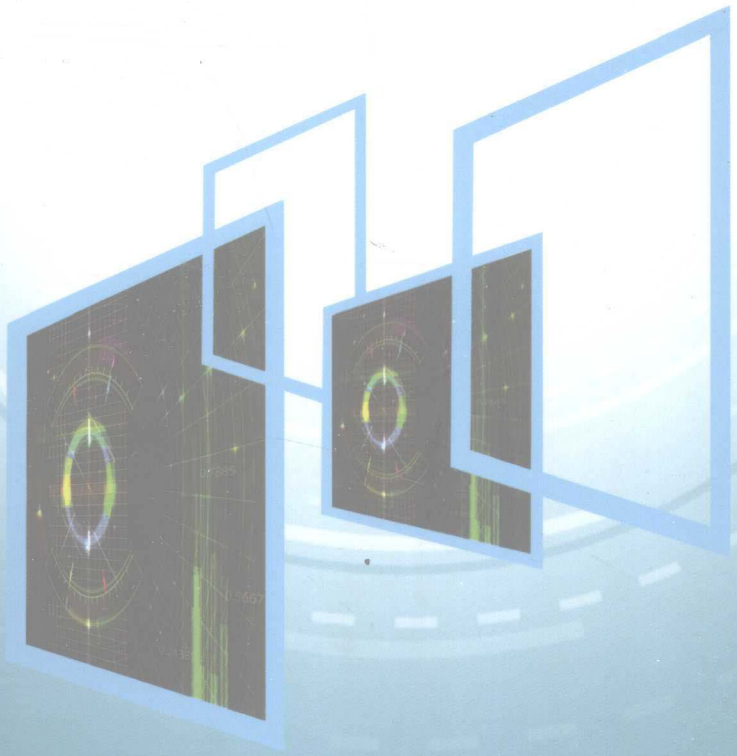
大学计算机基础课程“十二五”规划教材

C编程方法学

王舜燕　李　民　◎主编



武汉理工大学出版社
WUTP

础课程“十二五”规划教材

C 编程方法学

C Programming Methodology

主　编　王舜燕　李　民
副主编　钟　钰　吕　曦
　　李　宁　毛　薇

武汉理工大学出版社
·武汉·

内 容 提 要

C 语言是目前仍然广泛使用的计算机程序设计语言。它适用范围广、语法简洁、执行效率高，是典型的结构化程序语言，也是学习面向对象程序语言的基础。C 语言编程方法是普通本科院校的必修课程。

本书针对普通本科院校一、二年级学生的实际，从初学者角度介绍 C 语言编程方法。本书改变了学术论文式教科书的风格，通过诸多的图示和典型例题，深入浅出地阐述 C 编程方法学的内容。书中的主要知识点都辅之以英文表述，以期使读者更准确地理解相关内容。本书共有 8 章，涵盖 C 语言程序设计中的基本控制结构、数组和指针、模块化编程、指针进阶与内存空间管理、结构体和共用体、数据文件编程方法等内容。本书定位准确、内容适当、图例丰富、双语特色、叙述流畅、通俗易懂，凡具有初步计算机知识的读者都能读懂，是初学者掌握 C 语言编程方法的理想书籍。本书既适用于各类院校开设的 C 语言程序设计通识课程，也可供全国计算机等级考试二级 C 语言培训选用。

图书在版编目(CIP)数据

C 编程方法学/王舜燕，李民主编. —武汉：武汉理工大学出版社，2013.9
ISBN 978-7-5629-4150-7

Ⅰ.①C… Ⅱ.①王… ②李… 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(2013)第 200783 号

项目负责人：王兆国　　**责 任 编 辑**：王兆国
责 任 校 对：李兰英　　**装 帧 设 计**：语新文化
出 版 发 行：武汉理工大学出版社
社　　址：武汉市洪山区珞狮路 122 号
邮　　编：430070
网　　址：http://www.techbook.com.cn
经　　销：各地新华书店
印　　刷：荆州市鸿盛印务有限公司
开　　本：787×1092　1/16
印　　张：23.25
字　　数：589 千字
版　　次：2013 年 9 月第 1 版
印　　次：2013 年 9 月第 1 次印刷
定　　价：38.00 元

凡购本书，如有缺页、倒页、脱页等印装质量问题，请向出版社发行部调换。
本社购书热线电话：(027)87785758　87381631　87165708(传真)

