CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

Joy with Java

Understanding object-oriented programming is vital for a modern computer programmer, and the Java programming language has been one of the most powerful tools available to computer programmers since its inception in 1995. It has also consistently changed since then, making it a vast and powerful resource for object-oriented programming today, finding many applications like mobile programming, Internet programming and software development.

This lucid textbook introduces the student not only to the nuances of object-oriented programming, but also to the many syntaxes and semantics of the modern Java language. Each concept of programming is explained, and then illustrated with small but effective ready-to-run programs. Important points to be noted have been emphasized and hints have been given at the end of each discussion so that programmers are careful to avoid common pitfalls. Finally, a number of practice problems taken from real world scenarios encourage the student to think in terms of problem solving, consolidating the knowledge gained.

Some key features of the book include:

- A pedagogy-rich treatment of programming in the Java language.
- Complete coverage of all syntaxes, semantics and principles.
- Important points and typical pitfalls are highlighted in simple and succinct language.
- Frequently asked questions, over and above the normal discussions, are added at the end of every chapter.
- Short annotations alongside discussions help students understand the concepts better.
- Practice problems and a large variety of questions, including MCQs, help consolidate the knowledge gained.
- Includes an online supplements package comprising solutions to the practice problems and correct outputs of each programme for students and lecture slides for instructors.

Debasis Samanta is Professor, Department of Computer Science and Engineering, at the Indian Institute of Technology Kharagpur, India. His areas of interest include human computer interaction, biometric security and data analytics.

Monalisa Sarma is Assistant Professor, Subir Chowdhury School of Quality and Reliability, at the Indian Institute of Technology Kharagpur, India. Her areas of interest include big data analytics, software testing and verification, and human reliability analysis.

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

Joy with Java

Fundamentals of Object Oriented Programming

Debasis Samanta
Monalisa Sarma

CAMBRIDGE
UNIVERSITY PRESS

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

CAMBRIDGE
UNIVERSITY PRESS

University Printing House, Cambridge CB2 8BS, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of the University of Cambridge.

It furthers the University's mission by disseminating knowledge in the pursuit of education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781009211918

© Debasis Samanta and Monalisa Sarma 2023

This publication is in copyright. Subject to statutory exception and to the provisions of relevant collective licensing agreements, no reproduction of any part may take place without the written permission of Cambridge University Press.

First published 2023

Printed in India

A catalogue record for this publication is available from the British Library

ISBN 978-1-009-21191-8 Paperback

Cambridge University Press has no responsibility for the persistence or accuracy of URLs for external or third-party internet websites referred to in this publication, and does not guarantee that any content on such websites is, or will remain, accurate or appropriate.

© in this web service Cambridge University Press & Assessment www.cambridge.org

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

To

Maa

The most admirable person in our life

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

CONTENTS

List of Figures xv
List of Tables xix
List of Programs xxiii
List of Symbols xxxiii
Preface xxxv
Acknowledgments xxxix

1 Getting Started **1**
1.0 Introduction 1
1.1 Concept of Programming 2
1.1.1 Computer and Its Working 2
1.1.2 Programming Paradigms 3
1.2 Object-Oriented Programming Paradigms 6
1.2.1 Encapsulation 6
1.2.2 Inheritance 7
1.2.3 Information Hiding 7
1.2.4 Polymorphism 8
1.3 About Java Programming 9
1.3.1 Why Java? 10
1.3.2 Java Is Unique! 14
1.3.3 Advantages of Java 15
1.3.4 Popularity of Java 18
1.4 Features of Java Programming 18
1.4.1 Core Programming 20
1.4.2 GUI Programming 20
1.4.3 Advanced Programming 20
1.5 Tools and Resources for Java Programming 21
1.5.1 Java Program Development Environment 21
1.5.2 Java Integrated Development Environment 22
1.5.3 Java Browsers 23
1.5.4 Java Program Editor 24
1.5.5 Java Tutorials 24

