New Perspective for C Programming

C语言程序设计新视角

周幸妮　编著 >>

西安电子科技大学出版社
http://www.xduph.com

C 语言程序设计新视角

New Perspective for C Programming

周幸妮　编著

西安电子科技大学出版社

内 容 简 介

程序设计是给出用计算机解决特定问题的过程，具体是用计算机编程语言实现的。本书以通俗易懂的语言介绍了编程语言之一——C 语言的语法基础以及开发环境，并且运用大量程序实例深入浅出地阐明了程序设计的基本方法与技巧。本书把重点放在对程序的设计方法及调试要点的讲解上，而非对基本语法的简单罗列。全书图(表)文并茂，生动简洁。

本书共 10 章。第 1 章简要介绍了程序设计的基本概念与基本方法；第 2～9 章在依序讲解 C 语言基础知识的同时，循序渐进地引入了程序设计的步骤、方法、要领等；第 10 章对 C 语言的开发环境 VC6.0 做了简要介绍，并给出了在开发环境中进行程序调试的基本方法。

本书可供相关专业的本、专科学生以及低年级研究生作为教材使用，也可供自学计算机编程的读者参考。

图书在版编目(CIP)数据

C 语言程序设计新视角/周幸妮编著. —西安：西安电子科技大学出版社，2012.12

ISBN 978-7-5606-2960-5

Ⅰ. ① C… Ⅱ. ① 周… Ⅲ. ① C 语言—程序设计 Ⅳ. ① TP312

中国版本图书馆 CIP 数据核字(2012)第 282698 号

策　　划　戚文艳

责任编辑　雷鸿俊　戚文艳

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)88242885　88201467　邮　　编　710071

网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com

经　　销　新华书店

印刷单位　陕西华沐印刷科技有限责任公司

版　　次　2012 年 12 月第 1 版　2012 年 12 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印张 24.5

字　　数　578 千字

印　　数　1～2000 册

定　　价　42.00 元

ISBN 978 – 7 – 5606 – 2960 – 5 / TP

XDUP 3252001-1

前　言

“老师，书里那些语法规则之类的概念我都知道，也记得很清楚，编程的例子也都能看明白，但最后还是不会自己编程序，总觉得编程是很难掌握的事情，真的高不可攀啊……”

——这是我的学生在学习 C 语言课程时常常对我说的。多年的程序设计教学，使我发现，用传统的 C 语言教材教授课程，学生们学完后，仍没有掌握程序设计也就是编程的方法，这是一个普遍存在的现象。这也促使我在教学过程中不断地思考、不断地改进教学方法，目的是让学生在课堂上真正学会编程。

“老师，您上课时所用的教学方法，那些漂亮的、形象的 PPT，对我们太有帮助了，您为什么不总结总结，编出一本风格独特的教材帮助大家，让更多的人受益。”

——这是一位获全国信息学奥赛联赛一等奖的学生的建议，也是我编写这本教材的初衷和最终完成它的动力。

本书是作者多年教学实践经验的总结，其主要思想是针对问题从而设计解决问题之道。

➢ 学生学习时的主要问题

- 概念理解不透彻；
- 读程困难；
- 编程困难；
- 不会调试；
- 没有测试的概念；
- 程序设计思想不易把握。

➢ 本书特别设计的解决方法

- 概念理解不透彻——

(1) 注重介绍编程相关机制与规则是如何从实际问题而来的背景；

(2) 注重做类比和归纳(如数组、结构与单个变量的类比等)；

(3) 要素提炼(如循环三要素、函数设计三要素、程序设计要素)。

- 读程困难——设计列表法，分析程序的执行步骤。
- 编程困难——将“自顶向下、逐步细化”的方法贯彻在大多数实际程序设计的例子中。
- 不会调试——增加程序调试环节，对程序中的经典情形都有实际跟踪调试的介绍。
- 没有测试的概念——增加测试概念的介绍。
- 程序设计思想不易把握——通过实例，总结出程序设计适用的要素。

