抽象类（abstract class）：抽象类不能创建对象，主要用来创建子类。Java 中的抽象类使

用 abstract 修饰符定义。

抽象数据类型（abstract data type ADT）：抽象数据类型指明了可能的类型和允许进行

的操作，但是没有提供实现。

访问标识符（access specifier）：用于方法或变量定义，限定了哪些类可以访问该方法或

变量。Java 中的访问标识符有 public、protected 和 private。没有访问标识符修饰的方

法或变量默认可见性为“package”。

活动记录（activation record）：活动记录是包含了实现子程序调用必须的所有信息，包

括参数值、子程序中的本地变量和子程序调用结束时计算机的返回地址。活动记录存储在栈

中，使得多个子程序调用同时活跃成为可能。这对递归非常重要，递归时对同一个子程序的

多个调用要求同时激活。

实参（actual parameter）：子程序的参数叫做声明。当调用执行时，通过声明把值传递

给自函数。实参也叫做“参数”。

地址（address）：计算机内存中的每个位置都有一个地址，表示该位置的编号。内存中的

位置按序号排列。在现代计算机中，内存中的每个字节都有自己的地址。在存储或读取内存

信息时用需要用到地址。

算法（algorithm）：完成某项任务所需要的一步一步的过程，过程本身没有歧义，且可以

保证在有限的步骤内完成该任务。

阿尔法颜色组成（alpha color component）：颜色组成用来描述颜色的透明度或不透明

度。阿尔法组成越高，颜色越不透明。

API：应用编程接口。针对软件包或“工具箱”的接口说明。API 包含了工具箱中所有类或

子程序及其使用说明。

Applet：与可以单独运行的应用程序不同，Applet 是一种在 Web 浏览器中运行在 Web

页面上的 Java 程序。

动画（animation）：由一组静态图片快速显示展示出动态效果。每一幅静态图片叫做帧。

在 Java 中，动画通常由 Timer 对象驱动。每次定时器触发时，会显示动画的下一帧。

反锯齿（antialiasing）：当图形和文本以像素方式显示时，可以通过调整像素的颜色减轻

“锯齿”效应。反锯齿画图时，图形只覆盖像素的一部分，图形的颜色与该像素之前的颜色

混合而成。混合的程度由覆盖像素的多少决定。

数组（array）：一个顺序排列的元素列表。列表中，每个元素都可以由自己的索引标识，

即序号。在 Java 中，数组里所有元素必须类型相同，该类型也称作数组的基类型。数组是

一种可随机访问的数据结构，也就是说，你可以随时直接访问数组中的任意元素。

数组类型（array type）：这种数据类型的值是数组。比如类型的名字为 Type，那么 Type[]

就是数组类型，基类型为 Type。

赋值语句（assignment statement）：计算机程序中的一种语句，可以读取或计算数值，

并将其存储到变量中。Java 中的赋值语句形式为：变量名 = 表达式。

异步事件（asynchronous event）：异步事件指发生时间不可预料的事件，计算机程序

无法对其控制。像点击鼠标、按键这样的用户输入事件都是异步的。

ASCII 码：美国信息交换标准码。这种编码使用 7 个比特对字符编码。ASCII 码只支持 128

个字符，不支持重音字符、非英语字符、特殊符号或非字符化语言的表意符号，比如中文。

Java 采用了容量更大、更加完整的 Unicode 编码处理字符。

基线条件（base case）：在递归算法中，基线条件可以直接处理不需要继续递归。

二进制数（binary number）：数值被编码为一组 0、1 序列。一般数字以“10 为基数”，

二进制数与其类似，只是以“2 为基数”。

二叉树（binary tree）：二叉树是一种链式数据结构。可以为空树，或者由两棵更小的二

叉树（可能为空树）与根节点组成。根节点包含了指向两棵子树的指针。这两棵更小的二叉

树被称作左子树和右子树。

比特（bit）：一位二进制数，可能是 0 或 1。

黑盒（black box）：指系统或组件在使用时无需关心内部结构。黑盒包括接口和实现。在

系统中，被当做组件使用的黑盒叫做模块。

块（block）：在 Java 编程中，被花括号（{}）包围的一组语句称为块。（代码）块用来将

一组语句组合成一条语句。块可以为空，表示不包含任何语句，即一对空的花括号。

阻塞操作（blocking operation）：一个操作如果需要等待某些事件发生就称为“阻塞”

