Mc Graw Hill Education

学习指导与习题解答

Schaum's Outline of Data Structures with Java

Second Edition

数据结构

学习指导与习题解答（Java语言版）

（第2版）

John R. Hubbard 著

孙燕 陈伊文 等 译

清华大学出版社

Mc Graw Hill Education

学习指导与习题解答

Schaum's Outline of Data Structures with Java

Second Edition

数据结构

学习指导与习题解答（Java语言版）

（第2版）

John R. Hubbard 著

孙燕 陈伊文 等 译

清华大学出版社

北京

John R. Hubbard
Schaum's Outline of Data Structures with Java, Second Edition
EISBN: 978-0-07-147698-0

北京市版权局著作权合同登记号 图字:01-2009-5147 号

图书在版编目(CIP)数据

数据结构学习指导与习题解答:Java 语言版:第 2 版/(美)哈伯德(Hubbard,J. R.)著;孙燕,陈伊文等译. —北京:清华大学出版社,2012.4
书名原文:Schaum's Outline of Data Structures with Java, Second Edition
ISBN 978-7-302-27463-6

Ⅰ. ①数… Ⅱ. ①哈… ②孙… ③陈… Ⅲ. ①数据结构—高等学校—教学参考资料 ②JAVA 语言—程序设计—高等学校—教学参考资料 Ⅳ. ①TP311.12 ②TP312

中国版本图书馆 CIP 数据核字(2011)第 249227 号

责任编辑:龙啟铭
封面设计:常雪影
责任校对:李建庄
责任印制:何 芊

出版发行:清华大学出版社
网 址:http://www.tup.com.cn, http://www.wqbook.com
地 址:北京清华大学学研大厦 A 座 邮 编:100084
社 总 机:010-62770175 邮 购:010-62786544
投稿与读者服务:010-62776969,c-service@tup.tsinghua.edu.cn
质 量 反 馈:010-62772015,zhiliang@tup.tsinghua.edu.cn
印 刷 者:北京密云胶印厂
装 订 者:北京市密云县京文制本装订厂
经 销:全国新华书店
开 本:185mm×230mm **印 张**:26 **字 数**:568 千字
版 次:2012 年 4 月第 1 版 **印 次**:2012 年 4 月第 1 次印刷
印 数:1~3000
定 价:49.00 元

产品编号:034568-01

译者序

数据结构是计算机专业的重要专业基础课，对培养学生的软件素质，提高学生的软件开发能力与软件项目管理能力具有重要的意义。

本书系统地介绍了数据结构的基础知识，内容实用而丰富，主要内容如下：

- 第 1 章概括面向对象编程。
- 第 2 章介绍数组结构。数组具有高效性，应用非常广泛。
- 第 3 章说明链接的数据结构，这种结构可以快速插入和删除有序列表。
- 第 4 章讲解 Java 汇集框架，它提供统一框架来实现常见数据结构，使得生成的类可以以一种一致有效和直观的方式使用。
- 第 5 章介绍栈，这是实现后进先出协议的集合。
- 第 6 章说明队列，这是实现先进先出协议的集合。
- 第 7 章讲解的线性表是可顺序存取的容器。
- 第 8 章的内容是哈希表，它是容器型的数据结构，允许通过任意索引类型直接访问元素。
- 第 9 章介绍递归，利用此技术，可针对复杂问题提供优美而简单的解。
- 第 10 章说明树，即层次组织结构的非线性数据结构。树结构常用来存储数据，这种组织结构能提供对数据的有效访问。
- 第 11 章介绍二叉树。这是一种重要的数据结构。
- 第 12 章介绍了搜索树。搜索树是对所存储的数据进行了某种排序的树结构。

本书特点是，详细说明重要知识点，用丰富的例子，让读者全面透彻地理解和掌握概念。每章最后都有大量习题，覆盖面广，难易适中，并给出详尽答案，使读者能举一反三，全面透彻地理解概念，灵活运用理论知识解决实际问题，并检验知识的掌握程度。

