HZ BOOKS 华章教育
Pearson
经 典 原 版 书 库

程序设计基础

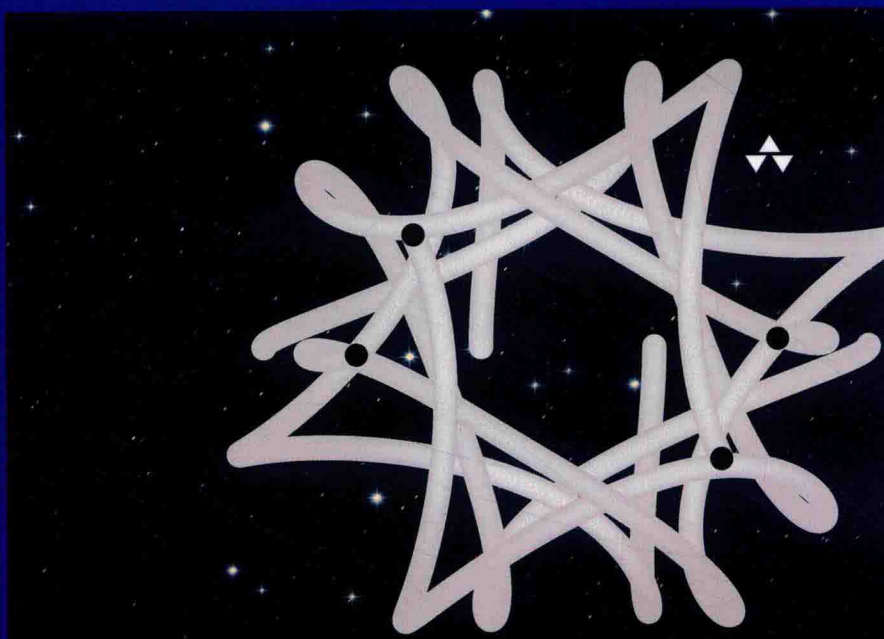
跨学科方法

[美] 罗伯特·塞奇威克（Robert Sedgewick） 凯文·韦恩（Kevin Wayne） 著
普林斯顿大学

（Java语言描述·英文版）

Computer Science
An Interdisciplinary Approach

机械工业出版社
China Machine Press



COMPUTER SCIENCE

An Interdisciplinary Approach

ROBERT SEDGEWICK
KEVIN WAYNE

经 典 原 版 书 库

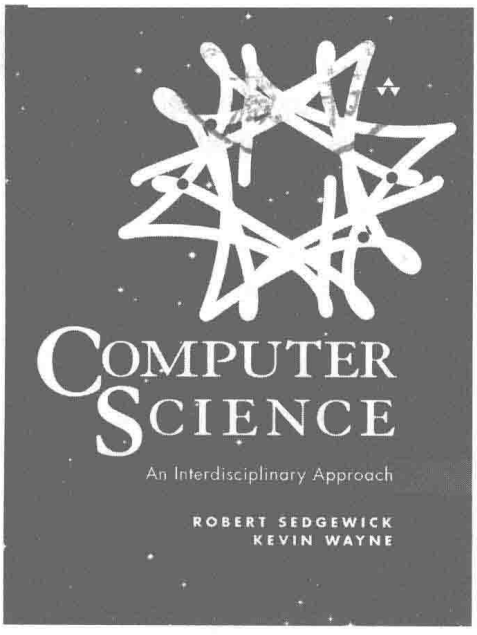
程序设计基础

跨学科方法

（Java语言描述·英文版）

Computer Science

An Interdisciplinary Approach



[美] 罗伯特·塞奇威克（Robert Sedgewick） 凯文·韦恩（Kevin Wayne） 著

普林斯顿大学

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

程序设计基础：跨学科方法（Java 语言描述 · 英文版）/（美）罗伯特 • 塞奇威克（Robert Sedgewick），（美）凯文 • 韦恩（Kevin Wayne）著 . —北京：机械工业出版社，2018.5
（经典原版书库）
书名原文：Computer Science: An Interdisciplinary Approach

