A Laboratory Course in Java

大学实验课程丛书

Java

上机实践指导教程

[美] Nell Dale 著

刘 谦 苏建平 等译

Jones and Bartlett

美迪亚

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

A Laboratory Course in Java

大学实验课程丛书

Java上机实践指导教程

〔美〕 Nell Dale 著

刘 谦 苏建平 等译

電子工業出版社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 提 要

本书代表了当前最流行的程序设计教学方法，每一章节都为读者精心设计了一组真实案例，目的是为了强化读者对程序设计概念的理解。在本书的每个章节中还提供了一定数量的程序案例分析，以便读者提高分析问题的能力。每章中的程序测试和调试练习也是针对提高读者实际解决问题的能力而设计的。由于程序设计是一门基于实践的科学，因此本书的重点就在于通过程序设计练习来强化读者对Java语法规则和程序设计方法的理解。

本书适用于初学Java程序设计的程序员，大专院校计算机软件专业的教师和学生。

版权贸易合同登记号：01-2002-5463

图书在版编目（CIP）数据

Java上机实践指导教程/（美）戴尔（Dale, N.）著；刘谦等译.—北京：电子工业出版社，2003.2
书名原文：A Laboratory Course in Java
ISBN 7-5053-8363-9

Ⅰ. J… Ⅱ. ①戴… ②刘… Ⅲ. Java语言－程序设计－教材 Ⅳ. TP312

中国版本图书馆CIP数据核字（2002）第102815号

责任编辑：徐云鹏 叶皓彤
印　　刷：北京天竺颖华印刷厂
出版发行：电子工业出版社 http://www.phei.com.cn
北京市海淀区万寿路173信箱 邮编：100036
北京市海淀区翠微东里甲2号 邮编：100036
经　　销：各地新华书店
开　　本：787×1092 1/16 印张：13 字数：330千字
版　　次：2003年2月第1版 2003年2月第1次印刷
定　　价：20.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换，若书店售缺，请与本社发行部联系。联系电话：（010）68279077

致　谢

没有一个作者是与世隔绝地进行写作的。总是有正式的或非正式的帮助从各方而来。在这里我要对我的同事对本书提出的各种建议表示感谢，特别是我所在部门的同事耐心地回答了关于Java语言的问题。除了要向曾经教我Java语言的Chip Weems表示感谢外，我还要向审阅《Java和软件设计》一书的下列朋友致谢，他们是John Connely，John Beidler，Hang Lau，Thomas Mertz，Bina Ramamurthy，James Roberts，David Shultz，Kenneth Slonneger，Sylvia Sorkin。特别需要感谢的是科尔比学院的Dale Skrien先生在百忙中抽出时间对本书原稿进行审阅并提出宝贵建议。

除了同事以外，我还要对Jones and Bartlett出版社的Brooke Albright和Mike以及Sigrid Wile和出版社的其他工作人员表示特别谢意。特别感谢J.Michael Stranz，Anne Spencer，Tara McCormick和Amy Rose等给与本书的特别关注。

简　介

为什么要选用Java语言

在将近20年的时间里，Pascal语言一直作为首选的程序设计语言用来教授计算机课程。但大约从7年前开始，C++语言开始取代Pascal语言，而紧接着又转向Java语言。但大家都承认对大多数初学者来说，学习C++和Java语言要比学习Pascal语言困难得多。

从学习的角度来看，C++和Java语言都不是专门为程序初学者设计的。这两种语言都是用于专业程序设计的高级程序设计语言。这两种语言都要求使用它们的程序设计人员具有程序设计的专业背景知识。对于初学程序设计的学生或其他人员来说，这两种语言所涉及的知识范围已经超出了他们的计算机专业基础。为了使初学者能够理解Java语言的基本语法和各种语义，我们设计了这套封闭式计算机科学实验教程来满足初学者掌握Java语言的要求。

计算机科学封闭式实验的特点

“计算机科学封闭式实验”[1]这一提法是由计算机专家Denning在一篇论文中首次提出的。到目前为止，人们对这一提法的含义有下面四种不同的定义：

1. 在规定的日程及时间之内，学员在老师的指导下进行程序设计。
2. 在规定的日程及时间之内，学员在老师的指导下对某一个题目进行反复实践。
3. 通过专门准备的实验教材，在老师的帮助下使学员通过与计算机交互对话来掌握某种原理和知识。本定义从内容上与Denning提倡的方法最为接近。
4. 组合使用上述两种或两种以上教学法。