➢ 本书与同类书籍的比较

- 传统的程序设计语言教材基本上都是以高级语言自身的体系为脉络展开教学的，重点都是注重语法规则、基本概念之类的知识点的细节讲解。如何从一个问题入手，如何从

实现的角度去看程序设计问题，如何具体设计算法，传统的教材没有总结出一个一般性的方法，即语法规则有余而设计思想不足，因此，学生们普遍的反映就是学了程序设计语言，只会一些语法规则，而很难很快掌握编程的方法。再者，程序是“一分编三分调”，一般的教材都只给出调试环境的说明，而欠缺调试方法和技巧的介绍。而且传统教材缺乏程序测试概念的介绍。

• 本书从学以致用的角度，强调程序的设计思路、分析方法、测试及调试方法，弥补传统教材中的不足，教学生以“程序的思维”看问题，快速理解并掌握程序设计方法，达到课程期望的目标。

➢ 本书的特色

• 先从大的面入手，从开始就给出如何做程序设计的整体印象，然后逐步深化，让初学者能快速把握整体框架，树立信心，形成初步应用能力。

• 注意分析各种机制设置的本质原因，不仅能让读者清晰地理解各个概念，还通过比较与类比，把握住相关概念间的联系，有利于读者建立系统的思路。

• 把重点从详细讨论 C 语言的语法及各种规则转移到编程方法论及具体实现上，让读者知道哪些是需要重点记忆的规则，哪些不必强记，使用时知道用什么关键字去查阅手册即可。

• 设置问答环节——注重方法的引导及思维的启发。
• 设置上机调试环节——在相关知识点间穿插调试方法及测试方法的介绍。
• 设置读程环节——介绍读程分析方法，注重训练快速有效的读程能力。
• 强调代码风格，培养良好的软件工程习惯。
• 示例简洁，要点突出。
• 注重语言的生动性和可理解性，适合初学者。

➢ 本书的主要内容

第 1 章　走马观花看编程。本章不仅给出了算法和自顶向下分析法的概念，而且结合相应实例，给出了程序设计的全貌。

• 着重让读者了解计算机解题的特点；
• 运用❽自顶向下、逐步细化”的方法分析问题并在之后的所有章节都使用这种方法；
• 提出程序列表分析法；
• 总结算法的普遍规律；
• 介绍软件测试的概念；
• 设置关于做算法的习题，以帮助读者逐步建立算法设计的思想。

第 2 章　程序中的数据。本章在介绍数据类型使用规则及方法的基础之上，给出了概述性的归类总结。

• 数据要素的归类总结；
• 给出数据类型的本质含义，重点介绍整型和实型存储机制的不同点；
• 强调各类运算的结果归类(这点非常重要，否则在后续的使用中会造成很多重要概念的不清楚)；

• 给出数据表现形式的总结和需要记忆的知识点。

第 3 章　程序语句。本章在介绍语句语法规则及使用方法的基础之上，给出了同类语句的比较及共性总结。

• 同类语句的特点、相互间的联系、选用条件等；
• 流程图分析程序的效率训练；
• 自顶向下算法设计的训练；
• 读程序的训练；
• 介绍语句调试要点。

第 4 章　数组。本章在介绍数组的概念、使用规则及方法实例的基础之上，注意强调数组与普通变量的不同。

• 数组和数组元素与单个变量的类比，说明其表现形式与本质含义；
• 数组的空间存储特点及调试要点；
• 多维数组的编程要点；
• 自顶向下算法设计的训练。

第 5 章　函数。本章在给出函数的三种形式、函数参数的含义、使用规则及方法实例的基础上，揭示了函数的本质特点、设计要素的把握方法。

• 函数三种形式机制设置的原因及原理，它们之间的相互关系的本质含义、多函数实现与一个函数实现的不同，会引发的问题及解决的方法；
• 用多个函数实现功能的设计要素；
• 读程序的训练；
• 自顶向下算法设计的训练；
• 介绍函数间信息传递调试要点。

第 6 章　指针。本章在给出指针的含义、使用规则及方法实例的基础上，将指针与普通变量做类比，使之易理解和掌握。

• 通过指针变量与普通变量的类比，说明其表现形式与本质含义并说明指针变量与普通变量的不同之处以及使用的相同之处；
• 强调指针偏移量的本质含义；
• 读程序的训练；
• 自顶向下算法设计的训练；
• 介绍指针调试要点。