ISBN 978-7-111-59908-1

Ⅰ. 程… Ⅱ. ①罗… ②凯… Ⅲ. JAVA 语言 - 程序设计 - 高等学校 - 教材 - 英文
Ⅳ. TP312.8

中国版本图书馆 CIP 数据核字（2018）第 091759 号

本书版权登记号：图字 01-2016-8662

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）

责任编辑：曲 熠	责任校对：殷 虹
印 刷：三河市宏图印务有限公司	版 次：2018 年 6 月第 1 版第 1 次印刷
开 本：186mm × 240mm 1/16	印 张：45.5
书 号：ISBN 978-7-111-59908-1	定 价：99.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：（010）88378991 88361066 投稿热线：（010）88379604

购书热线：（010）68326294 88379649 68995259 读者信箱：hzjsj@hzbook.com

出版者的话

文艺复兴以来，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的优势，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自 1998 年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与 Pearson，McGraw-Hill，Elsevier，MIT，John Wiley & Sons，Cengage 等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出 Andrew S. Tanenbaum，Bjarne Stroustrup，Brian W. Kernighan，Dennis Ritchie，Jim Gray，Afred V. Aho，John E. Hopcroft，Jeffrey D. Ullman，Abraham Silberschatz，William Stallings，Donald E. Knuth，John L. Hennessy，Larry L. Peterson 等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力相助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专门为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

华章网站：www.hzbook.com

电子邮件：hzjsj@hzbook.com

联系电话：（010）88379604

联系地址：北京市西城区百万庄南街 1 号

邮政编码：100037

华章教育

华章科技图书出版中心

前 言

20 世纪的教育基础是“阅读、写作和算术”，而现在是“阅读、写作和计算机科学”。学习编程是每个学生在科学和工程领域接受教育时的重要组成部分。除了直接应用外，这也是理解计算机科学的第一步，进而理解为什么计算机对现代世界影响巨大。本书的目的是在科学的应用环境中讲解编程相关知识。

本书的主要目标是介绍高效计算所必需的经验和基本工具，以此来增强学生的能力。我们的方法是教学生按照一种自然的、令人满意的、创造性的方式编写程序。书中逐步引入基本概念，并通过应用数学和科学中的经典实例来说明这些概念，同时让学生亲自动手编程来解决问题。我们设法帮助学生揭开计算的神秘面纱，建立对计算机科学领域的重要知识的基本认识。

本书中的所有程序都使用 Java 编程语言。我们首先教授解决计算问题的基本技能，这些技能适用于许多现代计算环境，并且是一种自成体系的解决方案，即使没有编程经验的人也能够学会。这里强调的是编程的基本概念，而不是 Java 本身。在后续章节中，我们会更多地偏重计算机科学知识而不再是编程，但是我们仍然经常使用 Java 程序来说明主要想法。

本书采用跨学科方法讲解传统的 CS1（computer science）课程，我们强调计算在其他学科中的作用，从材料科学到基因组学到天体物理学到网络系统。这种方法帮助学生将现代世界中数学、科学、工程和计算的基本思想融会贯通。虽然是一本为大一学生设计的 CS1 教科书，但也可用于自学。

内容与特色[⊖]

前三章围绕学习编程的三个阶段进行组织，包括基础知识、函数和面向对象编程。在每个阶段，我们都提供了学生需要熟练掌握的基本技术，为进入下一个阶段的学习做好准备。本书的一大特色是使用示例程序来解决有趣的问题，并辅以丰富的练习，既有自学练习，也有需要创造性解决方案的挑战性的问题。

第 1 章主要介绍变量、赋值语句、内置数据类型、控制流、数组和输入 / 输出（包括图形和声音等）。

第 2 章是学生首次接触模块化编程。我们假设学生已经熟悉数学中的函数，在此基础上引入 Java 函数，然后讨论函数编程的意义，包括函数库和递归。我们强调编程时要将