操作，比如从网络连接读取数据。执行阻塞操作的线程会一直处在“阻塞”状态，直到事件

发生。处于阻塞状态时，线程不能执行任何指令。而程序中的其它线程可以继续执行。

阻塞队列（blocking queue）：当阻塞队列为空时，出队操作会引发阻塞，直到队列中有

新成员加入。如果阻塞队列有大小限制，当队列填满时，入队操作也会引起阻塞。

自底向上设计（bottom-up design）：自底向上设计是一种软件设计方法。从系统的基

础组件开始设计，然后将它们组合成更复杂的组件，诸如此类。

BufferedImage 类：BufferedImage 类展示了“屏外画布”，即图片存储在计算机内存

中，可以在屏幕外进行绘制。

分支（branch）：分支是一种控制结构，计算机通过分支从 2 个或多个不同的执行路径中

进行选择。Java 有两种分支语句：if 语句和 switch 语句。

字节（byte）：字节是一种由 8 个比特组成的内存单元。一个字节可以保存 8 个比特二进

制数。

字节码（bytecode）：“Java 字节码”是 Java 虚拟机机器语言的常用名称。Java 程序会

被编译成 Java 字节码，后者由 JVM 执行。

字符集（charset）：字符集是一种将字符数据编码为二进制的特定编码形式。例如 UTF-8

和 ISO-8859-1。

受检异常（checked exception）：在 Java 中受检异常必须处理，可以通过 try catch 语

句捕获，或者在方法上使用 throw 语句抛出该异常。如果没有用这两种方式处理受检异常，

会报告语法错误。

类（class）：类是 Java 的基础编程单元。类是静态方法、非静态方法和变量的集合。静态

成员是类自身的一部分，非静态或“实例”成员是创建对象的蓝本，由此创建的对象“属于”