第 7 章　复合的数据类型。本章在给出结构体类型及变量的定义、使用规则及方法实例的基础之上，对相关概念做了比较和总结。

• 通过结构与数组的类比，说明其表现形式与本质含义；
• 通过结构体类型与基本类型的类比，说明其表现形式与本质含义；
• 通过结构成员与普通变量的类比，给出其使用规则；
• 读程序的训练；
• 自顶向下算法设计的训练；
• 结构的空间存储特点及调试要点。

第 8 章　文件。本章给出文件的定义、文件的操作步骤、文件操作库函数的介绍及使

用方法实例。

- 通过人工操作文件与程序操作文件的类比，说明文件操作的基本步骤；
- 分类简要介绍可以对文件操作的库函数功能。

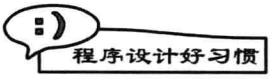
第 9 章　编译预处理。本章简要介绍以下三部分内容：

- 宏定义；
- 文件包含；
- 条件编译。

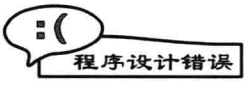
第 10 章　程序调试及测试。

- 程序开发环境 VC6.0 介绍；
- 程序测试方法；
- 程序调试方法。

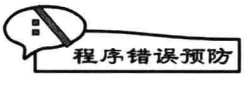
➢ 书中的符号约定



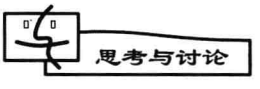
编写具有更高可读性和更易维护的程序。



对常见的程序错误提高注意力。



可以减少 bug，从而简化测试和调试过程。



提出问题，引起思考或展开讨论。

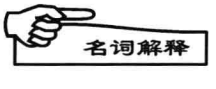
给出程序的调试步骤。

给出需要掌握的重要结论。

给出相关内容的知识背景或扩展内容的介绍。

给出程序和运行结果，让读者自己做读程练习。

给出编程中的一些规则。



给出名词的定义及相关概念的解释。

➢ 致谢

大概在十年前，在我最初做教师时，常会和教了一辈子高等数学的父亲讨论一些 C 语言的教学方法和设想。他觉得有些思路很好，鼓励我写出来，可当时觉得只有一点点体会，总没有信心。

在随后多年的 C 语言和数据结构的教学中，通过与学生的互动，逐渐发现学生们学习编程的困难所在，由此不断地改进授课的方法与技巧，以期让学生们容易掌握编程的思想与方法。

在两年前给教改班的学生上数据结构课程时，屈宇澄同学给我建议说，老师上课的思路不错，可以写本数据结构的书，应该会比较有特色。这时，我也觉得好像应该可以写些东西了，由于数据结构的内容毕竟比 C 语言要深一些，但二者不少授课的基本思想是一致的，所以我决定还是先写 C 语言的内容。

在本书的写作过程中，屈宇澄同学完成了全部作业题目、部分程序样例及附录文档的收集整理工作，并调试了相关程序，对书稿提出了许多建设性的建议；屠仁龙同学做了部分样例的修改和测试，并认真阅读了最初的书稿，提出了许多建设性的建议；孙蒙、袁斌、黄山等同学也阅读了最初的书稿，提出了许多建设性的建议。

感谢我的父亲，让我有了最初的梦想；感谢我的学生们，让这个梦想得以实现。师生情谊的温暖，令人感动，你们的支持和帮助，是本书得以完成的最大动力；你们的鼓励，激励着我去尽力完成这本以方便初学者入门为初衷的书。愿此书像烛光，照亮学习者探索的路，在学习编程的过程中，少些困扰多些乐趣，得到享受。

西安电子科技大学出版社的戚文艳、雷鸿俊编辑为本书的编辑出版做了大量工作，张香梅、刘芳侠、刘非为本书的排版付出了艰辛的劳动，西安电子科技大学出版社为本书的出版提供了帮助与支持。在此向所有相关人员及单位表示衷心的感谢！

本书的编写得到了西安电子科技大学教材基金的资助。

由于水平有限，书中不足之处在所难免，欢迎广大读者提出宝贵意见。若想获得本书的源程序或需交流有关问题，可通过下列 E-mail 直接与作者联系：xnzhou@xidian.edu.cn。