前　言

C语言编程方法已经成为普通本科院校一年级学生的必修课程。如何在有限的课时内，有效地组织教学，成为授课教师面临的问题。如何在有限的课时内，有效地学好C语言，也成为学生们面临的问题。因此，探讨更有效率的教学方法，编撰更符合教学目的的教材，就成为急需解决的问题。本书就是为了满足这个层次的读者需求而编写的。

本书共有8章，从内容组织上分为3个部分。第1章、第2章是入门基础，结合Visual C++ 6.0的安装使用，介绍了C语言的基本语法。第3章是程序设计的基本结构，介绍了基本的结构化程序设计方法、简单算法的设计和表示方法，读者可以学会根据算法编制程序，设计出结构较为复杂的程序。第4章到第8章是模块化程序设计，读者通过模块化函数掌握程序模块的设计方法，并进一步通过数组、指针、结构体、位运算、数据文件等全面掌握C语言编程方法，培养分析问题和解决问题的能力。

本书作者都具有长期的C语言教学实践经验，也各自总结了许多的教案、教法，借此《C编程方法学》的编撰之际，大家集思广益，形成了本书的几大特色：

1. 改变学术论文式教科书的编制风格，以深入浅出的案例教学为线索，通过贴近实践、浅显易懂和生动有趣的实例，展开对C语言语句、语法的学习。

2. 强调编程实践，从编译环境的安装调试入手，要求读者通过上机实践，尤其是通过Debug调试程序的纠错实践，掌握C语言编程的基本技能。

3. 书中的每个例题给出了运行结果，并尽可能多地配置了各例题的程序流程图和示意图，使读者通过直观图示理解程序的编制逻辑，加深对C语言编程方法的了解。

4. 每章的本章小结中都给出了本章所涉及的语法总结表，和本章涉及编程方法中常见错误总结表，使得读者可以通过本章小结回顾和提炼所掌握的编程方法。

5. 采取贯穿始终的学习方式，介绍C语言中的指针问题。通过循序渐进的方法，通过各种应用实例，读者可逐步地建立指针、指针运算、指针数组、函数指针等一系列概念，并逐步地学习掌握相应的编程方法。

6. 鉴于C语言的原始发布是英语，本书对主要知识点和主要编程方法的阐述，都加入了英文的表述，以期更原始、更准确地表达这些内容。同时，也为目前实际上已经在许多大学开始的双语教学提供一本更合适的教学参考书。相信读者通过对照阅读本书中的各段英文表述，可以更清晰地了解到C语言编程方法的内涵。

本书由王舜燕、李民担任主编，钟钰、吕曦、李宁、毛薇担任副主编。第1章由王舜燕、吕曦编写，第2章由李宁编写，第3章由钟钰、毛薇编写，第4章由李民、孙骏编写，第5章由李岫、王舜燕编写，第6章由吕曦、段翠萍编写，第7章由秦珀石编写，第8章由汤练兵、李民编写。附录部分由李宁编写。全书由王舜燕、李民、钟钰和吕曦分章校定，王舜燕、李民统稿校定。在本书的大纲制订过程中，郑敬、陈志铭、张建宏、马德骏、李捷、杨朝阳等提出了不少建议，在此深表谢意。

由于作者水平有限，书中难免存在不妥与疏漏之处，敬请广大读者批评指正。

编　者

2013年6月于武汉理工大学

目　录

第1章
C语言概述

引言　为什么要学习C语言

虽然在C语言之后，C++、Java等各式各样的计算机高级语言层出不穷，但不少程序员仍旧认为，C语言简洁、高效、灵活的特性令其具有独特魅力。

与其他编程语言（像C++、Java）相比，C语言是个低级语言。而低级的编程语言可以让你更好地了解计算机的运行机理。

实现相同的功能，用C语言写的程序所用的代码行数更少，但程序的运行效率却更高。

学好了C语言，就能学好许多流行的编程语言，因为它们大多是以C语言为基础的（像Java，C++，C#等等）。

C语言已经有几十年的应用历史，有着大量的现成代码可以利用。这就使你能更加快速和高效地编写新的算法和函数。

C语言是唯一阐述指针本质的语言。虽然有争议，可是指针确实使C语言变得更加强大。

大多数的大学生都在学习C语言，因为它会给你提供更多的工作机会。如果你应聘编程开发方面的工作，C语言仍然是最普遍需要的语言。

让我们看看表1-1中独立评估机构TIOBE发布的2013年6月编程语言排行榜（本书撰稿时的最新数据），和C语言在TIOBE编程语言排行榜中的表现（表1-2）。C语言竟然不可思议地从1988年到2013年，在长达1/4个世纪的时间里，多次位居编程语言排行榜榜首，成为程序设计语言里真正的“不老松”。

