华章教育

全美经典
学习指导系列

数据结构习题与解答

(英文版)

Java语言描述

DATA STRUCTURES WITH JAVA

- Covers Computer Science I-II— supplements any class text
- Simple explanation of the ideal programming language for the Internet
- Over 260 solved problems step-by-step
- Ideal for independent study!

(美) John R. Hubbard 著

机械工业出版社
China Machine Press

全球销售超过3000万册!

全美经典
学习指导系列

数据结构习题与解答

Java语言描述

（英文版）

DATA STRUCTURES WITH JAVA

（美）John R. Hubbard 著

机械工业出版社
China Machine Press

John R. Hubbard: Data Structures With Java（ISBN 0-07-136128-6）.

本书版权登记号：图字：01-2002-2195

图书在版编目（CIP）数据

数据结构习题与解答：JAVA语言描述/（美）哈伯特（Hubbard, J. R. ）著. - 北京：机械工业出版社，2002.8

（全美经典教学辅导系列）

书名原文：Data Structures With Java

ISBN 7-111-10419-6

Ⅰ. 数… Ⅱ. 哈… Ⅲ. ① 数据结构 - 习题 - 英文 ② JAVA语言 - 程序设计 - 习题 - 英文

Ⅳ. TP311.12-44

中国版本图书馆CIP数据核字（2002）第038499号

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：华 章

北京牛山世兴印刷厂印刷 · 新华书店北京发行所发行

2002年8月第1版第1次印刷

787mm × 1092mm 1/16 · 24印张

印 数：0 001-3 000册

定 价：38.00元

Preface

Like all Schaum's Outline Series books, this volume is intended to be used primarily for self study, preferably in conjunction with a regular course in data structures. However, it is also well-suited for use in independent study or as a reference.

The book includes over 200 examples and solved problems. I firmly believe that the principles of data structures can be learned from a well-constructed collection of examples with complete explanations. This book is designed to provide that support.

Source code for all the examples and problems in this book can be downloaded either from my website at `http://projectEuclid.net/schaums/dswj/` or from its mirror site at `http://www.richmond.edu/~hubbard/schaums/dswj/`. Any corrections or addenda for the book will also be available at these sites.

I wish to thank all my friends, colleagues, students, and the McGraw-Hill staff who have helped me with the critical review of this manuscript. Special thanks to my wife and colleague, Anita H. Hubbard, for her advice, encouragement, and creative ideas for this book. Many of the original problems used here are hers.

JOHN R. HUBBARD
Richmond, Virginia

JOHN R. HUBBARD is Professor of Mathematics and Computer Science at the University of Richmond. He received his Ph.D. from the University of Michigan in 1973 and has been a member of the Richmond faculty since 1983. His primary interests are in numerical algorithms and database systems. Dr. Hubbard's other *Schaum's Outline* books include *Programming with C++*, *Programming with Java*, *Fundamentals of Computing with C++*, and *Data Structures with C++*.

国外经典教材

《程序设计实践》
Brian W. Kernighan, Rob Pike 著/裘宗燕 译/20元
《Linux操作系统内核实习》
Gary J. Nutt 著/陆丽娜 等译/29元
《程序设计语言概念和结构》(原书第2版)
Ravi Sethi 著/裘宗燕 等译/45元
《编译原理》
Alfred Aho 著/李建中 等译
《C++程序设计语言》(特别版)
Bjarne Stroustrup 著/裘宗燕 译/85元
《并行计算机体系结构》(原书第2版)
David E. Culler, Jaswinder Pal Singh 著/李晓明 等译
《人工智能》
Nils J. Nilsson 著/郑扣根 等译/39元
《神经网络设计》
Hagan 著/戴葵 等译
《计算理论导引》
Sipser 著/张立昂 等译/30元
《数据库系统概念》(原书第4版)
Silberschatz 著/杨冬青 等译
《数据结构、算法与应用——C++语言描述》(中文版)
Sartaj Sahni 著/戴葵 等译/49元
《离散数学及其应用》(原书第4版)
Kenneth H. Rosen 著/袁崇义 等译/75元
《计算机图形学的算法基础》(第2版)
David F. Rogers 著/石教英 等译/55元
《可扩展并行计算:技术、结构与编程》
Kai Hwang, Zhiwei Xu 著/陆鑫达 等译/49元
《软件工程:实践者的研究方法》(原书第5版)
Roger S. Pressman 著/梅宏 译
《数据通信与网络》(第2版)
Behrouz Forouzan 著/吴时霖 等译/68元
《数据库原理、编程与性能》(原书第2版)
Patrick O' Neil, Elizabeth O' Neil 著/周傲英 等译/55元
《现代操作系统》(第2版)
Andrew S. Tanenbaum 著/陈向群 等译
《C程序设计语言》(第2版)
Kernighan, Ritchie 著/徐宝文 等译/28元
《组合数学》
Richard A. Brualdi 著/冯舜玺 等译/38元
《结构化计算机组成》
Andrew S. Tanenbaum 著/刘卫东 等译/46元