西安电子科技大学　周幸妮

2012 年 6 月于长安

目　录

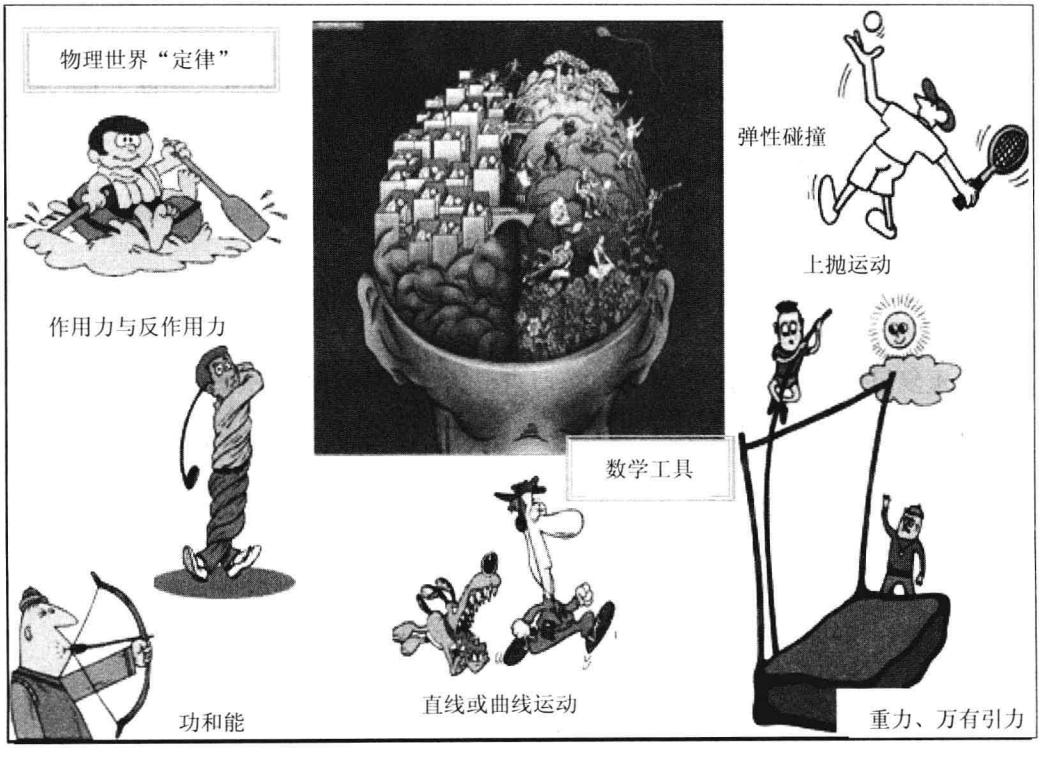
引　言

大千世界，千差万别，人类在进化过程中学会了许多发现问题和解决问题的途径和方法，但是，当我们希望用电脑的智慧去处理这些问题的时候，人的大脑所习惯的方法未必适合机器去实施。“程序设计”就是借助人脑的智慧结合机器的特点来寻求问题的解决之道。

1. 人脑的惯性思维

> 我们的许多观念都是如此深地陷入理当如此的假定中，以至于在正常情况下，我们根本不会想到要去质疑它们。
>
> ——(美)迈克尔·施瓦布(Michael Schwalbe)
> (《生活的暗面——日常生活的社会学透视》的作者)

人们的大脑里塞满了物理定律。电脑里面装满了软件，按“软件定律”运行。软件运行原理和物体不一样，物理定律不再适用于软件。结果导致人们很难明白电脑内部发生的事情。我们的头脑中存在很多解题陷阱，但是我们自己不知道。我们在观察事物的时候总是用头脑中已经有的概念系统或已经有的判断方式对事物进行解读，比如物理法则。