虽然1991年出版的《Curriculum Report》[2]中已经建议把实验室教学推广到更广泛的学习领域，但该报告中没有对封闭式实验教学的具体内容给出明确的定义。事实上，该报告中提到的许多教学实验方法完全可以在开放式环境下实现。

请读者注意，本书中使用的教学实验方法是在定义2和定义3的基础上实现的。

开放与封闭实验环境

尽管《Denning Report》和1991年版的《Curriculum Report》中都指出实验教学应在指导下实施，但这并不是绝对的。我们认为封闭式实验的特点首先在于实验本身，然后是可以增加学员与其他学员和老师的接触时间。实际上，如果没有封闭学习环境的话，学员通过自己动手实验也可以有所收获。

注1：参见ACM通信第32卷No.1第9～第23页的Denning, P. J. (chair) “Computing as a Discipline（编程训练）”。

注2：Tucker, A. B. (Ed.) 编写的 “Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force（计算课程1991：ACM/IEEE-CS联合课程组报告）”，12月7日最终草案。ACM序号为201910。IEEE Computer Society Press序号为2220。

本书各章节的组织结构

本书每章都由三部分内容组成：预习、实验以及课后练习。预习的内容包括阅读和笔头练习。实验则由若干课组成，每课的内容都用来练习本章中讲授的某个基本概念。课后练习部分的主要内容是向学员布置应用每章所学内容进行程序设计的作业。上述三个部分内容的重点都是要求学员运用本章学习的基本概念。

如果在封闭试验教学环境下使用本书的话，我们建议最好在学员进入计算机房之前首先完成预习。这样一来，学员就可以利用进入机房的前几分钟对他们的练习答案进行核对（也就是每章第一课的内容）。本书规定的上机练习时间一般为两个小时左右。根据实际情况，教员也可以对各章的练习内容进行裁减，以适应特殊需要。

课后练习部分主要是向学员布置一些程序设计课题。教员在布置这些练习时，可根据学员的情况进行适当的调整。在大多数情况下，向学员布置一个具体的题目就可以了。

如果无法在封闭环境下使用本书的话，则教员可以根据具体情况要求学员自己独立完成全部或部分上机练习内容（有关这方面的详细内容，请参阅“灵活掌握”一节）。需要提醒大家的是，不管是在封闭或开放环境下，本书各章中的上机练习和课后作业都可以让学员以编组方式完成。

各项练习所涉及的理论基础

将每一章分为三部分是根据Benjamin Bloom在其教育分类学一书中提出的六级分类教学法安排的。在本书的具体情况下，我们把六级分类教学法简化为三个层次实施[3]。这三个层次是基于学习一个计算机算法（或一个计算机语言结构）的具体案例设计出来的，它们分别是：

学习阶段：在这一阶段，学员可以跟踪给定的算法并确认算法的输出结果。

模仿阶段：学员自己动手实现类似的算法。

设计阶段：学员可以自己动手对算法进行较大幅度的修改使其实现不同的功能，或者在不同的环境下应用所学的算法解决新的问题，以及自己将所学的多个算法进行组合应用或比较各算法的优缺点。

学习阶段与本书各章的预习部分是对应的。而大多数上机练习则是与模仿阶段有关的。设计阶段的内容主要对应各章的课后练习部分。

上述各种练习内容的设置还与学者Svinicki，Marilla D.以及Dixon Nancy提出的Kolb课堂教学模型有关[4]。该模型指出，学员的掌握程度与他们的主动参与程度成正比。由于阅读和写作是参与的两种形式，因此我们在各章的预习部分都要求学员进行阅读，而大量的练习则要求学员动手写出所发生的情况。仅仅观察程序的运行和输出结果是一种被动的学习方式，只有自己动手记下各种答案才是主动的参与。

注3：Bloom, Benjamin编写的“Taxonomy of Educational Objectives: Handbook I: Congnitive Domain（教育目标分类：手册I：认知领域）”，纽约David McKay于1956年出版。

注4：Svinicki, Mrilla D.和Dixon Nancy M.著“The Kolb Model Modified for Classroom Activities（针对课堂教学而修改的Kolb模型）”，选自College Teaching的Vol. 35, No. 4, Fall, 第141～146页。