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

1.6 Writing Java Programs 25
1.6.1 JDK Installation 25
1.6.2 First Java Programming 29
1.6.3 First Java Applet Programming 35
Summary 37
FAQs 37
Programs for Practice 38
MCQs 39

2 Java Language Basics 41
2.0 Introduction 41
2.1 Basic Elements of the Language 42
2.1.1 Comments 42
2.1.2 Java Lexical Elements 43
2.1.3 Variables, Identifiers, and Constants 49
2.2 Primitive Data Types 53
2.2.1 Integer Data Types 53
2.2.2 Floating Point Data Types 58
2.2.3 Character Data Type 62
2.2.4 Boolean Data Type 64
2.3 Expressions and Operators 65
2.3.1 Arithmetic Operators 65
2.3.2 Bitwise Operators 73
2.3.3 Relational Operators 83
2.3.4 Boolean Logical Operators 84
2.3.5 Precedence and Associativity of Operators 88
2.3.6 Data Enumeration 90
2.4 Arrays 92
2.4.1 One-Dimensional Arrays 93
2.4.2 Multi-dimensional Arrays 96
2.5 Strings 98
2.5.1 `String` Literals 98
2.5.2 Creating String Objects 99
2.5.3 String Operations 100
2.5.4 Arrays of Strings 107
2.6 Control Structures 108
2.6.1 Sequence 108
2.6.2 Decision 110
2.6.3 Looping 124
2.6.4 Nested Control Structures 131
2.6.5 Jump Statements 132

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

Summary 136
FAQs 136
Programs for Practice 137
MCQs 140

3 Encapsulation 146
3.0 Introduction 146
3.1 Concept of Encapsulation 147
3.1.1 Structure of Class 147
3.1.2 Creating Object 150
3.1.3 Structure of a Java Program 151
3.1.4 Nested Class 157
3.1.5 Parameters Passing to Methods 158
3.1.6 Constructors 159
3.2 Polymorphism 161
3.2.1 Constructor Overloading 161
3.2.2 Method Overloading 163
3.3 Frequently Used Keywords 168
3.3.1 `static` Keyword 168
3.3.2 `this` Keyword 174
3.3.3 `final` Keyword 178
3.3.4 A Close Look at a Java Program 180
3.4 Java Run-Time Input 184
3.4.1 Command Line Input 184
3.4.2 Using `System.in.read()` 186
3.4.3 Using `Scanner` Class 187
3.4.4 Using `Console` Object 189
3.4.5 Using `BufferedReader` Class 190
3.4.6 Using `DataInputStream` Class 191
3.5 Recursion in Java 193
Summary 198
FAQs 199
Programs for Practice 199
MCQs 201

4 Inheritance 207
4.0 Introduction 207
4.1 Concept of Inheritance 208
4.1.1 Syntax of Declaring Single Inheritance 208
4.1.2 Syntax of Declaring Multi-level Inheritance 213

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

4.2 Method Overriding 214
4.2.1 Method Overriding versus Method Overloading 215
4.2.2 Constructors in Inheritance 217
4.3 Access Specification 220
4.3.1 Access Specifier `default` 220
4.3.2 Access Specifier `public` 222
4.3.3 Access Specifier `private` 223
4.3.4 Access Specifier `protected` 227
4.3.5 Summary of Access Specifiers 231
4.4 Java Special Keywords Related to Inheritance 232
4.4.1 `super` Keyword 232
4.4.2 `final` Keyword 237
4.4.3 `abstract` Keyword 241
4.5 Run-Time Polymorphism 244
4.6 Universal Super Class 248
4.6.1 Casting Object 249
4.6.2 Object Management Activities 251
4.6.3 Generic Method and Class 255
4.6.4 Methods with Variable List of Arguments 258
4.6.5 Wrapper Classes 263
Summary 272
FAQs 273
Programs for Practice 273
MCQs 277