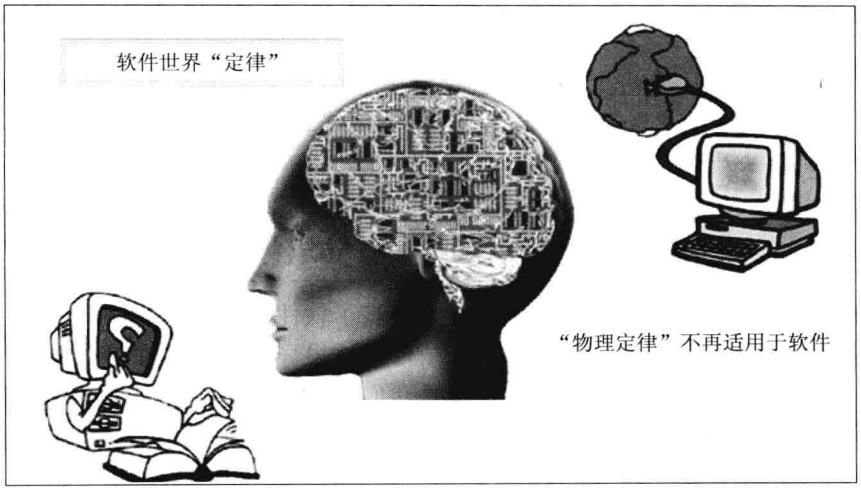


2．电脑的另类思维

在软件世界里，物理定律不再适用，编程要用另一套有别于以前经验的处理方式。

实际上，C 语言不仅仅是一种语言，也是一种进一步抽象的意识形态，通过它你可以进一步理解计算机的思维方式。学习编程，也就是要学会用电脑的方式看世界。

计算机能完成许多有趣和令人惊异的工作，它是由程序来控制的，本书将让你了解如何命令计算机去完成这些工作，带领你进入程序设计的世界。我们将要踏上的是一条充满挑战且回报丰厚的旅途，期望你能够在学习的过程中获得享受的乐趣！



3．程序设计课程的特点

(1) 思维另类。编程的思维方式与数学等需要逻辑推理的课程不太一样，入门有一个过程。

(2) 规则琐碎。要记忆的规则多，比较琐碎。

(3) 实践积累。重实践及经验积累，仅仅纸上谈兵的练习是远远不够的。

4．学习方法

(1) 把握关键。站在计算机的角度观察问题。编程要用另一套有别于以前经验的方式处理问题(软件法则)。

(2) 重复记忆。尽量通过不断重复练习来记忆、熟练规则。

(3) 多多上机。要下功夫，多上机练习。

5．课程主要内容

(1) 程序设计的基本概念与基本方法；

(2) 程序的基本结构、语句、数据类型；

(3) 数组：数据的组织方式之一，可解决一组同类型数据的存储运算问题；

(4) 函数：模块化，可解决程序规模足够大时产生的问题；

(5) 指针：逻辑指代与物理指代；

(6) 结构：数据的组织方式之二，可解决一组非同类型数据的存储运算问题；

(7) 文件：数据的组织方式之三，它是对数据的永久存储与重复使用；

(8) 程序的调试与测试的基本概念和方法。

6. C 语言的作用

每次在给新同学上课时，学生最常问的问题之一就是：“老师，您教的这门课有什么用？”

C 语言是用来编程序的，也就是做代码开发的，它在下面的领域有重要的用途：

(1) 单片机、电子、嵌入式行业。C 语言具有很强的功能性和结构性，同汇编语言开发相比，它可以缩短单片机控制系统的开发周期，而且易于调试和维护，已经成为目前单片机语言与嵌入式系统中最流行也是应用最广泛的编程语言，在将来很长一段时间内仍将在嵌入式系统应用领域占重要地位。

(2) 游戏开发。我们玩的 PC 游戏很多都是使用 C/C++语言编写的。

(3) 系统软件开发。C 语言允许直接访问物理地址，可以直接对硬件进行操作，因此既具有高级语言的特点，又具有低级语言的特性，能够像汇编语言一样对位、字节和地址进行操作，而这三者是计算机最基本的工作单元，可以用来编写系统软件。目前最著名、最有影响、应用最广泛的三个操作系统 Windows、Linux 和 UNIX 都是用 C 语言编写的，因此 C 语言适用范围大、可移植性好。

7. C 语言适用的机器

> 今天，事实上所有新的主流操作系统都是用 C 或 C++语言编写的。C 语言可以应用于多数计算机上。通过仔细设计，程序员可以编写出能够移植到大多数计算机上的 C 程序。
>
> ——(美)迪特尔(H. M. Deitel)(《C How to Progran》的作者)