灵活掌握

本书为指导老师的教学提供了最大可能的灵活性。每一章的开始部分都有一页以表格形式呈现的作业记录清单。该清单的第一列中给出了本章的各项内容，在第二列中学员应列出所留的作业，而该表的第三列则用来记录输出内容，该清单的最后一列用于教员为学员计分。为了便于提交该记录，学员可按虚线部分将其撕下。

学员用光碟

本书提供的光碟中包括了类、部分程序案例和数据文件。在本书的每个练习之前都列出了要使用的类文件或程序。本书没有提供用于调试练习程序的代码，这是因为书中的有些练习要求学员使用光碟中提供类的原始代码。使用本书的学员最好在开始学习前复制本书选配的光碟。

本书选配光碟中的每个子目录相应地与书中的一章相对应。各章使用的类和类文件都存储在扩展名为.java的类名文件中。程序包存储在与包名相同的目录中。各程序包中使用的编译单元则存储在扩展名为.java的公用类文件中。

译 者 的 话

本书作者Nell Dale是具有丰富程序设计教学经验的专家。Nell Dale 1960年毕业于波士顿大学并同时获得数学和心理学学士学位。1964年，Nell Dale又在得克萨斯大学获得数学硕士学位。并于1972年再次获得得克萨斯大学计算机科学博士学位。从1975年开始，作者开始在得克萨斯大学任教至今。

我们知道，由于Java语言是一种面向对象的程序设计语言，因此对于初学者来说，学习Java语言并不是一个轻松的任务。为了解决这一问题，作者利用自己多年教授程序设计语言积累的经验，在本书中向读者提供了一种快速高效学习Java程序设计方法的学习模式。本书为读者设计一种称之为三段教学法的学习模式。具体来讲，本书把学习Java语言每种结构的过程分为三个阶段实施，这三个阶段分别是课前预习、课中练习、课后复习。所谓课前预习就是集中介绍本课必须掌握的基本程序设计概念和Java语法规则。课中练习是通过向学员布置大量与本课内容有关的程序练习作业来强化本课概念的掌握。课后练习则是通过综合程序设计练习来扩展读者对本课内容灵活运用的能力。

本书的最大特点在于代表了当前最流行的程序设计教学方法，本书的每一章节都为读者精心设计了一组真实案例，其目的是为了强化读者对程序设计概念的理解。除此之外，本书的每个章节中还提供了一定数量的程序案例分析，以便读者提高解决分析问题的能力。每章中的程序测试和调试练习也是针对提高读者实际解决问题的能力而设计的。由于程序设计是一门基于实践的科学，因此本书的重点就在于通过程序设计练习来强化读者对Java语法规则和程序设计方法的理解。

正是基于上述考虑，本书借助于下面三种手段来提高读者的学习效率：

- 本书向初学者提供了三种类型的学习方案：课前预习、课中练习、课后复习。
- 以光碟形式向读者提供了练习使用的各种必备材料，其中包括本书各章程序设计练习使用的类、类框架和数据文件。
- 本书向读者提供了一种封闭式的教学模式，通过预习和强化程序设计练习为读者掌握Java语言结构的语法和语义提供快速学习Java语言的高效途径。

本书的读者对象是初学Java程序设计的程序员以及大专院校计算机软件专业的教师和学生以及有兴趣从事Java程序设计的初级程序员。

本书由刘谦、苏建平负责校审和统稿，参与本书翻译工作的其他人员还有王军平、刘丽云、钱云、刘颖、田红等人参与了本书的校对和录入。由于本书内容较新，篇幅较多，再加上译者的时间和水平有限，在翻译过程中难免有疏漏和错误，敬请读者给予批评指正。

