 1 个数据到存储单元 A 中
Input B;          输入第 2 个数据到存储单元 B 中
Input C;          输入第 3 个数据到存储单元 C 中
Add A B D;        将 A、B 相加并将结果存在 D 中
Add C D D;        将 C、D 相加并将结果存在 D 中
Output D;         输出 D 的内容
```

虚拟程序 2

```
1. Input A;          输入第 1 个数据到存储单元 A 中
2. Input B;          输入第 2 个数据到存储单元 B 中
3. Set 0 X;          将 X 设为 0，此处 X 用以统计 A 累加的次数
4. Set 0 Z;          将 Z 设为 0，此处 Z 用以存放 A*B 的结果
5. BranchEq X B 9;   判别 X 与 B 是否相等；若相等说明 A 已累加了 B 次，程序跳转到第 9 条
                     指令，输出结果
6. Add Z A Z;        Z = Z + A
7. Add 1 X X;        X = X + 1
8. Jump 5;           程序跳转到第 5 条指令，继续循环执行第 6 条、7 条指令
9. Output Z;         输出 Z 的值，该值等于 A*B
```

把 A 累加 B 次就可以获得 A*B 的结果，虚拟程序 2 反复执行加法运算 Z = Z + A，实现两个整数 A 和 B 的乘法功能。A 和 B 用于存储准备相乘的两个数（第 1 和第 2 条指令），Z 用于存储乘法运算的结果，开始时 Z 初始化为 0（第 4 条指令），随后不断将 A 累加到 Z 上（第 6 条指令）。累加次数的控制通过 X 来实现，X 开始时设为 0（第 3 条指令），随后 A 每累加一次就将 X 加 1（第 7 条指令），当 X 被累加 B 次时（第 5 条指令），Z 的值就是 A*B 的结果，最后输出结果 Z（第 9 条指令）。

一般情况下程序是按指令排列的顺序一条一条地执行，但稍微复杂一点的程序往往需要通过判断不同的情况执行不同的指令分支，如第 5 条指令（BranchEq），还有些指令需要被反复地执行，如第 6、7 两条指令，这时就需要强行改变程序中指令执行的顺序，如第 8 条指令（Jump）。

程序在计算机中是以0、1组成的指令码来表示的，即程序是0、1组成的序列，这个序列能够被计算机所识别。程序与数据一样，共同存放在存储器中。当程序要运行时，当前准备运行的指令从内存被调入CPU中，由CPU处理这条指令。这种将程序与数据共同存储的思想就是目前绝大多数计算机采用的冯·诺依曼模型的存储程序概念。

为什么程序一定要由计算机中的指令组成呢？一方面，通过定义计算机可直接实现的指令集使得程序在计算机中的执行变得简单，计算机硬件系统只要实现了指令就能方便地实现相应的程序功能；另一方面，需要计算机实现的任务成千上万，如果每一个任务都相对独立，与其他程序之间没有公共的内容，编程工作将十分困难。这就是计算机科学中一个很重要的概念“重用”的体现。

如果程序设计直接用0、1序列的计算机指令来写，那将是一件难以忍受的事。所以，人们设计了程序设计语言，用这种语言来描述程序，同时应用一种软件（如编译程序）将用程序设计语言描述的程序转换成计算机能直接执行的指令序列。

总的来说，计算机程序是人们为解决某种问题用计算机可以识别的代码编排的一系列数据处理步骤，计算机能严格按照这些步骤去做。

1.2.2 程序设计语言的功能

程序设计语言是人用来编写程序的手段，是人与计算机交流的语言。人为了让计算机按自己的意愿处理数据，必须用程序设计语言表达所要处理的数据和数据处理的流程。因此，程序设计语言必须具有数据表达和数据处理（称为控制）的能力。

1. 数据表达（Data Representation）

世界上的数据多种多样，而语言本身的描述能力总是有限的。为了使程序设计语言能充分、有效地表达各种各样的数据，一般将数据抽象为若干种类型。数据类型（Data Type）就是对某些具有共同特点的数据集合的总称。如人们常说的整数、实数就是数据类型的例子。数据类型涉及两方面的内容：该数据类型代表的数据是什么（数据类型的定义域）？能在这些数据上做什么（即操作，或称运算）？例如，我们熟悉的整数类型所包含的数据是{…，−2，−1，0，1，2，…}，而+、−、*、/ 就是作用在整数上的运算。

在程序设计语言中，一般都事先定义几种基本的数据类型，供程序员直接使用，如整型、实型（浮点型）、字符型等。这些基本数据类型在程序中的具体对象主要是两种形式：常量（又称常数，Constant）和变量（Variable）。常量值在程序中是不变的，例如123是一个整型常量，12.3是一个实型常量，'a'是一个字符型常量。变量则可对它做一些相关的操作，改变它的值。例如，在C语言中可以通过int i来定义一个新的变量i，然后就可以对该变量进行某种操作，如赋值i = 20。

同时，为了使程序员能更充分地表达各种复杂的数据，程序设计语言还提供了构造新的具体数据类型的手段，如数组（Array）、结构（Structure）、文件（File）、指针（Pointer）等。例如，在C语言中可以通过int a[10]来定义一个由10个整数组成的数组变量。这时，变量a所代

表的就不是一个整数，而是10个整数组成的有序序列，其中的每一个整数都称作a的分量。

程序设计语言提供的基本数据类型以及构造复杂类型（如数组、结构等）的手段，为有限能力的程序设计语言表达客观世界中多种多样的数据提供了良好的基础。