8. C 与 C++的用武之地

曾经在教“数据结构”课程时，有学生对笔者说，当初上 C 语言课时没有好好学，原因是认为 C++比 C 更高级，所以上 C 语言课时，就在下面看 C++的书，结果是 C 没学好，C++也没学好。

对 C 与 C++的关系，C++ 之父 Bjarne Stroustrup 是这样描述的：“C++是 C 的一个直接后代，它几乎包含整个 C 即将其作为一个子集。C++支持 C 语言的编程风格。”C++是以 C 为基础的，先学 C 则比较容易入门。无论是 C 还是 C++，都是编程的工具而已，应该根据应用的需要选择采用哪个，没有哪个更高级的问题。

如果要做内核开发、嵌入式开发算法实现或编写驱动程序，C 是更合适的语言；如果要开发应用软件(一般有用户界面)，用 OO(Object-Oriented，面向对象)语言就会更加得心应手。

第 1 章　走马观花看编程

【主要内容】

- 程序的概念；
- 算法设计方法；
- 程序设计方法；
- 简单的 C 程序介绍。

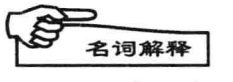
【学习目标】

- 理解程序设计的基本步骤；
- 能够用自顶向下、逐步求精的方法确定算法。

1.1　程序的概念

为了使计算机能够按人的意图工作，人类就必须要将需要解决问题的思路、方法和手段，通过计算机能够理解的形式告诉它，使得计算机能够根据人的指令一步一步去工作，完成特定的任务。

编程就是让计算机为解决某个问题而使用某种程序设计语言编写程序代码，并最终得到结果的过程。



➢ 程序：为了让计算机解决特定问题而专门设计的一系列计算机可执行的指令集合。

➢ 程序设计语言：即 Programming Language，是用于书写计算机程序的语言。

➢ C 语言：Combined Language(组合语言)的中英文混合简称，是一种计算机高级语言。

计算机语言

计算机语言的种类非常多，总的来说可以分成机器语言、汇编语言和高级语言三大类。目前通用的编程语言有两种：汇编语言和高级语言。

1．机器语言

由于计算机内部只能处理二进制代码，因此，用二进制代码 0 和 1 描述的指令称为机器指令。全部机器指令的集合构成计算机的机器语言。用机器语言编写的程序称为目标程序。只有目标程序才能被计算机直接识别和执行。但是用机器语言编写的程序无明显特征，难以记忆，不便阅读和书写，且依赖于具体机种，局限性很大。机器语言属于低级语言。

2．汇编语言

汇编语言的实质和机器语言是相同的，都是直接对硬件操作，只不过其指令采用了英文缩写的标识符，更容易识别和记忆。它同样需要编程者将每一步具体的操作用命令的形式写出来。

汇编源程序一般比较冗长、复杂、容易出错，而且使用汇编语言编程需要有更多的计算机专业知识。但汇编语言的优点也是显而易见的，用汇编语言所能完成的操作不是一般高级语言所能实现的，而且源程序经汇编生成的可执行文件不仅比较小，而且执行速度很快。

3．高级语言

高级语言是目前绝大多数编程者的选择。和汇编语言相比，它不但将许多相关的机器指令合成为单条指令，并且去掉了与具体操作有关但与完成工作无关的细节，这样就大大简化了程序中的指令，编程者也就不需要有太多的计算机硬件知识。

高级语言主要是相对于汇编语言而言的，它并不是特指某一种具体的语言，而是包括了很多编程语言，如目前流行的 VB、C++、Delphi 等，这些语言的语法、命令格式都各不相同。

用高级语言所编制的程序不能直接被计算机识别，必须经过转换才能被执行。按转换方式可将它们分为两类：

(1) 解释类：执行方式类似于我们日常生活中的“同声翻译”。应用程序源代码一边由相应语言的解释器“翻译”成目标代码(机器语言)，一边执行，因此效率比较低，而且不能生成可独立执行的可执行文件，应用程序不能脱离其解释器。但这种方式比较灵活，可以动态地调整、修改应用程序。