计算机科学丛书

《设计模式 可复用面向对象软件的基础》
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides 著/吕建 等译/35元
《UNIX环境高级编程》
W. Richard Stevens 著/尤晋元 等译/55元
《TCP/IP详解 卷1:协议》
W. Richard Stevens 著/谢希仁 等译/45元
《TCP/IP详解 卷2:实现》
Gary R. Wright, W. Richard Stevens 著/谢希仁 等译/78元
《TCP/IP详解 卷3:TCP事务协议、HTTP、NNTP和UNIX域协议》
W. Richard Stevens 著/谢希仁 等译/35元
《数据库系统导论》
C. J. Date 著/王珊 等译/66元
《计算机图形学原理及实践——C语言描述》(原书第2版)
James D. Foley, Andries Van Dam, Steven K. Feiner, John F. Hughes 著/唐泽圣 等译
《C 语言解析教程》(原书第4版)
Ira Pohl 著/麻志毅 等译/48元
《C++语言的设计和演化》
Bjarne Stroustrup 著/裘宗燕 译/48元
《人本界面——设计交互式系统的最新指示》
Jef Raskin 著/史元春 译
《分布式系统设计》
Jie Wu 著/高传善 等译/30元
《数据通信与网络教程》
William A. Shay 著/高传善 等译/40元
《专家系统原理与编程》
Giarratano, Riley 著/印鉴 等译/49元
《编译原理及实践》
Kenneth C. Louden 著/冯博琴 等译/39元
《最新网络技术基础》
Palmer 著/严伟 等译/20元
《数字逻辑:应用与设计》
John M. Yarbrough 著/朱海滨 等译/49元
《计算机文化》
June Jamrich Parsons, Dan Oja 著/朱海滨 等译/50元
《信息系统原理》
Ralph M. Stair, George W. Reynolds 著/张靖 等译/42元

《计算机信息处理》

Mandell 著/尤晓东 等译/38元

《数据库管理系统基础》

Philip J. Pratt, Joseph J. Adamski 著/戴葵 等译/20元

《数据仓库》

W. H. Inmon 著/王志海 等译/25元

《数据广播》

Lars Tvede 著/徐良贤 等译/28元

《数据库设计》

Ryan K. Stephens Ronald R. Plew 著/何玉洁 武欣 等译/35元

《软件工程——实践者的研究方法》(原书第4版)

Roger S. Pressman 著/梅宏 等译/48元

《软件工程——Java语言实现》

Stephen R. Schach 著/袁兆山 等译/38元

《数据库系统概念》(中文版)

Abraham Silberschatz, Henry F. Korth, S. Sudarshan 著/杨冬青 等译/49元

《数据通信与网络》

Behrouz Forouzan 著/吴时霖 等译/48元

《计算机网络》(第2版，中文版)

Larry L. Peterson, Bruce S.Davie 著/叶新铭 等译/49元

《高性能通信网络》(原书第2版)

Jean Walrand, Pravin Varaiya 著/史美林 等译/55元

《嵌入式计算系统设计原理》

Wayne Wolf 著/孙玉芳 等译/65元

《数据挖掘：概念与技术》

Jiawei Han, Micheline Kamber 著/范明 孟小峰 等译/39元

《编码的奥秘》

Charles Petzold 著/陆丽娜 等译/24元

《软件需求》

Karl E. Wiegers 著/陆丽娜 等译/19元

《面向对象程序设计——图形应用实例》

Micheal J. Laszlo 著/何玉洁 等译/35元

《现代数据库管理》(原书第6版)