该类。

类变量（class variable）和类方法（class methods）：“静态变量”和“静态方法”的

别名。它们是类的一部分，与对象无关。

客户端/服务器（client/server）：一种网络通讯模式。其中，“服务器”在网络上守候某

个已知地址，等待“客户端”向它发起连接请求。这是 TCP/IP 协议的基础通讯模型。

命令行接口（command-line interface）：一种计算机交互方法。用户向计算机输入命令，

计算机对每个命令进行响应。

注释（comment）：在一个计算机程序中，注释是那些被计算机忽略的文本。注释的目的

是方便人们阅读，帮助理解程序。

编译器（compiler）：编译器是一种计算机程序，将某种计算机语言（通常是高级语言）

编写的程序翻译成机器语言程序。

组件（component）：组件是对 GUI 可视元素的泛称，包括窗口、按钮或菜单等。在 Java

中，组件表现为 java.awt.Component 子类创建的对象。

构造函数（constructor）：类的一种特殊子程序，主要用来创建类的对象。构造函数一般

使用 new 操作符进行调用，通常不被看做“方法”。

容器（container）：类似 JPanel 这样的组件，容器可以包含其它 GUI 组件。调用容器

的 add() 方法可以向其添加组件。

方法契约（contract of a method）：方法接口的语义组件。它指明了方法及其调用者的

职责，如何调用该方法，以及正确调用方法时会执行的任务。方法契约应当在该方法的

Javadoc 注释中完整说明。

控制结构（control structure）：类似 if 语句、while 循环这样可影响程序控制流（即

程序中指令执行顺序）的程序结构。

CPU：中央处理器。CPU 是计算机中实际执行计算和运行程序的部分。

数据结构（data structure）：经过组织的数据集合。在程序中被当做一个单元处理。

死锁（deadlock）：一种多个线程无限等待的情况。出现死锁的原因，比如每个线程都在

等待其它线程锁定的资源。

默认方法（default method）：Java 8 接口中的方法，该方法提供了自己的实现。所有

实现带有默认方法的接口都可以使用默认实现，但是不能覆盖默认方法。通过 default 保

留字标记默认方法。Java 7不支持默认方法。

默认包（default package）：默认包没有包名。没有在带有名字的包中声明的类都归属

默认包。

明确赋值（definite assignment）：在程序中，变量在使用前必须确保已经被赋值。局部

变量只有在赋值后才能合法使用。为了达到这个要求，编译器必须对变量从声明开始到使用

的每条路径都进行赋值检查。

弃用（deprecated）：表示已经废弃，但为了先后兼容仍然保留。弃用的 Java 类或方法

仍然是 Java 语言的一部分，但不建议在新代码中使用。在未来的 Java 版本中，弃用的内

容会被移除。

对话框（dialog box）：对话框是依赖其它窗体创建的新窗体。弹出对话框通常用作获取

用户信息或展示消息。Swing API中，对话框表示为 JDialog 创建的对象。

分布式计算（distributed computing）：一种在由网络连接的多个计算机中进行的并行

处理。

虚参数（dummy parameter）：调用子程序时，用来代替实际传入参数的标识符。虚参

数也叫“形式参数”（有时候会用“变元 argument”表示实参，这时虚参数也叫做“参

数”）。

enum：枚举类型。枚举类型的定义中列举了该类型所有可能值。在 Java 中，枚举类型是

一个类，所有可能的值都是对象。

事件（event）：在 GUI 编程中，事件指发生在程序控制以外的操作，比如点击鼠标。程

序必须对发生的事件进行响应。

异常（exception）：程序控制流程之外的错误或异常情况。在 Java 中，异常表示为

Throwable 对象，可以由 try..catch 语句捕捉并处理。

获取-执行周期（fetch-and-execute cycle）：也称指令周期。指 CPU 执行机器语言程序

的过程。CPU 会从内存获取（即读取）指令，执行（运行）指令，然后再循环重复该过程。

标志（flag）：一个布尔值。设为 true 时表示达到某些条件或发生了某种事情。可利用二

进制数中的某个比特位作为标志。

形式参数（formal parameter）：“虚拟参数”的另一种说法。

帧（frame）：组成动画的某一幅画面，也是活动记录的另一种说法。

函数（function）：带有返回值的子程序。

垃圾回收（garbage collection）：自动回收内存的过程。被回收的内存由对象占用但已

不再会对其访问。

泛型编程（generic programming）：编写的代码不仅限于单一数据类型，可适应多种数

据类型。Java 集合框架及其它使用了相似技术的类都是泛型编程的实例。

getter 方法：类中的一个实例方法，用来读取类的某个属性值。通常，属性代表一些实例

变量的值。按惯例，getter 方法被命名为 getXyz()，其中 xyz 是属性的名字。