5 Package and Interface 287
5.0 Introduction 287
5.1 Concept of Package 288
5.1.1 Built-in Java Packages 289
5.1.2 User-Defined Package 291
5.2 Accessing Packages 293
5.2.1 Name Control Mechanism 295
5.2.2 Visibility Control Mechanism 299
5.2.3 Package Sealing 301
5.3 Interfaces 302
5.3.1 Defining an Interface 304
5.3.2 Implementing Interface 306
5.3.3 Inheritance Hierarchy with Interfaces and Classes 311
5.3.4 Static Method in Interface 318

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

5.3.5 Explicitly Accessing an Interface Member without Inheritance 319
5.3.6 Default Method in an Interface 319
5.3.7 Nested Interface 320
5.3.8 Nested Class 322
5.3.9 Difference between an Abstract Class and Interface 322
5.3.10 Some Examples of User Defined Interfaces in User's Programs 323
5.3.11 Some Frequently Used Built-in Java Interfaces 334
Summary 337
FAQs 337
Programs for Practice 337
MCQs 355

6 Exception Handling 360
6.0 Introduction 360
6.1 Concept of Error and Exception 361
6.2 Types of Exceptions 366
6.2.1 Error Exceptions 367
6.2.2 Checked Exceptions 368
6.2.3 Unchecked Exceptions 371
6.3 Exception Handling Mechanisms 373
6.3.1 Simple `try-catch` Block 374
6.3.2 Single `try` Multiple `catch` Blocks 382
6.3.3 The `finally` Block 386
6.3.4 Nested `try` Construct 390
6.4 Explicit Exceptions Throwing 392
6.4.1 The `throw` 392
6.4.2 The `throws` 394
6.4.3 Difference between `throw` and `throws` 396
6.5 User Defined Exception 397
6.6 Some More Features of Exception Handling 401
6.6.1 Chained Exception 402
6.6.2 Multiple Exceptions by a Single Catch 404
6.6.3 The `rethrow` of Exception Handling 406
6.6.4 Exception Handling with `try-catch` Resource 407
Summary 411
FAQs 411
Programs for Practice 412
MCQs 417

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

7 Multithreading **425**
7.0 Introduction 425
7.1 Concept of Concurrent Programming 426
7.2 Java API for Multithreaded Programming 431
7.2.1 Thread Class 431
7.2.2 Runnable Interface 435
7.2.3 ThreadGroup Class 435
7.2.4 ThreadLocal Class 437
7.3 Creating Threads 438
7.3.1 Creating Thread with Sub Classing of the Class Thread 440
7.3.2 Creating Thread with Implementation of the Runnable interface 444
7.3.3 Thread versus Runnable Interface 449
7.4 Managing Execution of Threads 449
7.4.1 Thread Priority 450
7.4.2 Ordering of Threads 457
7.4.3 Synchronization of Threads 464
7.5 Other Aspects of Threads 483
7.5.1 Life Cycle of a Thread 484
7.5.2 Interthread Communication 489
7.6 Thread Group and Daemon 496
7.6.1 Thread Group 496
7.6.2 Daemon Thread 500
Summary 505
FAQs 505
Programs for Practice 506
MCQs 512

8 Java IO and File **519**
8.0 Introduction 519
8.1 About Java IO API 521
8.2 Streams 524
8.2.1 Standard IO Streams 526
8.2.2 Types of Streams 532
8.2.3 IO with Byte Streams 535
8.2.4 IO with Character Streams 593
8.2.5 Scanner Class 620
8.3 Files in Java 626
8.3.1 File Class 627
8.3.2 Creating Directory and Listing 632
8.3.3 Directory and File Related Information Retrieval 635

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

8.3.4 Reading and Writing with Files 637
8.3.5 FileDescriptor Class 638
8.4 Random Access File 640
8.5 Java New IO 645
Summary 645
FAQs 645
Programs for Practice 646
MCQs 647