本书由孙燕和陈伊文翻译，参加本书翻译工作的人员还有李志云、李晓春、陈安华、侯佳宜、许伟、戴文雅、于樊鹏、刘朋、王嘉佳、邓卫、邓凡平、李波、程云建、许晓哲、朱珂、韦笑、孙宏、李腾、陈磊、魏宇、周京平、徐冬、冯哲、李绯、李强、赵东辉、王雷、龚亚萍等人。在此表示感谢。

前言

像其他学习指导教材一样，本书主要供自学的读者使用，它适合于用作采用 Java 编程语言的数据结构课程的学习指南。在美国大学里，数据结构课程通常是计算机科学专业的第二门课程。本书也是数据结构和 Java 汇集框架方面的一本不错的参考书。

此书包含 200 多个详细示例，为 260 多个问题提供了解决方案。作者确信，按照一些带有完整说明的结构完备的示例进行练习是最好的编程学习方式，本书就是为了提供这种支持而写的。

与原来的 2001 版相比，现在的第 2 版有了很大的改进。大多数章节进行了全面的改写，而且添加了全新的 3 章，这 3 章介绍的是面向对象编程、链接结构和 Java 汇集框架。

本书通篇使用的是 Java 6.0，尤其关注了该语言的以下新特性：

- Scanner 类；
- StringBuilder 类；
- 格式化输出，包括 printf()方法；
- 增强的 for 循环(也称为 for-each 循环)；
- 静态导入；
- enum 类型；
- 可变长度参数列表；
- 自动装箱；
- 泛型类；
- Java 汇集框架中的 Deque、ArrayDeque、EnumSet 和 EnumMap 类以及 Queue 接口。

可以从作者的以下网站下载所有示例、已解决问题和补充编程问题的源代码：

http://www.mathcs.richmond.edu/~hubbard/books/

我要感谢帮助过我审阅本书原稿的所有朋友、同事、学生和 McGraw-Hill 的有关人员，还有 Stephan Chipilov 和 Sheena Walker。特别要感谢我的同事 Anita Huray Hubbard，谢谢她的建议、鼓励以及为本书提供的一些创造性题目。

第 1 章　面向对象编程

1.1　软件设计和开发

成功的计算机软件是经过一系列阶段后产生的，这些阶段由不同开发团队控制，图 1.1 给出了这些阶段。

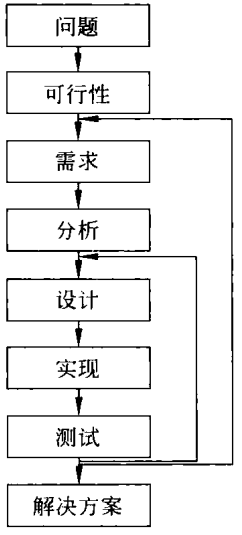

图 1.1　软件生命周期

第一个阶段是识别要解决的问题。对于某个公司来说，这一决策可以来自市场调查。

第二个阶段可以省略为一个正式流程，该阶段是分析项目的可行性的。例如，是否存在有关开发工具可以用来生成该软件？

在第三个阶段中，通常会生成一份文档，来明确指定该软件应该执行哪些内容。该需求文档应该足够详细，以用作进行完整软件测试时的一种标准。

第四个阶段是在为设计和实现项目做出任何努力或策划资源之前，进行彻底的分析，这可以包括调查已经可用的相关软件以及对所用预期资源价值的成本效益分析。决定继续开发后，软件设计团队就会根据需求文档来设计软件。这包括详述所有软件组件及其相互关系，还可能需要具体说明软件中应该实现的专门算法。