全局变量（global variable）：成员变量的别名。强调类中的成员变量可以在类方法外存

在。

图形上下文（graphics context）：用来绘制某些特定地点所必须得数据和方法。Java 中

的图形上下文是属于 Graphics 类的对象。

GUI：图形用户界面是与计算机的现代交互方式。计算机通过 GUI 在显示器上展示类似按

钮和菜单这样的接口组件，用户可以通过像鼠标点击这样的方式与之交互。

哈希表（hash table）：一种优化的数据结构，可以高效搜索、插入和删除对象。哈希表

包含对象的地址数组。对象存储的地址由自身的“哈希代码”决定。通过对象的内容可以高

效地计算出地址整数值。

堆（heap）：计算机内存中存储对象的区域。

高级语言（high level language）：类似 Java 这样的计算机语言，方便人们阅读，但在

执行前需要翻译成机器语言。

HSB：一种颜色系统。其中颜色由 3 个数值表示（在 Java 中，实际的数值在 0.0 到 1.0 之

间）。分别代表色调、饱和度和亮度。

IDE：集成开发环境。带图形用户界面的编程环境，集成了创建、编辑和执行程序的各种工

具。

标识符（identifier）：在程序中可用作名字的一组标识符。标识符可用作变量名、方法名

和类名。

索引号（index）：元素在数组中的位置编号。

实现（implementation）：黑盒的内部实现，比如子程序的实现代码。

不可变对象（immutable object）：不可变对象构造完成后不能改变，因为实例中所有变

量都标记为 final。

无限循环（infinite loop）：循环永远不会结束，因为它的循环条件永远判定为 true。

继承（inheritence）：一个类可以继承另一个类。继承者会从父类继承数据和行为。

类的实例（instance of a clas）：s 指归属于类（或者该类型子类）的对象。当类用作对

象模板时，对象由类中的构造函数创建的对象归属于这个类。

实例方法（instance method）：类中的非静态方法，该类的所有实例都具有该方法。

实例变量（instance variable）：类中的非静态变量，该类的所有实例都包含该变量。

接口（interface）：对如何使用类似子程序这样的黑盒子一种通用说法。接口对其内部发

生的情况没有提供任何信息。“interface”同时也是 Java 中的保留字。从这个意义上说，

接口是一种定义了一个或多个抽象方法的类型。实现该接口的对象必须提供这些方法的定义。

解释器（interpreter）：一种执行程序的计算机程序，被执行的程序由某种编程语言编写。

通过从程序中一个接一个读取指令然后逐条执行（将指令翻译为等价的机器语言）。

I/O：输入/输出。计算机程序与其它部分的通讯方式，比如向用户展示数据、从用户那里

获取信息、读写文件、通过网络发送和获取数据。

迭代器（iterator）：与 list 或 set 这样的集合相关联的对象。可用来对该集合进行遍历。

迭代器会轮流访问集合中的每个元素。

Java 集合框架（Java Collection Framework JCF）：一组实现了泛型数据结构的标准

类。包括 ArrayList、TreeSet 等。

JavaFX：新的应用程序 GUI 工具集。在 Java 8中推荐使用。JavaFX 不在本书的讨论范围。

JDK：Java 开发工具集。支持编译、运行 Java 程序的基本软件。JDK 包含命令行编程环境

以及 要编译 源代码或执行预编译程序时 需要使用

以及 JRE。要编译 Java 源代码或执行预编译程序时，需要使用 JDK。

JRE（Java Runtime Environment）：Java 运行时环境。支持运行已编译的标准 Java

程序。JRE 包括一个 Java 虚拟机和所有标准的 Java 类。

即时编译器（just-in-time compiler）：一种解释器和编译器的结合，在解释程序某部分

的同时可对其进行编译。接下来对该部分程序执行时比首次运行更快速。这样可以大大提高

执行速度。现代 JVM 都使用了即时编译器。

JVM：Java 虚拟机。将 Java 字节码作为机器语言执行的虚拟计算机。也用来称呼解析字节

码程序的计算机程序。要在计算机上运行 Java 程序需要使用 JVM。

布局管理器（layout manager）：负责对容器中组件进行布局的对象。进行的部分操作包

括设置大小和位置。不同类型的布局管理器实现的布局策略各不相同。

链式数据结构（linked data structure）：一组由之指针相互链接的对象数据。这些指针

存储在对象的实例变量中。链式数据结构包括链表和二叉树。

链表（linked list）：一种链式数据结构，节点之间由指针串连形成线性链表。

监听器（listener）：在 GUI 编程中，可以向对象注册特定事件的触发通知。因此可以说，

对象在“监听”这些事件。

文字（literal）：在程序中键入的一组字符序列，表示常量值。例如，当 A 在 Java 程序中

出现时，’A'是一个常量字符。

（内存）位置（location in memory）：计算机内存由一系列位置组成。这些位置顺序