以上种种理由还不值得你去花精力学习C语言吗？

表1-1　TIOBE发布的2013年6月编程语言排行榜

Position Jun 2013	Position Jun 2012	Programming Language	Ratings Jun 2013	Status
1	1	C	17.809%	A
2	2	Java	16.656%	A
3	4	Objective-C	10.356%	A
4	3	C++	8.819%	A
5	7	PHP	5.987%	A
6	5	C#	5.783%	A
7	6	(Visual) Basic	4.348%	A
8	8	Python	4.183%	A
9	9	Perl	2.273%	A

续表 1-1

Position Jun 2013	Position Jun 2012	Programming Language	Ratings Jun 2013	Status
10	11	JavaScript	1.654%	A
11	10	Ruby	1.479%	A
12	12	Visual Basic . NET	1.067%	A
13	17	Transact-SQL	0.913%	A
14	14	Lisp	0.879%	A
15	16	Pascal	0.779%	A
16	21	Bash	0.711%	A
17	19	PL/SQL	0.657%	A−−
18	13	Delphi/Object Pascal	0.602%	A−−
19	18	Ada	0.575%	B
20	22	MATLAB	0.563%	B

表 1-2 C语言在 TIOBE 编程语言排行榜中的表现

Programming Language	Position June 2013	Position June 2008	Position June 1998	Position June 1988
C	1	2	1	1
Java	2	1	3	—
Objective-C	3	42	—	—
C++	4	3	2	4
PHP	5	4	—	—
C#	6	8	—	—
(Visual) Basic	7	5	5	7
Python	8	7	30	—
Perl	9	6	7	—
JavaScript	10	9	17	—
Lisp	14	16	19	2
Ada	19	17	10	3

* TIOBE is specialized in assessing and tracking the quality of software.

The C programming language is a popular and widely used programming language for creating computer programs. Programmers around the world embrace C because it gives maximum control and efficiency to the programmer.

If you are a programmer, or if you are interested in becoming a programmer, there are a couple of benefits you gain from learning C:

It's ubiquitous, closer to the hard-ware, and used to create other languages and operating systems.

System programming (in pure C) or specialized areas when working with languages that are extensions of C or closely related.

Knowing C gets you closer to the hardware, to better understand how things work on the system level.

It's an important, foundational language that requires you to understand the full stack of the technology.

图 1-1 所示的语言抽象金字塔展示了计算机程序语言的发展演变过程和分层次结构。表 1-3 列出了在程序设计中常见符号的英文读法。

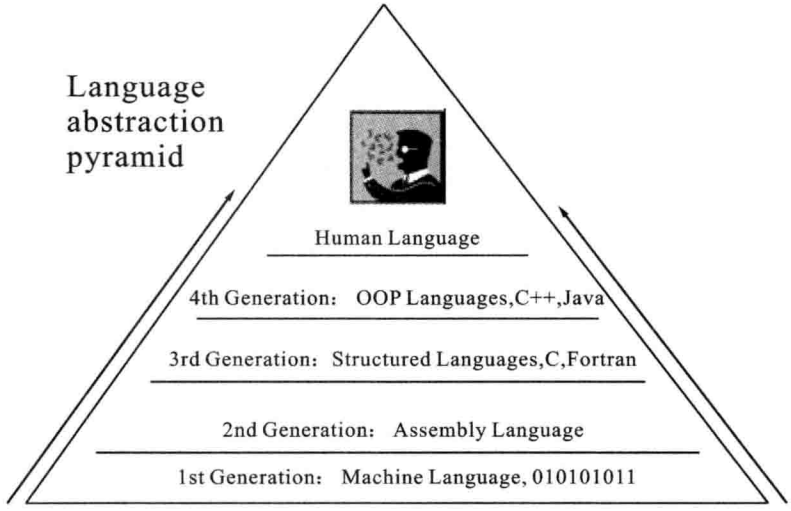


图 1-1　语言抽象金字塔

The strategy of learning C is a step-by-step approach, first laying a sound foundation on programming concepts, control statements, and functions.