(2) 编译类：编译是指在应用源程序执行之前，就将程序源代码“翻译”成目标代码(机器语言)，因此其目标程序可以脱离其语言环境独立执行，使用比较方便，效率较高。但应用程序一旦需要修改，必须先修改源代码，再重新编译生成新的目标文件(*.OBJ)才能执行，如果只有目标文件而没有源代码，则修改很不方便。现在大多数的编程语言都是编译型的，例如 C/C++、Delphi 等。

软件与程序的关系

常常听到这样的说法，编程(Programming)就是编写软件。但程序与软件在概念上是有区别的。软件应该包含以下三个方面的含义：

(1) 运行时，能够提供所要求功能和性能的指令或计算机程序集合。

(2) 程序能够满意地处理信息的数据结构。

(3) 描述程序功能需求以及程序如何操作和使用所要求的文档。

所以可以认为：

软件=程序+数据+文档

1.2　计算机解题过程

从“加工”的角度看，计算机解题的过程可以分成三个步骤，如图 1.1 所示。先由系

统分析员根据实际问题建立数学模型，当这个模型不完全适合计算机时，还要将它转换成计算机能够“接受”的模型，做这个转换的工作需要有数据结构的知识。解题模型建立后，程序员根据它编制程序，再交由计算机执行，得到最终的结果。

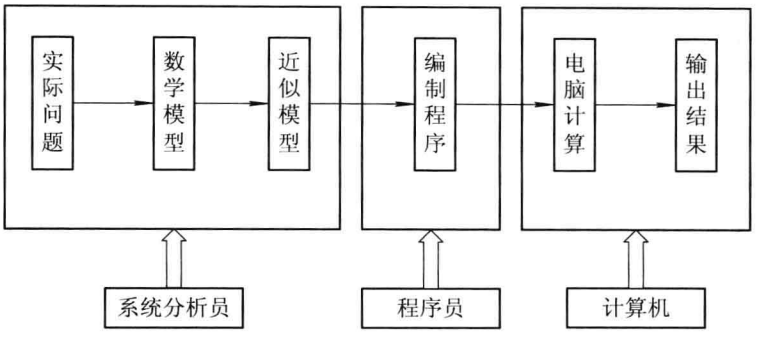


图 1.1　计算机解题过程

1.3　编制程序的全过程

编制程序有三个基本的步骤：定算法、编程序、调试通。

(1) 定算法——确定算法：先要明确实际问题要求完成的功能是什么，它要处理的信息有哪些(这些信息是要输入到计算机中的)；计算机对上述信息处理完毕，实现了指定的功能后，结果是什么，即输出的信息有哪些；计算机完成指定功能的具体步骤是什么。

(2) 编程序——代码实现：根据前面确定的算法，编写相应的程序代码。

(3) 调试通——调试通过：在 IDE 中，将编好的程序通过编译器翻译成机器码，这个过程叫做“编译”。

(注：IDE(Integrated Development Environment)即集成开发环境软件，是用于程序开发环境的应用程序，一般包括代码编辑器、编译器、调试器和图形用户界面工具。该程序可以独立运行，也可以和其他程序并用。)

在编译过程中，编译器可以查找程序中的语法错误，当没有语法错误后，生成后缀为 OBJ 的目标文件；若程序是由多个 OBJ 组合而成的，则还要把这多个 OBJ 文件进行链接，最后形成一个后缀为 EXE 的可执行文件，这时，就算是编写好了一个可以在计算机上运行(Run)的程序了。试着运行一下这个程序，计算机会严格按照用户编写的指令一步步执行下去。此时可以去看运行的结果，若结果和预设的一致，那就完成了任务。

但事情往往没那么简单，除非是很简单的程序，一个很有经验的程序员也不能保证程序首次运行就得到正确的结果。为什么会这样呢，编译完成，程序不是已经没有语法错误了吗？没有语法错是程序可以运行的前提，但程序中还会有一种叫做“逻辑错”的错误，它会造成运行结果的不正确。逻辑错是需要程序员自己去查找的。找逻辑错的工作叫做程序调试，这是一个很需要耐心和专注力的工作，难度往往比编程还要大。

调试的基本方法将在第 10 章“程序调试及测试”中专门介绍。