⊖ 本书共 7 章，篇幅超过 1000 页，为便于阅读，您手中的这本书只包含第 1 ~ 4 章基础内容，第 5 ~ 7 章进阶内容为在线资源，可访问华章网站 www.hzbook.com 免费下载。——编辑注

程序划分为可独立调试、可维护和可重用的组件，这是编程的基本思路。

第 3 章介绍数据抽象，重点是数据类型的概念，以及如何使用 Java 类机制来实现数据类型。我们教授学生如何使用、创建和设计数据类型。模块化、封装和其他现代编程范例是这个阶段的核心概念。

接下来的四章介绍计算机科学的高级主题：算法和数据结构、计算理论以及计算机体系结构。

第 4 章将现代编程范例与组织和处理数据的经典方法相结合，这些方法在现代应用中仍然有效。我们介绍了用于排序和搜索的经典算法、基本数据结构及其应用，并且强调了如何使用科学方法来理解某段代码的性能特征。

第 5 章通过简单的计算机抽象模型来解决计算的基本问题，这些知识不仅具有重要的理论意义，而且许多想法在实际计算中也非常重要，甚至可以直接应用。

第 6 章和第 7 章关注计算机器及其构建方法，学习体系结构可以帮助我们理解真实世界中的计算结果——在计算理论的抽象机器和我们使用的真实计算机之间建立联系。而且，这还有利于回顾历史，因为当今计算机和移动设备中使用的微处理器与 20 世纪中叶开发的第一台计算机并没有太大区别。

本书的核心特色是注重编程在科学和工程中的应用。我们通过分析编程对其特定应用领域产生的巨大影响来引导和激发学生学习相应的概念。我们以应用数学、物理和生物科学以及计算机科学本身为例，涉及的问题包括物理系统模拟、数值方法、数据可视化、声音合成、图像处理、财务模拟和信息技术等。具体的例子包括第 1 章中基于马尔可夫链的网页排名，以及后续章节中关于渗流问题、N 体模拟和小世界现象的案例分析。这些应用程序是本书的重要组成部分，它们不仅能够引导学习过程，说明编程概念的重要性，而且为计算在现代科学与工程领域所发挥的重要作用提供了有说服力的证据。

最后有一个短小的章节介绍了编程的发展历史。在各章中，我们还穿插介绍了图灵、冯·诺依曼等人关于计算基本思想的发展和应用的有趣故事。

我们的主要目标是让学生掌握有效解决编程问题所需的特定方法和技能。书中使用的代码都是完整的 Java 程序，读者可以试着运行起来。需要说明的是，本书主要面向编程初学者，并没有涉及大型编程问题。

教学建议

本书适用于科学应用类相关专业大一学生的计算机科学导论课程。在课堂上，学生可以在相对熟悉的专业背景下学习编程。通过本书和相关课程的学习，学生能够熟练地将编程技能应用到他们所选专业的后续课程中，并能够认识到进一步深入学习计算机科学是很有意义的。

对于计算机科学专业的学生，在科学应用的背景下学习编程也受益良多。为了更好地从事科研工作，计算机科学家需要具备关于科学方法的基本知识，也需要了解计算在科学

研究中的作用，并与生物学家、工程师或物理学家等传统领域的专家对于这类问题的认识保持一致。

事实上，我们的跨学科方法使得计算机科学专业和其他专业的学生可以在同一门课程中学习编程。书中涵盖 CS1 规定的所有内容，而且更注重通过具体应用引入编程概念，这更容易引起学生的学习兴趣。同时，跨学科方法让学生接触到来自不同学科的多种问题，能够帮助他们更明智地选择自己的专业。

无论使用哪种具体的教学方法，本书都适合在学生进入大学的初期使用，这样教师可以利用高中数学及其他科学的知识来开展教学。此外，在这一阶段学过编程的学生，之后进入专业课程时能够更有效地使用计算机。就像阅读和写作一样，编程对于任何科学家或工程师来说肯定都是必不可少的技能。掌握了书中相关概念的学生将会在他们的一生中不断提升这种技能，从而更好地理解其专业领域中遇到的问题和项目，并且更有效地利用计算机来解决它们。