The fundamental of C programming is a stepping stone that will prepare you to embark on the journey of learning other programming language.

表 1-3　C 语言中常见符号的英文读法

space	( )	semicolon	(;)
dot	(.)	and	(&)
question	(?)	or	(|)
assign	(=)	not	(!) exclamation
equals	(==)	quote	(')
plus	(+)	double quote	(")
minus	(-)	parenthesis	( )
times	(*) asterisk	bracket	[ ]
divided	(/) slash	brace	{ }
backslash	(\ )	enter	(↵)
underscore	(_)	sharp	(#) hash,pound
comma	(,)	tick	(√)
colon	(:)	cross	(×)

1.1 程序和算法

1.1.1 程序与结构化程序设计方法

程序(Program)实际上就是为解决特定问题,按既定算法编制的1个计算机指令集。1个高效实用的程序与算法的设计、对数据结构的描述、对程序设计方法的选择、语言工具以及环境条件关系密切。程序是外表,算法才是灵魂。可以这样给出程序和算法的关系:

程序 = 数据结构 + 算法

计算机必须通过明确的程序指令才能完成人们要它完成的工作,计算机无法完成人们想要它完成的、却没有明确指令的工作。

A program is a set of instructions for a machine to accomplish a specific task or solve a specific problem.

Program is an algorithm written in a language a computer can execute.

Computers do not understand the intention of a program. (Computers do what you tell them, not what you meant to tell them.)

Programs must be precise.

Programs must be detailed.

Programs must be debugged…

开发1个程序的一般步骤如下:

- 确定要解决的问题。
- 给出解决这个问题的算法。
- 用某种程序设计语言表述这个算法并形成程序文件。
- 对文件进行编译或解释以形成可执行的指令。
- 运行这个程序。
- 测试或修正这个程序。

Steps in Creation of a Program:

- Define the problem.
- Find an algorithm to solve the problem.
- Express the an algorithm in a computer language and store in one or more files.
- Compile the files to create a program. (Most languages)
- Run the program.
- Test and Correct the program.

E. W. Dijkstra
(1930—2002)

人们经过长期的实践和不断地总结经验,提出了结构化程序设计方法(Structured Programming)。结构化程序设计方法的概念最早由荷兰计算机科学家、1972年图灵奖获得者、University of Texas at Austin的迪克斯特拉(E. W. Dijkstra,1930—2002)在1965年提出,它的提出是软件发展的一个重要的里程碑。

结构化程序设计方法的主要观点是采用自顶向下、逐步求精的程序设计方法;将算法的描述归纳为3种基本结构的顺序组合,即顺序结构、选择结构、循环结构(3种基本结构对照见表1-4)。它们的共同特点是:只有1

个入口，只有 1 个出口，每个基本结构中的每一部分都有机会被执行，结构内部不存在死循环。使用 3 种基本结构描述的算法是结构化的算法，按照结构化算法编写出来的程序具有良好的可读性和可维护性。

结构化程序设计方法同时也存在着缺点。譬如，用户要求难以在系统分析阶段准确定义，致使系统在交付使用时产生许多问题；用系统开发每个阶段的成果来进行控制，不能适应事物变化的要求；系统的开发周期长等。

Structured programming is a programming paradigm aimed on improving the clarity, quality, and development time of a computer program by making extensive use of subroutines, block structures and for and while loops.

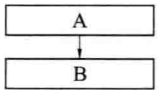
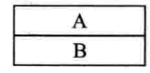
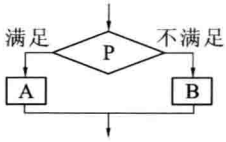
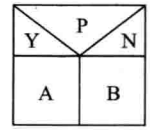
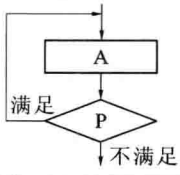
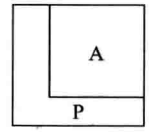
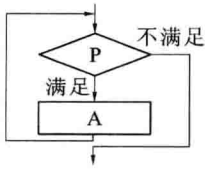
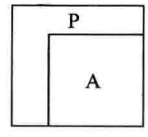
At a low level, structured programs are often composed of simple, hierarchical program flow structures. These are sequence, selection, and repetition:

"Sequence" refers to an ordered execution of statements.