Jeffrey A. Hoffer 著/施伯乐 杨卫东等译

《数据库系统基本原理》

Jeffrey D. Ullman 著/周傲英 等译

《数据库与事务处理》

Philip M. Lewis 著/施伯乐 杨卫东 等译

《数据库系统教程》

Hector Garcia-Molina 著/岳丽华 等译

《UNIX操作系统教程》

Sarwar 著/徐良贤 等译

《计算机网络与因特网》(第2版)

Douglas E. Comer 著/徐良贤 等译/40元

《数据库系统实现》

Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom 著/杨冬青 等译/45元

《ISDN、B-ISDN与帧中继和ATM》(原书第4版)

William Stallings 著/程时端 等译/48元

《UNIX操作系统设计》

Maurice J. Bach 著/陈葆珏 等译/35元

《Java 程序设计教程》(原书第3版)(上册)

Harvey Deitel 著/袁兆山 等译/55元

《Java 程序设计教程》(原书第3版)(下册)

Harvey Deitel 著/袁兆山 等译/69元

《C++精髓：软件工程方法》

Victor Shtern 著/李师贤 等译/85元

《Java编程思想》(第2版)

Bruce Eckel 著/侯捷 译/99元

《C++编程思想》(第2版)

Bruce Eckel 著/刘宗田 等译/59元

《并行程序设计》

Barry Wilkinson 著/陆鑫达 等译/43元

《神经网络原理》

Simon Haykin 著/史忠植 等译

《分布式操作系统：原理与实践》

Doreen L. Galli 著/徐良贤 等译

《现代操作系统》

Andrew S. Tanenbaum 著/陈向群 等译/40元

《Java编程思想》

Bruce Eckel 著/尤晓东 等译/60元

《Unix编程环境》

Brian W. Kernighan, Rob Pike 著/陈向群 等译/24元

《Internet技术基础》

Douglas E. Comer 著/袁兆山 等译/18元

《C++编程思想》

Bruce Eckel 著/刘宗田 等译/39元

《电子商务》

Gary P. Schneider, James T. Perry 著/成栋 等译/28元

《计算机网络实用教程实验手册》

Tamara Dean 著/陶华敏 等译/15元

《计算机网络实用教程》

Todd Meadors 著/陶华敏 等译/65元

《C程序设计教程》

Deitel 著/张祖荫 等译/33元

《C++程序设计教程》

Deitel 著/马鸣远 等译/22元

经典原版书库

《并行计算机体系结构》(英文版)
David E. Culler, Jaswinder Pal Singh 著/88元
《人工智能》(英文版)
Nils J. Nilsson 著/45元
《高性能通信网络》(英文版)
Jean Walrand, Pravin Varaiya 著/64元
《计算机网络》(英文版)
Larry L. Peterson, Bruce S. Davie 著/65元
《计算机体系结构：量化研究方法》第2版(英文版)
David A. Patterson, John L. Hennessy 著/88元
《计算机组织与设计：硬件/软件接口》第2版(英文版)
John L. Hennessy, David A. Patterson 著/80元
《计算机图形学的算法基础》(英文版·第2版)
David F. Rogers 著/45元