前导课程

本书适合大一学生学习，也就是说，我们只需要科学和数学的入门级基础知识，不需要其他额外的知识作为前导。

熟练掌握数学工具和知识是学习本书的重要前提。虽然我们不讨论数学的细节内容，但是书中确实涉及高中数学知识，包括代数学、几何学和三角学。我们认为本书的目标读者大多已经掌握了这些内容。事实上，书中很多编程概念是建立在初等数学课程基础上的，对此学生通常比较熟悉。

好奇心也是一个重要的组成部分。理工科学生天然就带着对科学世界的迷恋，喜欢探究自然界发生的一切。为此，我们会通过简单的程序来揭示自然世界的规律。除了高中所讲授的数学、物理、生物或化学的基础知识以外，理解这些程序不需要任何特定的知识。

我们不要求学生具备编程经验，当然，有一定编程基础也不错。讲授编程是我们的主要目标之一，所以我们假设学生都没有编程经验。这是一门入门级的编程课程，但是那些在高中时编写过大量程序的学生也会有所收获，因为本书中涉及的问题都是跨学科的新问题，而编程解决新问题往往是一项具有挑战性的任务。本书可以用于教授具有不同背景的学生，因为这些应用程序对新手和高手都有吸引力。

理论上，本书也不要求学生具备使用计算机的经验，不过，现在的学生都经常使用计算机，例如，与亲友聊天、听音乐、处理照片等。通过学习本书，他们将意识到自己能够以更加有趣而且更加重要的方式利用计算机，这会是一个激动人心同时旷日持久的过程。

总之，几乎所有大学生都可以在第一学期的课程中学习本书的内容。

教学目标

对于那些完成了本书课程的学生，当他们开始学习理工科的高年级课程时，授课教师

会有什么期待呢?

本书主要面向计算机基础课程，但是任何教过编程基础课程的人都知道，在以后的课程中，教师的期望通常很高：每位老师都希望所有学生熟悉计算机环境和课程中所要使用的方法。物理学教授可能期望学生在周末设计一个程序来进行模拟，工科教授可能期望学生使用一个特定的包来实现微分方程的数值求解，计算机科学教授可能期望学生了解特定编程环境的细节。一门入门级课程要符合如此多样化的期望，这现实吗？应该为不同方向的学生开设不同的入门课程吗?

自 20 世纪末计算机广泛使用以来，高等院校一直受困于这些问题。我们利用本书回答了这些问题。这是一本介绍通用编程方法的教材，适用于数学、物理、生物和化学等学科的入门课程。计算机科学致力于为所有理工科学生提供所需的基础知识，同时传递出一个清晰的信号——计算机科学远不止程序设计这么简单。完成了本书的学习后，我们相信学生将会具备必要的知识和经验，能够适应新的计算环境，并在不同的应用程序中有效地利用计算机。

对于学生而言，学完了本书对应的课程后，可以在后续学习中尝试什么新的课程呢?

我们想说的是，编程并不难学，而且学会有效使用计算机是非常有意义的。如果能够掌握本书中的知识，那么在未来职业生涯中遇到各类计算机方面的挑战时，学生都有能力应对。虽然本书只涉及 Java 一种现代编程环境，但通过举一反三，学生将很快打开其他计算世界的大门，并有足够的信心学习、评估和使用新的计算工具。计算机科学专业的学生将会具备深入学习的基础知识，其他理工科专业的学生将会掌握将计算融入其他研究中的能力。

教学视频

本书配有一套完整的教学视频，网址如下：

http://www.informit.com/title/9780134493831

与传统的现场讲座一样，这些视频的目的是激发学生阅读本书的兴趣，并提升学习效果。我们的经验是这种方式比现场讲课要好得多，因为学生可以以选定的速度播放视频，并随时回放和复习。