在实现阶段中，编程人员根据设计编写相应的代码来生成软件。

测试团队试图确保生成的软件符合需求文档，如果这时出现失败，将需要重新设计，甚至需要对需求进行一些调整，在图 1.1 中，这些可能性是通过两个反馈循环表示的。

测试是按不同的级别进行的。各个类和方法需要单独测试，然后需要验证它们是否能够在一起成功运行，最后，需要根据需求文档测试整个产品。

该图中没有显示软件开发的一个最后方面：维护过程。软件发布后，其开发人员负责维护它，包括提供修正的版本、服务包，甚至包括重要的修订。任何重大修订都应该经历同样的生命周期步骤。

1.2 面向对象设计

一种常见的软件设计方法是自顶向下设计策略,该策略将问题逐步分解成更小的部分,这也称为逐步细化。它关注问题的功能方面和算法的实现,这种面向过程的设计在科学计算编程中很常见。

与此相比,面向对象设计关注的是软件的数据组件,围绕这些组件的表现进行设计。例如,一个空中交通控制系统可以根据飞机、降落场、飞行员、管理员和其他"对象"进行设计。

Java 编程语言特别适合于实现面向对象设计,Java 中的所有可执行代码被组织成不同的类来表示对象,基于这个原因,Java 被认为是一种面向对象编程语言。

对象是一种软件单元,一个独特的类描述就可以生成一个对象,它被称为其类的一个实例,对象的创建过程被称为类的实例化。例如,下列代码就是对 java. util. Date 类的实例化:

```
java.util.Date today=new
java.util.Date();
```

变量 today 是对对象的引用,如图 1.2 所示。忽略引用与所引用对象之间的区别,我们也可以说 today 是 java. util. Date 对象的名称。

Java 类包含 3 种成员:字段、方法和构造函数。字段包含类对象的数据;方法包含对象执行的语句;构造函数包含用来初始化对象字段的代码。

面向对象设计指定软件中将要实例化的类,该设计可以通过统一建模语言(Unified Modeling Language,UML)进行简化和说明。在 UML 中,每一个类是通过一个矩形表示的,该矩形的不同部分列出该类的名称、字段、方法和构造函数。

图 1.3 显示了 Person 类的 UML 图,它包含 4 个字段(name、id、sex 和 dob),一个构造函数以及 3 个方法(isAnAdult()、setDob()和 toString())。这 8 个类成员中的每一个都带有一个可见度符号前缀:

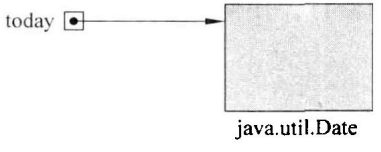

图 1.2 Java 对象

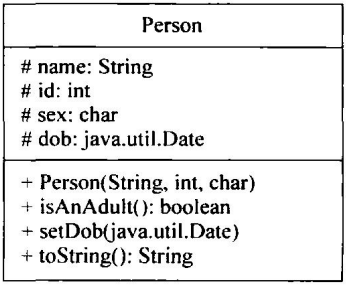

图 1.3 UML 图

＋表示 public

＃表示 protected

－表示 private

（包可见度没有相应的 UML 符号。）

UML 图独立于任何实现编程语言。在面向对象设计中，UML 图用于指定对象，这些图应该易于在 Java、C++ 或任何其他面向对象的编程语言中实现，它们为类提供一种伪代码。指定对象的状态（即字段）和行为（即方法），不指定如何实现相应的行为。UML 图不包含可执行代码。

指定对象的行为但不指定如何实现行为是一种抽象，这使得软件开发的设计阶段与实现阶段分离开来，另外，由于修改实现不会影响相关的模块，这也有助于软件的修改，只要一个方法的行为没有改变，该方法的实现发生改变将不会影响任何调用的模块。