《计算机组成》(英文版·第5版)
Carl Hamacher 著/48元
《可扩展并行计算：技术、结构与编程》(英文版)
Kai Hwang, Zhiwei Xu 著/69元
《数据库系统概念》(英文版)
Abraham Silberschatz, Henry F. Korth, S. Sudarshan 著/65元
《高级计算机体系结构》(英文版)
Kai Hwang 著/59元
《数据结构、算法与应用——C++语言描述(英文版)
Sartaj Sahni 著/66元
《软件工程——实践者的研究方法》(英文版)
Roger S. Pressman 著/68元
《软件工程——Java语言实现》(英文版)
Stephen R. Schach 著/51元
《通信网络基础》(英文版)
Jean Walrand 著/32元
《数据通信与网络》(英文版)
Behrouz Forouzan 著/59元
《离散数学及其应用》(英文版)
Kenneth H. Rosen 著/59元
《计算机体系结构：量化研究方法》(英文版·第3版)
John L. Hennessy, David A. Patterson 著/
《现代操作系统》(英文版·第2版)
Andrew S. Tanenbaum 著/48元
《组合数学》(英文版·第3版)
Richard A. Brualdi 著/35元
《ISDN、B-ISDN与帧中继和ATM》(英文版·第4版)
William Stallings 著/35元
《数据库系统导论》(英文版·第7版)
C. J. Date 著/65元
《C++编程思想》(英文版·第2版)
Bruce Eckel 著/58元
《程序设计实践》(英文版)
Brian W. Kernighan, Rob Pike 著/22元
《数据库系统实现》(英文版)
Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom 著/42元
《Internet技术基础》(英文版·第3版)
Douglas E. Comer 著/23元
《结构化计算机组成》(英文版·第4版)
Andrew S. Tanenbaum 著/38元
《UNIX环境高级编程》(英文版)
W. Richard Stevens 著/49元
《网络互连：网桥.路由器.交换机和互连协议》(英文版·第2版)
Radia Perlman 著/36元
《TCP/IP详解 卷1：协议》(英文版)
W. Richard Stevens 著/39元
《Java编程思想》(英文版·第2版)
Bruce Eckel 著/69元
《设计模式：可复用面向对象软件的基础》(英文版)
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides 著/38元
《TCP/IP详解 卷2：实现》(英文版)
Gary R. Wright,W. Richard Stevens 著/69元
《TCP/IP详解 卷3：TCP事务协议、HTTP、NNTP和UNIX域协议》(英文版)
W. Richard Stevens 著/28元
《人本界面——设计交互式系统的最新指示》(英文版)
Jef Raskin 著/28元
《程序设计语言：概念和结构》(英文版·第2版)
Ravi Sethi 著/39元
《C++语言的设计和演化》(英文版)
Bjarne Stroustrup 著/29元
《UNIX操作系统教程》(英文版)
Sarwar 著/49元
《高速网络与因特网——性能与服务质量》(英文版·第2版)
William Stallings 著/45元
《Linux操作系统内核实习》(英文版)