本书网站

在下面的网站上，可以找到本书的大量补充信息和其他相关材料：

http://introcs.cs.princeton.edu/java

网站上提供了面向授课教师、学生和普通读者的材料。这里对这些材料进行简要介绍，要想了解详细内容，读者可访问网站浏览并查看。除了一些用于考试的材料外，其他材料都是公开的。

建立本书网站最重要的意义之一就是帮助教师和学生使用自己的计算机来讲授和学习

本书。任何拥有电脑和浏览器的人都可以按照网站上列出的方法开始学习编程。这个过程不会比下载媒体播放器或歌曲更困难。和其他网站一样，本书网站也在不断更新。对于拥有本书的人来说，这是必不可少的资源。特别需要说明的是，网站上提供了更多的补充材料，这对于本书知识的学习是至关重要的，能够促进计算机科学成为所有科学家和工程师所接受的基础教育的一个重要组成部分。

对于教师来说，该网站包含有关教学的信息。在过去十年中，我们每周组织两次大规模的授课，并辅以每周两次的讨论，学生以小组为单位与老师或助教进行交流。在教学过程中，我们形成了一套自己的教学风格，并按照这套风格组织了教学材料。在本书网站中可以下载教学幻灯片。

对于助教来说，这个网站包含了详细的习题集和编程任务，这些都是与书中的练习相对应的，而且加入了更多细节。每个编程任务都会在有趣的应用环境中介绍一个相关概念，同时向学生提出一个充满吸引力又有些难度的问题。任务难度的递进体现了我们的教学方法。本书网站上详细说明了所有作业，并提供了详细的、条理清楚的辅助材料，以帮助学生在规定的时间内完成作业。辅助材料包括建议的解决方法，以及在讨论课上讲授的内容大纲。

对于学生来说，该网站可以快速访问本书中的大部分内容，包括源代码以及一些建议自学的课外材料。本书网站上也提供了许多习题答案，包括完整的程序代码和测试数据。网站上还有大量与编程任务相关的信息，包括建议的方法、清单、常见问题解答和测试数据等。

对于普通读者来说，本书网站可以用于访问与书籍内容相关的所有额外资源。网站上的所有内容都提供链接和其他信息渠道，以供读者获取关于该主题的更多信息。网站上的内容非常丰富，远远超过读者的需要，但我们的目标是提供足够多的信息，确保满足每位读者关于本书内容的兴趣。

致谢

写作本书这个项目自 1992 年启动以来一直处在不断的发展当中，到目前为止，有太多人为之做出了贡献，我们对此表示由衷的感谢。特别感谢 Anne Rogers 一直以来的帮助；感谢 Dave Hanson、Andrew Appel 和 Chris van Wyk 耐心地解释数据抽象；感谢 Lisa Worthington 和 Donna Gabai，他们第一次尝试向大一学生讲授这本教材，这是一个巨大的挑战；感谢 Doug Clark 的耐心，帮助我们完善了图灵机和电路方面的知识。我们也非常感谢 / dev / 126 的努力；感谢普林斯顿大学 25 年来一直致力于讲授这本教材的教师、研究生和教学人员，以及成千上万名努力学习本书的学生。

Robert Sedgewick
Kevin Wayne
2016 年 5 月

Contents

Online Content ⊖

⊖ 请访问华章网站 www.hzbook.com 下载在线内容。

目 录

在线内容[⊖]

⊖ 请访问华章网站 www.hzbook.com 下载在线内容。

Elements of Programming

OUR GOAL IN THIS CHAPTER IS to convince you that writing a program is easier than writing a piece of text, such as a paragraph or essay. Writing prose is difficult: we spend many years in school to learn how to do it. By contrast, just a few building blocks suffice to enable us to write programs that can help solve all sorts of fascinating, but otherwise unapproachable, problems. In this chapter, we take you through these building blocks, get you started on programming in Java, and study a variety of interesting programs. You will be able to express yourself (by writing programs) within just a few weeks. Like the ability to write prose, the ability to program is a lifetime skill that you can continually refine well into the future.