例如，假定一个航空订票系统使用图 1.3 中 UML 图指定的 Person 类。我们推测，该软件将会在此系统的各种模块中调用该类的 isAnAdult() 方法，软件设计指定的“合同”只要求该方法返回正确的答案：当且仅当 x 为成人时，x. isAnAdult() 才为 true，而不管如何计算该结果。实现时可能会计算 private 字段 x. dob 的值与当前日期值之间的时间差，但合同中没有指定有关内容。而且，如果该实现发生改变，不会影响此航空订票系统中的其他代码。通过选择不同的算法来计算时间差，或者通过重新定义“成人”的含义，可以实现这类改变。

对使用方法的客户隐藏该方法的实现就称为信息隐藏。对于软件设计人员来说，其保密原则就是“如果你不需要知道，就不允许你知道”。这使得软件更易于设计、实现和修改。

1.3　抽象数据类型

抽象被用于帮助理解复杂的系统。尽管石头和网球是不同的东西，但它们下落的速率是一样的。物理学者通过这种单个假想质点的抽象来理解落体物理学。通过忽略不相干的事物（直径、重量），分析人员可以借助抽象来关注核心事物（高度）。

软件开发中广泛运用了抽象。UML 图通过关注类的字段（即状态）和方法（即行为）来提供抽象。不过，在某些层次上，类的字段甚至都可能无关紧要。

抽象数据类型（abstract data type，ADT）就是一种只给出该类型实例行为的规格，这类规格所需要的就是设计一种使用该类型的模块。

基本类型与 ADT 类似。我们知道 int 类型可以实现的操作（加、减、乘等），但不需要知道它如何实现这些操作，甚至不需要知道 int 是如何实际储存的。作为客户，我们可以使用 int 的有关操作，而无须知道这些操作是如何实现的。实际上，如果我们需要知道这些操作是如何实现的，则可能不是在设计使用这些操作的软件。与此类似，当你在打方向盘时，如果你需要知道自己的汽车是如何控制其前轮转向的，则或许加大了开车的难度！

例 1.1 分数 ADT。

大多数编程语言包含整数和实数(小数)类型,但不包含分数类型,这类数字可以作为对象实现,下面是一种分数类型设计:

```
ADT: Fraction
plus(Fraction): Fraction
times(Integer): Fraction
times(Fraction): Fraction
reciprocal(): Fraction
value(): Real
```

该 ADT 指定 5 种操作,注意 times()操作是重载的。

注意,ADT 使用更一般的类型:Integer 代替 int,Real 代替 double。这是因为假定 ADT 独立于任何特定编程语言。

通常来说,一个完整的 ADT 还会包含明确解释各个操作应该如何表现的文档,例如:

```
x.plus(y)返回的 Fraction 代表的是 x+y
x.times(n)返回的 Fraction 代表的是 n * x
x.times(y)返回的 Fraction 代表的是 x * y
x.reciprocal()返回的 Fraction 代表的是 1/x
x.value()返回 x 的数值
```

UML 图可用于指定 ADT,只需要将状态信息省略。图 1.4 中的 UML 图给出了例 1.1 中定义的 Fraction ADT。

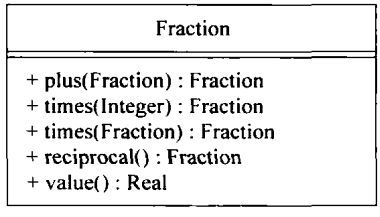

图 1.4 UML 中的 ADT

伪代码中可以使用 ADT 来实现独立于任何特定编程语言的算法,例 1.2 对此进行了说明。

例 1.2 在算法中使用 ADT。

两个数字 x 和 y 的谐函数是由公式 h=2/(1/x+1/y)定义的数字 h。在 Fraction 类型的伪代码中,这可以表示为

```
harmonicMean(x: Fraction, y: Fraction) returns Fraction
return x.reciprocal().plus(y.reciprocal()).reciprocal().times(2);
```

1.4 Java 接口

在 Java 中,ADT 可以通过接口来表示。我们回想一下,除了不包含可执行代码外,Java 接口就像 Java 类。