目 录

致谢 iv

简介 v

译者的话 viii

第1章 程序设计和解决方案概述 1
作业单 1
上机之前的准备 2
上机之前的练习 6
课后练习 10

第2章 Java语法语义和程序编制过程 11
作业单 11
上机之前的准备 12
上机之前的练习 16
课后练习 20

第3章 事件驱动的输出 21
作业单 21
上机之前的准备 22
上机之前的练习 26
课后练习 30

第4章 数值类型和表达式 32
作业单 32
上机之前的准备 33
上机之前的练习 38
课后练习 44

第5章 事件驱动输入和软件设计方法 46
作业单 46
上机之前的准备 47
上机之前的练习 53

课后练习 61

第6章 条件、逻辑表达式和选择控制结构 63
作业单 63
上机之前的准备 64
上机之前的练习 71
课后练习 82

第7章 类和方法 84
作业单 84
上机之前的准备 85
上机之前的练习 89
课后练习 96

第8章 继承、多态性和作用域 98
作业单 98
上机之前的准备 99
上机之前的练习 102
课后练习 108

第9章 文件输入输出和循环 110
作业单 110
上机之前的准备 111
上机之前的练习 118
课后练习 123

第10章 其他控制结构和异常处理 125
作业单 125
上机之前的准备 126
上机之前的练习 130
课后练习 138

第11章 一维数组 140
作业单 140
上机之前的准备 141
上机之前的练习 144
课后练习 149

第12章 基于数组的列表 151
作业单 151
上机之前的准备 152
上机之前的练习 156
课后练习 161

第13章 多维数组 163
作业单 163
上机之前的准备 164
上机之前的练习 165
课后练习 170

第14章 递归 171
作业单 171
上机之前的准备 172
上机之前的练习 173
课后练习 177

附录A Java保留字 178

附录B 操作符优先级 179

附录C Java基本数据类型 181

附录D 统一编码ASCII码子集 182

第1章　程序设计和解决方案概述

目标

- 登录到计算机
- 在计算机上进行下列操作
  - 改变活动（工作）目录
  - 显示目录中的文件
- 使用文本编辑器和Java编译程序完成下列任务
  - 载入程序文件
  - 变更程序文件名称
  - 保存文件
  - 打印文件
  - 编译程序
  - 运行程序
  - 修改程序并重新运行
  - 修改程序中的错误
  - 输入数据和运行程序
  - 退出系统

作业单

姓名_____　日期_____

小组_____

请读者在下表中给出提交的每一课的练习并指明所提交的形式。提交的形式有三种：（1）上机操作表；（2）输出文件的清单；（3）程序清单。教员或助教可以用该表最后一列来给学员的练习结果打分。

活动	布置的作业：检查或列出练习编号	提交 (1)	(2)	(3)	成绩
上机之前					
复习					
上机之前的练习					
上机实践					
课1-1：检查上机之前的练习					
课1-2：基本文件操作					
课1-3：程序的编译和运行					
课1-4：程序文件的编辑、运行和打印					
课1-5：运行有错误的程序					
课1-6：输入、编译和运行新程序					
课后练习					

上机之前的准备

复习

我们使用的计算机是一种可编程的，具有存储、检索和数据处理能力的电子设备。计算机的存储、检索和数据处理能力是借助于下面五个基本物理部件实现的，这五个部件分别是存储器部件、算术逻辑部件、控制部件、输入和输出部件。我们把上述五个组成计算机的基本物理部件称为计算机硬件。相比之下，我们把在计算机上运行的程序叫做软件。程序设计的任务就是编制可构成软件的程序。

编程

程序是由一系列用来执行特定操作的指令序列组成的。而程序设计就是用来确定指令序列的过程。程序设计可分为两个阶段来实现：程序设计的第一阶段决定所需实现的任务，第二阶段则是使用计算机指令来实现任务所需的具体操作。

一般来说，程序设计是针对某一特殊问题的。我们不会漫无目的地进行程序设计，程序的功能就是要解决某一问题或实现某种功能。因此确定程序的功能就是提出问题的解决方案。通常我们把程序设计的这一阶段称为问题解决阶段。