In "selection" one of a number of statements is executed depending on the state of the program.

In "repetition" a statement is executed until the program reaches a certain state, or operations have been applied to every element of a collection.

表 1-4　3 种基本结构对照表

传统流程图	结构名称	N-S 流程图
	顺序结构	
	选择结构	
	直到型循环(后测试循环)	
	当型循环(前测试循环)	

为了适应结构化程序设计方法的要求，美国学者 I. Nassi 和 B. Shneiderman 在论文《Flowchart Techniques for Structured Programming》中提出了用 N-S 流程图(又称盒图，Box chart)描述算法的方法。在 N-S 流程图中，取消了流程线，全部算法由一些基本的描述 3 种基本结构的矩形框图按顺序排列组成 1 个大矩形而成。N-S 流程图特别适合描述 1 个结构化的算法，常用于结构化程序设计中。

I. Nassi
State University of New York

B. Shneiderman
University of Maryland

有关流程图在算法设计中的应用，本书会在以后章节逐步加以介绍。

1.1.2 算法的概念和特点

使用计算机解决实际问题的时候需要编写程序。程序是可连续执行并完成特定功能的一系列指令的集合，主要包含两个部分，即数据结构和算法。数据结构就是在程序中对要处理的对象(数据)进行描述，指定数据的类型和组织形式；算法就是对操作步骤的描述。

算法(Algorithm)是指对解题方案准确而完整的描述，是一系列解决问题的清晰指令。也就是说，能够对一定规范的输入，在有限时间内使编程者获得所要求的输出。不同的算法能用不同的时间、空间来完成同样的任务。1个算法的优劣可以用空间复杂度与时间复杂度来衡量。

An algorithm is a step by step process used to carry out some function.

使用计算机解决实际问题的过程通常包括以下几个步骤：

(1) 分析问题，找出解决问题的模型。

(2) 根据模型设计出适合计算机特点的处理方法(即算法)。

(3) 选择合适的计算机语言进行编程，以实现算法。

(4) 上机编辑、调试、运行所编制的程序，得到结果。

(5) 对结果进行分析，整理出文字材料(即文档)。

在实际操作过程中，上述步骤可能会反复出现。需要指出的是，算法的好坏直接影响到程序的编制、运行效率以及程序的可读性和可维护性。

算法就是为解决某一问题，对数据进行操作和处理所采取的一系列步骤。算法可分为两大类，一类为数值数据运算并求解的算法；另一类为非数值数据运算及其处理的算法。前者用来描述求数值数据解的过程，如求定积分、微分方程求解、高阶方程求解等；后者用来描述非数值数据求解的过程，如情报检索、事务处理、数据管理等。

注意：*在计算机应用方面，非数值数据运算的应用大大多于数值数据运算的应用。*

一个合理的算法具备以下特点：

(1) 有穷性(Finiteness)。1个算法应该包括有限个步骤，不能是无限个步骤。即经过有限个步骤的处理以后，算法应该结束。

因此，有始无终的解题步骤绝不是合理算法。例如：任何不指定求解精度求 π 的公式，都不可能构成合理算法。

(2) 确定性(Definiteness)。算法中的每1个步骤的含义都是确定的、唯一的，不能具有其他的含义或可能被理解成其他的含义。

例如："若 x 大于 0，则 y 等于 1 个正数"是 1 个运算规则，但不是 1 个合理算法。

(3) 有输入(Input)。1个算法可以有0个或多个输入，用以描述运算对象的初始情况。所谓0个输入是指算法本身定出了初始条件。

例如：如果给定坐标点(3,4)，那么计算该点距原点(0,0)的距离 $L=\sqrt{x^2+y^2}=5$ 就不需要输入 x 和 y 的值了。

(4) 有输出(Output)。1 个算法可有 1 个或多个输出，用以反映算法执行的结果，没有输出的算法是无效的算法。

需要注意的是，这里的输出不仅指屏幕显示或打印输出，包括磁盘文件存储、网络数据的发送，乃至向另 1 个运算模块的数据传输等，都是算法的输出形式。

(5) 有效性(Effectiveness)。算法中的每 1 个步骤都应当能有效地执行，并得到确定的结果。不应存在逻辑上无法执行的语句。

例如，x 的符号函数 sign(x)的定义是："如果 x 大于 0，则函数值等于 1；如果 x 等于 0，则函数值等于 0；如果 x 小于 0，则函数值等于－1。"也就是说：参数 x 的符号决定了 sign 函数值。但如果加上 1 条定义："否则，x 等于 100"，逻辑上是正确的，但这个定义规则除了在运算出错的情况下，是不会被运用到的。在算法中这就是 1 条在逻辑上无法被执行的语句，也称为死语句。