2. 流程控制（Flow Control）

程序设计语言除了能表达各种各样的数据外，还必须提供一种手段来表达数据处理的过程，即程序的控制过程。程序的控制过程通过程序中的一系列语句来实现。

当要解决的问题比较复杂时，程序的控制过程也会变得十分复杂。一种比较典型的程序设计方法是：将复杂程序划分为若干个相互独立的模块（Module），使完成每个模块的工作变得单纯而明确，在设计一个模块时不受其他模块的牵连。同时，通过现有模块积木式的扩展就可以形成复杂的、更大的程序模块或程序。这种程序设计方法就是结构化的程序设计方法（Structured Programming）。C语言就是支持这种设计方法的典型语言。

在结构化程序设计方法中，一个模块可以是一条语句（Statement）、一段程序或一个函数等。

一般来说，从程序流程的角度看，模块只有一个入口和一个出口。这种单入单出的结构为程序的调试（Debug，又称查错）提供了良好的条件。多入多出的模块结构将使得程序的调试变得非常困难。

按照结构化程序设计的观点，任何程序都可以将模块通过3种基本的控制结构进行组合来实现。这3种基本的控制结构是顺序、分支和循环。

（1）顺序控制结构（Sequential Control Structure）：一个程序模块执行完后，按自然顺序执行下一个模块。

（2）分支控制结构（Branch Control Structure）：又称选择结构。计算机在执行程序时，一般按照语句的顺序执行，但在许多情况下需要根据不同的条件来选择所要执行的模块，即判断某种条件，如果条件满足就执行某个模块，否则就执行另一个模块。例如，在周末，我们根据天气情况决定是去郊游还是在房间里学习，这就是一种分支控制。

（3）循环控制结构（Loop Control Structure）：有时，需要反复地执行某些相同的处理过程，即重复执行某个模块。重复执行这些模块一般是有条件的，即检测某些条件，如果条件满足就重复执行相应的模块。

顺序结构是一种自然的控制结构，通过安排语句或模块的顺序就能实现。所以，对一般程序设计语言来说，需要提供表达分支控制和循环控制的手段。

例1-2中求两个整数A和B的乘积，该程序实际上存在着一个问题：当B为负数时，该程序将不能终止！当B是负数时，要得到正确的结果，可以将A和B均乘上-1，再应用例1-2中的思路求解。其求解过程描述如下：

① 分别输入两个数到A、B两个变量

② 如果B是负数，那么B=B*（-1），A=A*（-1）

③ 设X=0，Z=0

④ 当X不等于B时，重复做以下操作：

```
Z = Z + A;
X = X + 1;
```

⑤ 输出 Z

上述处理过程从步骤①顺序做到步骤⑤，其中步骤②是分支控制，而步骤④是循环控制。

以上 3 种控制方式称为语句级控制，实现了程序在语句间的跳转。当要处理的问题比较复杂时，为了增强程序的可读性和可维护性，常常将程序分为若干个相对独立的子程序，在 C 语言中，子程序的作用由函数完成。函数通过一系列语句的组合来完成某种特定的功能（如求整数 n 的阶乘）。当程序需要相应功能时，不用重新写一系列代码，而是直接调用函数，并根据需要传递不同的参数（如求阶乘函数中的 n）。同一个函数可以被一个或多个函数（包括自己）多次调用。函数调用时可传递零个或多个参数，函数被调用的结果将返回给调用函数。这种涉及函数定义和调用的控制称为单位级控制。所以，程序设计语言的另一个功能就是提供单位级控制的手段，即函数的定义（Definition）与调用手段。

1.2.3 程序设计语言的语法

程序员用程序设计语言编写程序以处理相应的问题。在程序中，一般要表达数据，包括定义用于存储数据的变量；还要描述数据处理的过程，包括语句级的控制和单位级的控制。为了让计算机能理解程序员在程序中所描述的这些工作，用程序设计语言所写的程序必须符合相应语言的语法（Grammar）。

编写程序就像用某种自然语言（如中文）来写文章，首先语法要通，即要符合语言所规定的语法规则。当然，语法通了并不意味着你的文章就符合要求了，你有可能词不达意、离题万里。后者就是在程序调试（查错）时需要发现的事，即找出程序中的错误（非语法错误）。这是一个需要耐心和经验的过程。而语法错误的检查则相对容易得多。

一般把用程序设计语言编写的未经编译的程序称为源程序（Source Code，又称源代码）。从语法的角度看，源程序实际上是一个字符序列。这些字符序列按顺序分别组成了一系列“单词”。这些“单词”包括语言事先约定好的保留字（Reserved Words，如用于描述分支控制的 if、else，用于描述数据类型的 int 等）、常量、运算符（Operator）、分隔符以及程序员自己定义的变量名、函数名等。

在这些“单词”中，除了运算符（如 +、−、*、/）、普通常量（如−12、12.34、'a'）、分隔符（如;、(、) 等）外，其他主要是一些用来标识（表示）变量、函数、数据类型、语句等的符号，这些标识符号称为标识符（Identifier）。任何程序设计语言对标识符都有一定的定义规范，只有满足这些规范的字符组合才能构成该语言所能识别的标识符。

例 1-1 程序中，include、int、factorial、n、fact、for、i、return 等都是标识符。

“单词”的组合形成了语言有意义的语法单位，如变量定义、表达式（Expression）、语句、函数定义等。一些简单语法单位的组合又形成了更复杂的语法单位，最后一系列语法单位组合成程序。就像在作文中，单词组合成主语、谓语、宾语等，而主语、谓语、宾语又组合成句子，