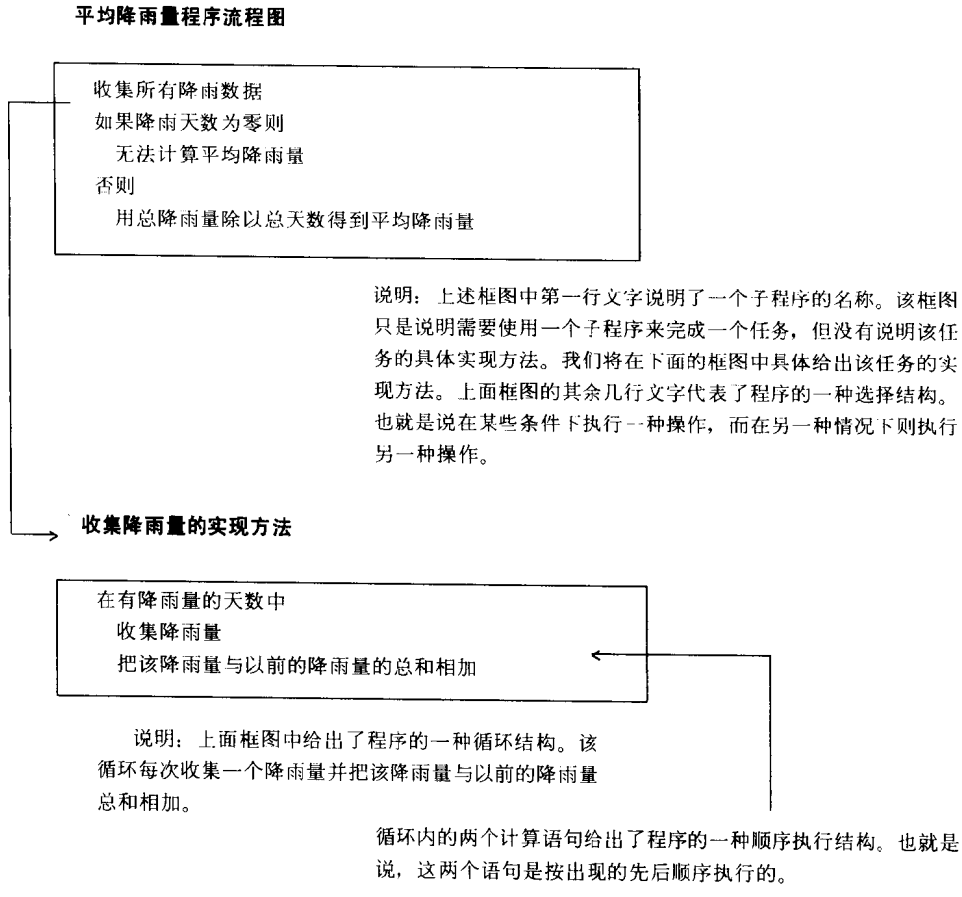
程序设计的第二阶段是具体实现程序功能的阶段，也就是用指令表达解决方案的过程。我们在本阶段的任务是把第一阶段提出的问题解决方案用具体的程序设计语言来表达。在程序设计的每个阶段都要对方案和程序的正确性进行测试。在开始具体编程之前必须确保设计方案的正确性。

下面让我们用一个具体案例来说明程序设计的过程。

问题：计算若干天的平均降水量。

讨论：如果我们用手工完成上述任务的话，首先要手工写下每天的降雨量。接着把所有的降雨量加起来再除以总天数，从而得到这些天的平均降雨量。我们可以完全按照手工方法来编制解决该问题的程序。

算法：



下面给出的代码是用来实现上述算法的Java源程序。读者没有必要理解该程序中使用的所有代码的用途。随着课程的逐步深入，读者最终将会加深对各种语句的理解。

在这里要说明的是，该程序中符号 /*和*/之间的文字是程序的注释。这种信息是用来说明程序用途的文字，Java编译程序在编译阶段是不会处理这些信息的。除了这种注释信息外，该程序中符号“//”后面的文字也是注释文字。

```
/* Class RainFall calculates the average rainfall over a
   period of days.  The number of days and the rain statistics
   are in file Rain.in */
```

```
import java.io.*;

public class RainFall
{
  // Declare dataFile for input
  private static BufferedReader dataFile;

  static double getInches(BufferedReader dataFile,
    int numberDays) throws IOException
  // Reads and returns the total inches of rain
  {
    double total = 0.0;
    double inches;
    int days = 1;
    while (days <= numberDays)
    {
      inches =
        Double.valueOf(dataFile.readLine()).doubleValue();
      total = total + inches;
      days++;
    }
    return total;
  }

  public static void main(String[] args) throws IOException
  // Main is where execution starts.  It opens the data file,
  //  reads the number of days of rain to be totaled, calls an
  //  auxiliary method to read and sum the rainfall, and
  //  prints the average rainfall on the screen
  {
    double totalRain;
    int numDays;
    double average;
    // Instantiate and open the data file
    dataFile = new BufferedReader(
      new FileReader("rainFile.in"));
    // Input the number of days
    numDays = Integer.valueOf(dataFile.readLine()).intValue();

    totalRain = getInches(dataFile, numDays);

    if (numDays == 0)
      System.out.println("Average cannot be computed "
          + "for 0 days.");
    else
    {
```

```
            average = totalRain / numDays;
            System.out.println("The average rainfall over " +
              numDays + " days is " + average);
          }
          dataFile.close();
        }
      }
```