In this book, you will learn the *Java programming language*. This task will be much easier for you than, for example, learning a foreign language. Indeed, programming languages are characterized by only a few dozen vocabulary words and rules of grammar. Much of the material that we cover in this book could be expressed in the Python or C++ languages, or any of several other modern programming languages. We describe everything specifically in Java so that you can get started creating and running programs right away. On the one hand, we will focus on learning to program, as opposed to learning details about Java. On the other hand, part of the challenge of programming is knowing which details are relevant in a given situation. Java is widely used, so learning to program in this language will enable you to write programs on many computers (your own, for example). Also, learning to program in Java will make it easy for you to learn other languages, including lower-level languages such as C and specialized languages such as Matlab.

1.1 Your First Program

In this section, our plan is to lead you into the world of Java programming by taking you through the basic steps required to get a simple program running. The *Java platform* (hereafter abbreviated *Java*) is a collection of applications, not unlike many of the other applications that you are accustomed to using (such as your word processor, email program, and web browser). As with any application, you need to be sure that Java is properly installed on your computer. It comes preloaded on many computers, or you can download it easily. You also need a text editor and a terminal application. Your first task is to find the instructions for installing such a Java programming environment on *your* computer by visiting


```
http://introcs.cs.princeton.edu/java
```

We refer to this site as the *booksite*. It contains an extensive amount of supplementary information about the material in this book for your reference and use while programming.

Programming in Java To introduce you to developing Java programs, we break the process down into three steps. To program in Java, you need to:

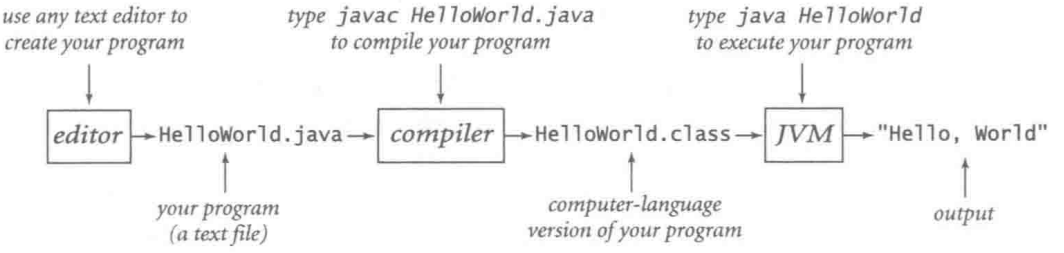
- *Create* a program by typing it into a file named, say, `MyProgram.java`.
- *Compile* it by typing `javac MyProgram.java` in a terminal window.
- *Execute* (or *run*) it by typing `java MyProgram` in the terminal window.

In the first step, you start with a blank screen and end with a sequence of typed characters on the screen, just as when you compose an email message or an essay. Programmers use the term *code* to refer to program text and the term *coding* to refer to the act of creating and editing the code. In the second step, you use a system application that *compiles* your program (translates it into a form more suitable for the computer) and puts the result in a file named `MyProgram.class`. In the third step, you transfer control of the computer from the system to your program (which returns control back to the system when finished). Many systems have several different ways to create, compile, and execute programs. We choose the sequence given here because it is the simplest to describe and use for small programs.

Creating a program. A Java program is nothing more than a sequence of characters, like a paragraph or a poem, stored in a file with a `.java` extension. To create one, therefore, you need simply define that sequence of characters, in the same way as you do for email or any other computer application. You can use any *text editor* for this task, or you can use one of the more sophisticated *integrated development environments* described on the booksite. Such environments are overkill for the sorts of programs we consider in this book, but they are not difficult to use, have many useful features, and are widely used by professionals.

Compiling a program. At first, it might seem that Java is designed to be best understood by the computer. To the contrary, the language is designed to be best understood by the programmer—that's you. The computer's language is far more primitive than Java. A *compiler* is an application that translates a program from the Java language to a language more suitable for execution on the computer. The compiler takes a file with a `.java` extension as input (your program) and produces a file with the same name but with a `.class` extension (the computer-language version). To use your Java compiler, type in a terminal window the `javac` command followed by the file name of the program you want to compile.