编号，标识特定位置的编号被称为该位置的地址。

本地变量（local variable）：在方法内部声明的变量，只能在该方法内部使用。代码块中

声明变量的有效性，从声明处开始到该代码块的尾部结束。

循环（loop）：一种控制结构，重复执行一组指令。Java 提供了 3 种循环：for 循环、while

循环和 do 循环。

循环控制变量（loop control variable）：for 循环中的变量，每次执行 for 循环时都会

修改循环变量值，通过检查该变量决定是否结束循环。

机器语言（machine language）：由计算机能够直接执行的指令组成的编程语言。机器

语言中的指令会被编码成二进制数。每种类型的计算机都有自己的机器语言。用其它语言编

写的程序必须翻译为该计算的机器语言，才能在它上面执行。

主内存（main memory）：程序和数据可以存储在计算机的主内存中，主内存可以被 CPU

直接访问。其它形式的内存，比如磁盘驱动器，虽然也能存储信息，但是唯有主内存可被

CPU 直接访问。磁盘分区中的程序和数据只有拷贝到内存中才能被 CPU 访问。

map：一种映射数组。这种数据结构将一组（Collection）中的某个对象与摸个集合（Set）

中的所有对象关联在一起。在 Java 中，map 由泛型接口 Map<T,S> 表示。

成员变量（member variable）：定义在类中的变量，但不属于任何方法。成员变量与本

地变量不同，后者在某个方法中定义。

内存（memory）：计算机中的内存用来存储程序和数据。

方法（method）：子程序的另一种称呼，在面向对象编程中使用。方法指包含在类或对象

中的子程序。

模块（module）：大型系统中的组件，与系统中其它部分以简单、定义清晰、直接的方式

进行交互。

多任务（multitasking）：一次执行多个编程任务。要么在多个任务之间快速来回切换，

要么同时逐个执行多个任务。

多重处理（multiprocessing）：进行多任务处理时使用多个处理器。这样，多个任务可

以同时逐个执行。

互斥（mutual exclusion）：防止两个线程同时访问相同的资源。在 Java 中，这种方法

应用于多个线程同时访问同步方法或同步语句中的资源。互斥可以阻止竞态条件，但是可能

引发死锁。

MVC 模式（MVC pattern）：模型-视图-控制器模式。一种用在 GUI 组件中进行职责划

分的策略。模型代表组件的数据，视图指该模型在屏幕上的展示，控制器负责响应模型变化

事件。在 MVC 模式中，这些职责由不同的对象负责处理。

非数值（NaN）：不是一个数值。

Double.NaN 表示一种特殊的 double 值，表示未定义

或非法值。

节点（node）：链式数据结构中，某个对象的常用称呼。

null：一种特殊的指针值，表示“没有指向任何东西”。

数值分析（numerical analysis）：使用逼近法研究算法的领域，比如实数以及从逼近计

算中得到的错误。

差一错误（off-by-one error）：一种常见错误，处理时多减或多加了一个元素。通常是

技术错误或者循环时由其它原因过早停止或过度执行造成的。

对象（object）：计算机程序中带有数据（变量）和行为（方法）的实体。Java 中的对象

必须以某个类作为创建模板。对象所属的类决定了对象包含的类和方法。

对象类型（object type）：这种类型的值是对象而非基础类型。类和接口都是对象类型。

OOP（Object-Oriented Programming）：面向对象编程。一种计算机程序设计和实现

的方法。OOP 使用类和对象创建、表示实体及实体间的交互。

操作系统（operating system）：在计算机中一直运行的基础软件。没有操作系统的计算

机无法工作。操作系统由 Linux、Mac OS 和 Windows Vista。

操作符（operator）：类似“+”、“<=”或”++”这样的操作符，可以在表达式中计

算一个或多个值。

（操作符）重载（overloading of operators）：相同操作符可以在不同类型的数据上使

用。比如“+”操作可以同时应用于数字和字符类型。

（方法名）重载（ l di f th d ） 同 个类中定义了几个名称相同

（方法名）重载（overloading of method name）：s 同一个类中定义了几个名称相同

的方法，区别在于各个方法的签名不同。

重写（覆盖）（overriding）：在子类中进行重定义。子类中，对从父类继承的方法重新

定义，新定义的方法就是对原方法进行重写。

包（package）：在 Java 中，相关类和子包的有名集合称为包，例如 java.awt 和

javax.swing。

并行处理（parallel processing）：同时执行多个任务，可以是多个处理器，也可以由一

个处理器在多个任务间返复执行。

参数（parameter）：调用子程序时，参数用来向子程序提供信息。在执行子程序代码前，

子程序调用语句中的“实参”会分配给子程序定义的“虚参数”。

参数化类型（parameterized type）：类似 ArrayList<String> 这样，包含了一种或多