上面的源程序向读者介绍的主要概念包括子程序、分支选择、循环和顺序执行结构。除此之外，Java程序还有一种称之为异步事件的第四种程序结构。如果我们的算法中需要用户输入降雨量数据的话，则处理输入的程序就是一个事件驱动的程序。事件驱动也是一种异步处理方法。如果上面的程序需要用户从屏幕上输入数据并按下显示按钮的话，则处理按钮的过程就是异步处理事件。

开始

有些Java系统向用户提供了可进行创建并执行Java程序的集成应用环境。例如，可在PC个人电脑和苹果电脑上运行的Caf和CodeWarrior的Java系统就是这种应用环境。这种Java系统的特点是把Java编译器、编辑程序和Java程序运行时系统集成在一个环境下使用。而非集成Java系统则要求用户使用一般的编辑器来建立Java文本，然后再单独进行编译和运行。由于使用非集成Java系统要求用户掌握多种程序的使用方法，为了便于集中精力学习Java语言，我们在下面的内容中将在模拟环境下向读者介绍建立运行Java程序的步骤。

当首次登录到某台电脑上时，我们首先接触到的软件就是在该电脑上正在运行的操作系统。我们可以把电脑的操作系统看做是用来连接其他各种软件的房间走廊。这时只要输入希望运行的软件名称，该操作系统就可以启动该软件运行。当不再使用该软件时，我们就要返回到该操作系统下，以便运行其他的软件。

操作系统连接的各种软件就像是房间的各个入口。当需要使用某个软件时，操作系统就打开相应的房门并把用户引入该软件所在的房间。如果正在使用编辑程序，则我们就可以在该软件下创建一个文件。该文件的内容可以是程序代码或者是程序使用的数据。

对于从来没有接触过编辑程序的读者来说，我们可以把该软件看成是允许我们利用键盘和显示器输入信息的打字机。而文件则是用来存储我们通过该打字机输入的信息载体。这时我们可以在电脑的屏幕上看到从键盘输入的文字。而编辑软件提供的各种命令就类似于我们用手工方法控制打字机的过程。编辑器提供的各种命令可用来修改或调整句子、文字或字符的内容和位置。文件存储在电脑的磁盘存储器中，每个文件都有其惟一的名称并用来存放一系列的数据。

当我们结束数据输入后，下面的任务就是为输入的数据指定一个文件名并告诉编辑器将该文件保存在磁盘存储器中。文件名就是为存储在文件夹中的数据指定一个名称，以便我们可以使用该名称来引用这些数据。

在完成编辑任务后，系统将返回到操作系统提示符下等待用户输入下一个命令。如果我们创建了一个Java程序的话，下面就可以输入编译该Java程序的命令。这时操作系统就会通知Java编译器来编译我们刚才输入的Java程序。

Java编译器的功能是把用户的Java源程序转换为操作系统可执行的命令。如果我们创建的Java源程序中有语法错误，则Java编译器就会通知我们。这时我们就必须再次使用编辑程序来改正程序中的错误语句。如果我们的Java程序可以通过Java编译器的检查和处理，则Java编译器就会为我们生成一个可执行文件。

完成编译后，现在的任务就是通知操作系统来运行我们的可执行文件。操作系统负责把带有目标代码的可执行文件载入到内存中运行。可执行文件的运行可以解决我们要处理的问题。

当可执行文件执行完毕后，我们就可以通过注销命令（log off）来退出操作系统。

某些Java系统把编辑软件、编译器以及Java运行时程序捆绑为一个系统。例如，著名的CodeWarrior就是一个Java集成系统。在该系统下，用户不仅可借助于一组菜单来实现编辑、编译和运行Java程序的各种任务，而且还可以在遇到问题时得到该系统提供的在线帮助。用户可以通过使用鼠标、键盘或功能键来进入级联菜单选择更多的功能。集成系统的特点是具有友好的用户界面，在菜单和帮助信息的引导下，用户可以轻松地编辑、编译并执行自己的Java程序。

上机之前的练习

姓名_____ 日期_____

小组_____

教员应在本节向学员提供讲义，介绍正在使用的计算机系统的配置。

练习1： 你正在使用何种类型的电脑？

练习2： 你使用的电脑配置了何种操作系统？

练习3： 你正在使用何种类型的Java系统？

练习4： 你使用的Java系统是否提供了集成环境？

练习5： 是否使用了编辑程序，该编辑程序的名称是什么？

练习6： 如果没有使用Java集成环境，请给出所使用编译、编辑和运行命令的名称。