Executing (running) a program. Once you compile the program, you can execute (or run) it. This is the exciting part, where your program takes control of your computer (within the constraints of what Java allows). It is perhaps more accurate to say that your computer follows your instructions. It is even more accurate to say that a part of Java known as the *Java Virtual Machine* (*JVM*, for short) directs your computer to follow your instructions. To use the JVM to execute your program, type the `java` command followed by the program name in a terminal window.



Developing a Java program

Program 1.1.1 Hello, World

```
public class HelloWorld
{
   public static void main(String[] args)
   {
      // Prints "Hello, World" in the terminal window.
      System.out.println("Hello, World");
   }
}
```

This code is a Java program that accomplishes a simple task. It is traditionally a beginner's first program. The box below shows what happens when you compile and execute the program. The terminal application gives a command prompt (% in this book) and executes the commands that you type (`javac` and then `java` in the example below). Our convention is to highlight in boldface the text that you type and display the results in regular face. In this case, the result is that the program prints the message `Hello, World` in the terminal window.

```
% javac HelloWorld.java
% java HelloWorld
Hello, World
```

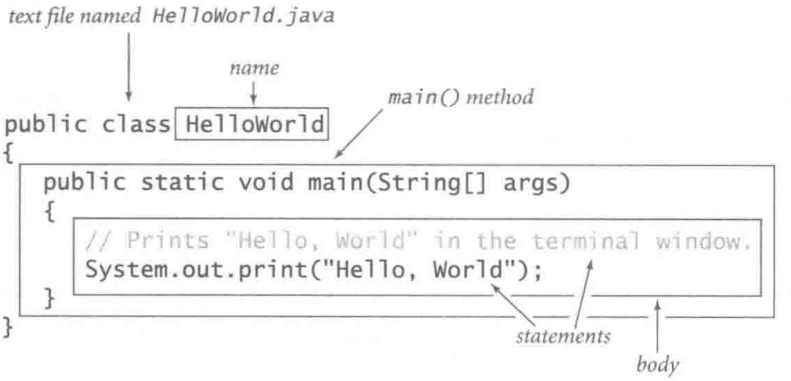
PROGRAM 1.1.1 is an example of a complete Java program. Its name is `HelloWorld`, which means that its code resides in a file named `HelloWorld.java` (by convention in Java). The program's sole action is to print a message to the terminal window. For continuity, we will use some standard Java terms to describe the program, but we will not define them until later in the book: PROGRAM 1.1.1 consists of a single *class* named `HelloWorld` that has a single *method* named `main()`. (When referring to a method in the text, we use () after the name to distinguish it from other kinds of names.) Until SECTION 2.1, all of our classes will have this same structure. For the time being, you can think of "class" as meaning "program."

The first line of a method specifies its name and other information; the rest is a sequence of *statements* enclosed in curly braces, with each statement typically followed by a semicolon. For the time being, you can think of "programming" as meaning "specifying a class name and a sequence of statements for its `main()` method," with the heart of the program consisting of the sequence of statements in the `main()` method (its *body*). PROGRAM 1.1.1 contains two such statements:

- The first statement is a *comment*, which serves to document the program. In Java a single-line comment begins with two '/' characters and extends to the end of the line. In this book, we display comments in gray. Java ignores comments—they are present only for human readers of the program.
- The second statement is a *print statement*. It calls the method named `System.out.println()` to print a text message—the one specified between the matching double quotes—to the terminal window.

In the next two sections, you will learn about many different kinds of statements that you can use to make programs. For the moment, we will use only comments and print statements, like the ones in `HelloWorld`.

When you type `java` followed by a class name in your terminal window, the system calls the `main()` method that you defined in that class, and executes its statements in order, one by one. Thus, typing `java HelloWorld` causes the system to call the `main()` method in PROGRAM 1.1.1 and execute its two statements. The first statement is a comment, which Java ignores. The second statement prints the specified message to the terminal window.



Anatomy of a program