Gary J. Nutt 著/32元

《具体数学：计算机科学基础》（英文版·第2版）

Ronald Graham 著/49元

《计算机图形学原理及实践——C语言描述》（英文版·第2版）

James D. Foley, Andries Van Dam, Steven K. Feiner, John F. Hughes 著/88元

《编写有效用例》（英文版）

Alistair Cockburn 著/25元

《系统分析与设计》（英文版）

John W. Satzinger, Robert B. Jackson, Stephen D. Burd 著/60元

《计算机文化》（英文版·第4版）

Parsons Oja 著/55元

《电磁场与电磁波》（英文版）

Guru 著/68元

《面向对象与经典软件工程》（英文版）

Stephen R. Schach 著

《数据通信与网络教程》（英文版）

William A. Shay 著

《数字逻辑应用与设计》（英文版）

John M. Yarbrough 著

《神经网络设计》（英文版）

Hagan 著

《专家系统原理与编程》（英文版）

Giarratano, Riley 著

《编译原理与实践》（英文版）

Kenneth C. Louden 著

《计算理论导引》（英文版）

Sipser 著

全美经典学习指导系列

《关系数据库习题与解答》（英文版）

Ramon A.Mata-Toledo 著/26元

《关系数据库习题与解答》（中译本）

Ramon A. Mata-Toledo 著/周云晖 等译/19元

《计算机体系结构习题与解答》（英文版）

Nicholas Carter 著/30元

《计算机图形学习题与解答》（英文版·第2版）

Zhigang Xiang 著/35元

《计算机图形学习题与解答》（中译本）

Zhigang Xiang 著/陈泽琳 等译/29元

《数据结构习题与解答——Java语言描述》（英文版）

John R. Hubbard 著/38元

《数据结构习题与解答——Java语言描述》（中译本）

John R. Hubbard 著/阳国贵 等译/39元

《C++编程习题与解答》（英文版·第2版）

John R. Hubbard 著/39.5元

《C++编程习题与解答》（中译本）

John R. Hubbard 著/徐漫江 等译/39元

《操作系统习题与解答》（英文版）

J.Archer Harris 著/25元

《Java 编程习题与解答》（英文版）

John R. Hubbaro 著/28元

《Java 编程习题与解答》（中译本）

Hubbard 著/王强 等译/29元

《计算机科学导论习题与解答》（英文版）

Ramon A. Mata-Toledo 著/30元

《数据结构习题与解答——C++语言描述》（英文版）

John R. Hubbard 著/38元

《软件工程系统与解答》（英文版）

David A. Gustafson 著

《计算机网络习题与解答》（英文版）

Ed Tittel 著

《计算机导论习题与解答》

Pauline K. Cushman 著/薛静锋 等译/29元

《Visual Basic 编程习题与解答》

Byron S. Gottfried 著/周旭 等译/29元

《SQL 编程习题与解答》（中译本）

Mata Toledo 著/胡志军 等译/29元

Contents

CONTENTS

CONTENTS

Chapter 1

Review of Java

This chapter outlines those features of the Java programming language that are essential to the design and implementation of data structures. For more details, see the book **[Hubbard]** listed in Appendix D on page 362.

1.1 OBJECT-ORIENTED PROGRAMMING

Java is an *object-oriented* programming language:

- Every data item is encapsulated in some object.
- Every executable statement is performed by some object.
- Every object is an instantiation of some class or is an array.
- Every class is defined within a single inheritance hierarchy.

Java's single inheritance hierarchy is a tree structure whose root is the `Object` class. (See Section 3.4 on page 58.) This single structure might seem to render Java a simple language. However, that structure includes the Java 1.3 Class Libraries, which contain 2,130 classes with 23,901 members.

Object orientation makes Java particularly well-suited for the design and implementation of data structures. Its Collections Framework (see Chapter 4) provides an excellent platform for the programmer to construct data structures.

1.2 THE JAVA PROGRAMMING LANGUAGE

A *Java program* is a collection of one or more text files, at least one of which contains a unique `public` class with a unique method whose signature is

```
public static void main(String[] args)
```

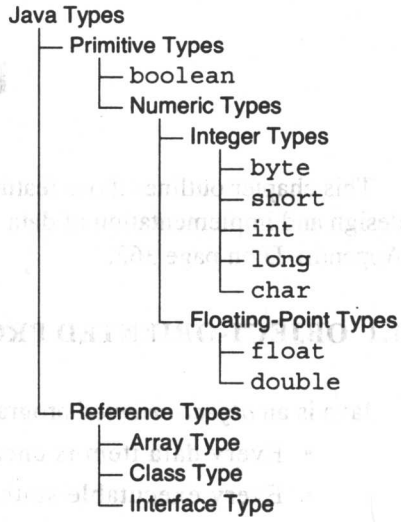
Each file in the program must contain a unique `public` class or `public` interface. Each file is named *X*`.java`, where *X* is the name of its unique `public` class or `public` interface. Each file *X*`.java` is compiled into a corresponding class file named *X*`.class`. Then the program is run by executing the *X*`.class` file that contains the `main(String[])` method that in turn contains the statements to be executed. This programming can be done from the command line or from within an IDE. (See Appendix C.)

1.3 VARIABLES AND OBJECTS

In any programming language, data are accessed through variables. In Java, a variable is either a reference to an object or one of the eight primitive types, as shown in the diagram at the top of the next page. If it is a reference, then its value is either `null` or it contains the address of an instantiated object.

Each object belongs to a unique class. The class defines its objects' properties and operations, just as a primitive type defines its variables' properties and operations.

A variable is created by a declaration that specifies its type and optional initial value. An object is created by the new operator invoking its class constructor.