种类型参数的类型（这里的参数类型是 String）。

解析（parsing）：确定预演中字符串语法结构的过程。解析字符串用来确定字符串中是否

遵循该语言的语法；如果是，那么会确定该字符串是如何根据语法进行创建。

部分完全数组（partially full array）：数组是用来存储数量各异的元素。部分完全数组表

示为一个带有追踪元素个数计数器的普通数组。

像素（pixel）：指屏幕或图片中的“图像元素”。一幅图像由像素的行和列组成。每个像

素的色彩都可以单独设置。

多态（polymorphism）：多态是指调用实例方法的意义取决于调用方法时对象的实际类

型。也就是说，如果变量的类型是 var，那么调用方法的语句，比如 var.action() 取决于

执行时 var 所指向的对象类型，而非 var 变量的类型。

指针（pointer）：代表计算机内存中某个地址的值，因此可以看做“指向”具有该地址的

位置。在 Java 中，变量不存有对象；变量只是指向存储该对象的位置。指针也称作“引用”。

语用学（pragmatics）：描述如何编写好程序的经验法则。例如样式规则、程序组织指南

都是编程语用学的一部分。

优先级（precedence）：操作符的优先级指，在没有括号的情况下，表达式中多个操作符

的作用顺序。

前置条件（precondition）：在程序的执行过程中，为了让程序正确运行，前置条件必须

判定为 true。子程序的前置条件是指，为了让子程序正确运行必须满足的前置条件。子程

序的前置条件通常是对传入的子程序的实参值进行的限制。

优先级队列（priority queue）：一种表示元素结合的数据结构，其中每个元素都有自己

的“优先级”。优先级队列具有添加和移除操作。可以按照任意的顺序添加元素，但移除时

总是先移除优先级最低的元素。（某些版本的优先级队列会先移除优先级最高的元素）

后置条件（postcondition）：在执行程序的某个节点，该条件的计算结果为 true。子程

序的后置条件在子程序执行结束后必须为 true。函数的后置条件通常表示为函数的返回值。

基本类型（primitive type）：Java 的 8 个内建基本数据类型有 double、float、long、

int、short、byte、boolean 和 char。基本类型变量存储了真实的值，而非指向数值的指

针。

线程优先级(priority of a thread：) 与线程关联的整数值，可以影响线程的执行顺序。优

先级高的线程比优先级低的线程提前执行。

生产者/消费者（producer/consumer）：并行编程中的一种经典模式，一个或多个生产

者生产的产品被一个或更多的消费者使用。生产者和消费者设计为可以并行执行。这里的难

点在于，如何安全、高效地从生产者向消费者非配产品。在 Java 中，通过阻塞队列实现生

产者/消费者模式。

程序（program）：用某种合适的编程语言编写的一组指令，由计算机执行。用做动词时，

表示创建该指令的动作。

编程语言（programming language）：用来为计算机编程的一种语言。编程语言的复杂

性，从机器语言到像 Java 这样的高级语言跨度很大。

协议（protocol）：在指定上下文中，构成合法通信的一组规范。协议中规定了合法的消

息、传送的时间、期待的恢复类型等。

伪代码（pseudocode）：非正式算法规范。与实际编程语言相比，伪代码更加接近英语。

并且，通常无需明确地写出过程的每个细节。

队列（queue）：由一组元素构成的数据结构。只能在列表的一头添加数据，在列表的另

一头移除数据。

竞态条件（race condition）：并行编程中可能的错误源。由于某个线程改变了第二个程

序依赖的程序状态（比如变量值），从而引发错误。

RAM（Random Access Memory）：随机访问内存。计算机主内存的同义词。然而，从

技术角度看，RAM 是指在任意时间内都可以同样访问内存地址。RAM 也意味着可以同时

读写数据。

递归（recursion）：用自身的形式定义自己。特别地，递归子程序可以直接或通过一系列

其它子程序间接调用自己。递归算法的工作方式，通过将一个复杂问题拆分成更小的子问题。

子问题要么可以直接解决，要么可以“递归”使用相同的算法。

RGB：一种颜色系统。颜色由 3 个数值定义（在 Java 中，数值的范围从 0 到 255）。分别

表示颜色中的红色、绿色和蓝色组成。

引用（reference）：“指针”的另一种称呼。

函数返回类型（return type of a functio）n ：函数返回值的类型。

保留字（关键字）（reserved word）：一组字符串序列，看起来和标识符很像。但是由

于在编程语言中有特殊含义，所以不能用作标识符。例如 class、public 是 Java 中的保留

字。

资源（resource）：图片、声音、文字或程序中的数据文件都是资源。Java 资源文件与编

译好的 class 文件存储在相同的路径（class path）。

健壮的程序（robust program）：健壮的程序不仅正确，同时也能合理地处理错误，比如