1.1.3　算法的表示

表示 1 个算法的方法很多，如自然语言法、传统流程图法、N-S 流程图法、伪代码法等，理论上都可用来表示算法，但是效率上有很大差异。

Algorithms are often expressed in pseudo-code.

There are no real standards for expressing algorithms in pseudo-code.

例如，M＝5!，求 M 的值。

自然语言法描述算法如下：

(1) 设定变量 M，M 置初值为 1，设定变量 i，置初值为 1。

(2) 如果 i 的值小于 6，则执行(3)，否则执行(4)。

(3) 将 M 乘以 i 并置于 M 中存放，将 i 中的值加 1 并置于 i 中存放，再执行(2)。

(4) 将 M 中的值输出。

用传统流程图法描述算法如图 1-2 所示；用 N-S 流程图描述算法如图 1-3 所示。

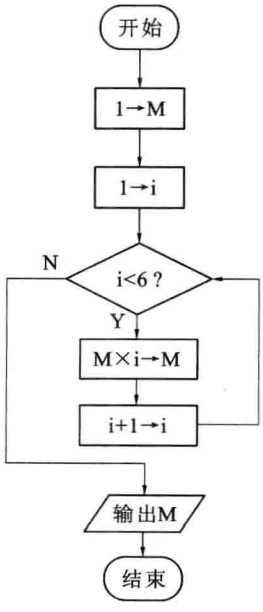


图 1-2　传统流程图

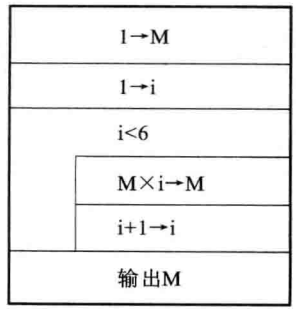
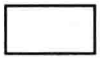
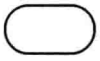
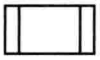


图 1-3　N-S 流程图

说明：传统流程图使用几何图形、流程线、文字说明(表 1-5)来描述 1 个算法，它的优点是直观、易懂，便于初学者掌握。但是，由于在使用传统流程图描述 1 个算法时，可以根据需要不受限制地使用流程线，这样设计出来的算法结构性不好，会导致阅读和编程困难，编制出来的程序结构性也不好，不利于以后的使用和维护。

表 1-5 传统流程图基本图形及其含义

图形	名称	说明
	处理框	表示确定的处理或步骤，又称矩形框
	判断框	允许有 1 个入口，2 个或 2 个以上的可选择的出口
	输入输出框	表示数据的输入或经处理后结果的输出
	起始结束框	表示算法的开始或结束
	连接点	用于将不同的流程线连接起来
	功能调用框	表示调用 1 个处理过程
	流程线	表示算法中处理流程的走向
	注释框	用于书写注释或说明信息

1.2 安装 Visual C++ 6.0

很多学完了 C 语言的人，却不会使用 IDE 工具(Integration Design Environment，集成开发环境)，甚至不会编程，其原因主要在于学习书本内容和实际的上机实践脱节，因此本书并不从语言入手，而从 1 个 IDE 的安装开始。

Visual C++ 6.0 是 Microsoft 公司推出的一款针对 C/C++语言开发的 IDE，虽然推出的时间比较长，但是对于学习 C 语言来说，是完全足够的，下面就开始安装 Visual C++ 6.0。

(1) 打开 VC++6.0 文件夹。

(2) 双击文件夹下的 SETUP.EXE。

(3) 如果在 Windows 7 上安装，将可能出现下面的弹出对话框，勾选“不再显示此消息”，然后点击“运行程序”，如果在 Windows XP 上安装，则不会出现这个对话框。