EXAMPLE 1.1 Creating Variables and Objects

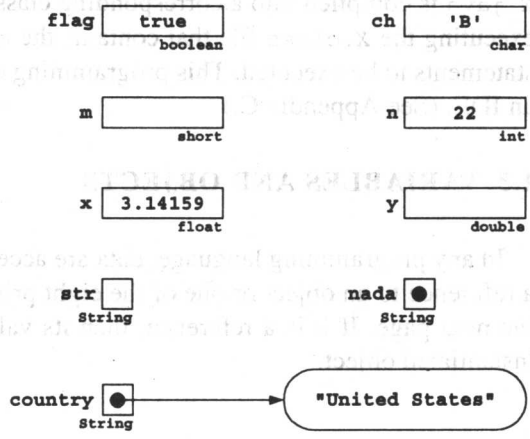
```
public class Ex0101
{ public static void main(String[] args)
  { boolean flag=true;
    char ch='B';
    short m;
    int n=22;
    float x=3.14159F;
    double y;
    String str;
    String nada=null;
    String country = new String("United States");
    System.out.println("flag = " + flag);
    System.out.println("ch = " + ch);
    System.out.println("n = " + n);
    System.out.println("x = " + x);
    System.out.println("nada = " + nada);
    System.out.println("country = " + country);
  }
}
```

```
flag = true
ch = B
n = 22
x = 3.14159
nada = null
country = United States
```

This program declares nine variables and then prints the six that were initialized. The first six variables are primitive types. The last three variables are references to String objects.

The program contains only one object: the String object named country. (Technically, country is the name of the variable that refers to the String object.)

The picture here shows the nine variables and the one object. Each variable has a name and a type. If the type is a class, then the variable is a reference to an instance (i.e., an object) of that class. The uninitialized variables appear empty; the others contain their values. The only possible value of a



reference variable is a reference, drawn as a black dot. If the reference is not `null`, then the dot has an arrow pointing to the object to which it refers. In Java, an object cannot exist unless some reference variable refers to it.

1.4 PRIMITIVE TYPES

Java defines eight primitive types:

`boolean` either `false` or `true`

`char` 16-bit Unicode character; e.g., `'B'` or `'π'`

`byte` 8-bit integer: `-128` to `127`

`short` 16-bit integer: `-32768` to `32767`

`int` 32-bit integer: `-2147483648` to `2147483647`

`long` 64-bit integer: `-9223372036854775808` to `9223372036854775807`

`float` 32-bit floating-point decimal: (plus or minus) `1.4E-45F` to `3.4E38F`

`double` 64-bit floating-point decimal: (plus or minus) `4.9E-324` to `1.8E308`

Note that `float` literals require an `F` suffix to distinguish them from values of type `double`.

Every Unicode character literal can be expressed in the form `'\uxxxx'`, where x is any hexadecimal digit. For example, `'B'` is `'\u0042'`, and `'π'` is `'\u03C0'`.

In addition to their numeric values, floating-point variables can also have any of the three special values `NEGATIVE_INFINITY`, `POSITIVE_INFINITY`, and `NaN` (for "not a number"). These special values result from improper arithmetic operations.

EXAMPLE 1.2 Special Values for Floating-Point Types

```
public class Ex0102
{ public static void main(String[] args)
  { double x=1E200;  // x=10000...0 (1 followed by 200 zeros)
    System.out.println("x = " + x);
    System.out.println("x*x = " + x*x);
    System.out.println("(x*x)/x = " + (x*x)/x);
    System.out.println("(x*x)/(x*x) = " + (x*x)/(x*x));
  }
}
x = 1.0E200
x*x = Infinity
(x*x)/x = Infinity
(x*x)/(x*x) = NaN
```

The value of `x*x` is `Infinity` because $(10^{200})(10^{200}) == 10^{400}$, which is greater than the maximum value $1.3(10^{308})$ for `double` type.

Algebraically, $(xx)/x = x$, but not so when x is infinite: `Infinity` divided by any nonnegative finite value will still be `Infinity`.

Finally, `Infinity` divided `Infinity` is not even infinite; it is algebraically indeterminate, so its Java value is `NaN`.

For each of the eight primitive types, Java defines a *wrapper class* that provides object-oriented services for manipulating primitive types. For example, the wrapper class for type `double` is `Double`, and the wrapper class for type `int` is `Integer`.