文件不存在或者无法建立网络连接。

集合（set）：没有重复的一组对象。在 Java 中，集合用泛型接口 Set<T> 表示。

作用域（ ） 表示程序中声明的标识符的有效作用域

作用域（scope）：表示程序中声明的标识符的有效作用域。

语义（semantics）：语义是指，语言的语义规则决定了该语言中字符串符号（例如句子或

语句）的含义。

哨兵值（sentinel value）：表示数据序列结束的特殊标记，表明该数据已结束。

setter 方法：类中的实例方法，用来设置类中的一些属性值。通常，这些属性只是实例变

量值。按照惯例，setter 方法命名为 setXyz()，其中 xyz 是属性的名字。

方法签名（signature of a method）：方法名、方法定义中形参的个数以及每个形参的

类型。编译器通过方法签名来识别指定子程序调用语句究竟调用了哪个方法。

套接字（socket）：网络上两个计算机之间的连接，对其中一端的抽象表示。套接字表示

计算机程序间的一个逻辑连接，而非计算机间的物理连接。

栈（stack）：一组元素组成的列表数据结构，只能在列表的尾部，也称作“栈顶”。向栈

中添加元素叫“入栈 push”，移除元素称作“出栈 pop”。栈也指用来实现子程序调用

的活动记录。

标准输入（standard input）：程序读取输入数据的标准源，由 System.in 对象表示。通

常，标准输入来自用户键入的文本，但是标准输出也可以重定向到其它源，比如文件。

标准输出（standard output）：程序将输出文本写入的标准目标，由 System.out 对象

表示。通常，标准输出会向用户展示，但是也可以“重定向”到其它目标，比如文件。此外，

System.err 对象用来输出错误信息。

状态机（state machine）：一种计算机模型。抽象的“机器”可处于有限状态集中的任何

一种状态。机器会根据自身的状态决定行为，根据输入或事件进行状态切换。GUI 程序的基

本逻辑可表现为状态机。

逐步精化法（step-wise refinement）：一种算法开发技术。开始列出算法步骤大纲，通

常用伪代码表示，然后逐步填充细节。

流（stream）：输入数据源或输出数据目标的抽象表示。Java 为字符和二进制数据输入、

输出流提供了 4 个基础类。这些类组成了 Java I/O API的基础。

源代码（source code）：用高级编程语言编写的文本。在计算机执行前，必须翻译成机器

语言，比如 Java 字节码。

子类（subclass）：一个类，直接或间接继承了其它类，从而继承了它的数据和行为。前

一个类就叫做第二类的子类。

子程序（subroutine）：一系列程序指令。通过子程序名可以“调用”该子程序。在面向

对象编程中，子程序也称作方法。

子程序调用语句（subroutine call statement）：程序中调用子程序的语句。当执行子程

序调用语句时，计算机会执行子程序内部的代码。

super：一个特殊变量，在实例方法中自动定义。super 指包含该方法的对象，归属于方法

所在类的父类。super 可以访问父类中被隐藏的成员变量，这些变量与子类中的成员变量同

名。

语法（syntax）：定义了该语言中哪些字符串符号符合该语言的语法规则。

TCP/IP：Internet 上用来网络通讯的协议。

this：一个特殊变量，在实例方法中自动定义，归属于方法所在的类。

线程（thread）：对一系列逐条执行语句的抽象表示。计算机可以并行执行多个线程。

线程池（thread pool）：执行任务时可以使用的一组“工作线程”。当任务就绪时，可分

配给池中的线程。线程池通常使用阻塞式队列存储任务。

自顶向下设计（top-down design）：一种软件设计方法。开始把问题作为一个整体，然

后拆分成更小的问题，再逐步细化，直到子问题可以直接解决。

类型（type）：数值的特定类型。例如，int 类型规定了整形值可以表示为 32 比特二进制

数。在 Java 中，类型可以是基本类型、类名或接口名。类型名可用来定义变量的类型，子

程序中的虚拟参数和子程序的返回类型。

类型转换（type cast）：把一种类型的值强制转换为另一种类型。例如，在

(int)(6\*Math.random()) 中，(int) 是一个类型转换操作，将 (6\*Math.random()) 的浮点

值转换为了整形，丢弃了实数中的小数部分。

Unicode：将字符编码成二进制数的一种方式。Unicode 字符集包含了许多语言的字符，

不仅限于英语。Java 内部使用的就是 Unicode 字符集。

URL：全球资源定位器。Internet 上资源的地址，比如网页。

变量（variable）：一个带名字的内存位置（或一系列位置），用来存储数据。在程序中创

建变量，通过变量声明语句将名字赋予该变量。根据使用情况，这个名字在程序中可以代指

该内存位置，或者向内存地址中存储数据。在 Java 中，变量具有类型，可以表示变量持有

数据的种类。

包装类（wrapper class）：类似 Double、Integer 这样的类，将一个基本类型的值“包