EXAMPLE 1.3 Using Wrapper Classes

```
public class Ex0103
{ public static void main(String[] args)
  { String s = "2.7182818284590";
    System.out.println("s = " + s);
    Double x = new Double(s);
    System.out.println("x = " + x);
    double y = x.doubleValue();
    System.out.println("y = " + y);
    s += 45;
    System.out.println("s = " + s);
    y += 45;
    System.out.println("y = " + y);
    int n = x.intValue();
    System.out.println("n = " + n);
    n = Integer.parseInt("3A9",16);
    System.out.println("n = " + n);
  }
}
```

```
s = 2.7182818284590
x = 2.718281828459
y = 2.718281828459
s = 2.718281828459045
y = 47.718281828459
n = 2
n = 937
```

The constructor `Double(s)` creates the `Double` object `x` from the `String` object `s`, storing the numeric value 2.7182818284590 inside the object `x`. The second `println()` statement implicitly invokes the `Double.toString()` method, which converts that numeric value back into a string to be printed. The `x.doubleValue()` invocation returns the stored numeric value that is assigned to `y`. Then the `+=` operator is applied to `String` object `s` and to the double variable `y`, with very different results. For strings, "+=" means "append to," but for numbers, it means "add to."

Next, the `Double.intValue()` method truncates that stored numeric value to the integer 2 and initializes `n` with it. Finally, `Integer.parseInt("3A9",16)` returns the `int` value (in decimal form) for the integer whose hexadecimal representation is `3A9`. That is $3(16^2) + 10(16) + 9 = 937$.

Note that the `+` operator automatically converts the number `y` into its string equivalent when combined with a string, as in the call `System.out.println("y = " + y)`.

1.5 FLOW CONTROL

Java supports the `if` statement, the `if..else` statement, the `switch` statement, and the *conditional expression operator* `?..:`.

EXAMPLE 1.4 Using Conditional Statements

This program generates random integers in the range 0 to 99 and then uses conditional statements to classify them.

```
public class Ex0104
{ public static void main(String[] args)
  { int n = (int)Math.round(100*Math.random());
    System.out.println("n = " + n);
```

```
    if (n>25 && n<75) System.out.println(n + " is between 25 and 75");
    else System.out.println(n + " is not between 25 and 75");
    switch ((int)n/20)
    { case  0: System.out.println(n + " < 20");
      case  1: System.out.println(n + " < 40");
      case  2: System.out.println(n + " < 60");  break;
      case  3: System.out.println(n + " < 80");  break;
      default: System.out.println(n + " >= 80");
    }
    System.out.println(n + (n%2>0 ? " is odd" : " is even"));
  }
}
n = 19
19 is not between 25 and 75
19 < 20
19 < 40
19 < 60
19 is odd
```

Note how the absence of the `break` statement in `case 0` and `case 1` allows control to "fall through" each of those cases on to `case 2`, executing all three of those `println()` statements.

The expression `(n%2>0 ? " is odd" : " is even")` evaluates to the string `" is odd"` if the condition `n%2>0` is true (i.e., `n` is not divisible by 2); otherwise it evaluates to the string `" is even"` (when `n` is divisible by 2).

Java supports the `while` statement, the `do..while` statement, and the `for` statement.

EXAMPLE 1.5 Using Loops

This program empirically tests Gauss's Prime Number Theorem: if $p(n)$ is the number of prime numbers that are less than n, then the ratio $p(n)(\ln n)/n$ approaches 1.0 as n grows without bound. It counts the number `p(n)` of prime numbers for each odd `n` in the range 3 to 1,000,000. As $p(n)$ increases, each time it passes a multiple of 5,000, the program prints the values of n, $p(n)$, $\ln n$ (the natural logarithm) and the ratio $p(n)(\ln n)/n$, which gets close to 1.0.

```
public class Ex0105
{ public static void main(String[] args)
  { System.out.println("n\tp(n)\tln(n)\t\t\tp(n)*ln(n)/n");
    final String DASHES18="\t------------------";
    System.out.println("------\t-----" + DASHES18 + DASHES18);
    int p=1;  // p = number of primes that are <= n
    for (int n=3; n<1000000; n += 2)
    { int d=3;
      while (d<=Math.sqrt(n) && n%d>0)
        d += 2;
      if (n%d==0) continue;
      ++p;
      if (p%5000>0) continue;
      double ln=Math.log(n);
      System.out.println(n + "\t" + p + "\t" + ln + "\t" + p*ln/n);
    }
    System.out.println("------\t-----" + DASHES18 + DASHES18);
  }
}
```