9 Generic Programming 654
9.0 Introduction 654
9.1 Generic Method 655
9.1.1 Syntax of Defining a Generic Method 657
9.1.2 Generic Method with Variable List of Arguments 661
9.2 Generic Class 666
9.2.1 Defining Generic Classes 671
9.2.2 Generic Classes with Multiple Type Parameters 675
9.2.3 Bounded Types in Generic Class Definition 677
9.3 Wildcard in Generic Programming 686
9.3.1 Bounded Wildcard Arguments 690
9.3.2 Illustration of Bounded Wildcard Arguments 696
9.3.3 When to Use Which Wildcard 704
9.4 Applications of Generic Programming 704
9.4.1 Generic Programming for Stack Using an Array 705
9.4.2 Generic Programming for Linked List 710
9.4.3 Generic Programming for Stack Using Linked List 719
9.5 Some Facts of Generic Programming 721
9.5.1 Type Inference 722
9.5.2 Code Compatibility with Generic Programming 723
9.5.3 Java Type Erasure 725
9.5.4 Generic Constructor 726
9.5.5 Generic Interface 727
9.5.6 Generic Inheritances 729
9.5.7 Type Casting in Generics 733
9.5.8 Some Restrictions in Generic Programming 740
Summary 744
FAQs 744
Programs for Practice 744
MCQs 748

Index 755

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

FIGURES

1.1	Computing system	2
1.2	Program translation schemes	3
1.3	Function-oriented programming	4
1.4	Object-oriented paradigm	5
1.5	Encapsulation in OOP	6
1.6	Inheritance in OOP	7
1.7	Information hiding in OOP	8
1.8	Polymorphism in OOP	9
1.9	History of Java	10
1.10	Characteristics of Java language	10
1.11	Platform independent mechanism	12
1.12	Platform independent execution	13
1.13	Popularity of Java	19
1.14	Programming in Java	19
1.15	Command window in Windows	25
1.16	Terminal window in Ubuntu	25
1.17	Editing the program with NotePad++	30
1.18	Editing the program with gedit program	30
1.19	Saving the program in Windows	31
1.20	Saving the program in Ubuntu (with gedit)	31
1.21	Saved file in Windows directory “java”	32
1.22	Compiling the program (in the directory “java”) in Windows	32
1.23	The compiled file (in the directory “java”) in Windows	33
1.24	Output of the program in Windows	33
1.25	Applet on display	36

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

2.1 Type promotion rule 68
2.2 Representation of `byte` data 74
2.3 Representation of `char` data 75
2.4 Illustration of `AND` (&), `OR` (|), `XOR` (^), and `NOT` (~) operations 77
2.5 Extraction of multiple values from single data 82
2.6 Different types of arrays 92
2.7 Multidimensional array with variable length 97
2.8 `if` statement 111
2.9 `if-else` statement 114
2.10 Program flowchart to find the largest of three numbers a, b, and c 115
2.11 `Switch` construct 117
2.12 Structure and syntax of numerical for-loop 125
2.13 Structure and syntax of collection-based `for` loop 127
2.14 Structure and syntax of `while` loop 129
2.15 Structure and syntax of `do-while` loop 131

3.1 Concept of encapsulation 148
3.2 General structure of a Java program 152
3.3 Assigning references to objects 169
3.4 Class and instance variables 171
3.5 Close look at `main()` 180
3.6 Recursion concept 193
3.7 Illustration of Tower of Hanoi puzzle 197

4.1 Taxonomy of animals 208
4.2 Types of inheritances 209
4.3 Multiple single inheritance 211
4.4 Comparison between method overloading and method overriding 216
4.5 Access modifiers in Java 221
4.6 Multiple single class inheritance 245
4.7 Class hierarchy in a Java application program 249

5.1 Java APIs 289
5.2 Intra- and inter-visibility of elements across classes in packages 300

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

5.3	Implementation of an interface by classes	326
5.4	Different ways of multiple inheritances	329
6.1	Compile-time versus run-time errors	365
6.2	Hierarchy of Java exception classes	367
6.3	Basic construct of try-catch block	374
6.4	Control flow in exception handling	376
6.5	Call stack and searching call stack for the exception handler	377
6.6	Single try multiple catch block constructs	383
6.7	`try-catch-finally` construct	386
6.8	Nested `try` construct	390
7.1	Composition of a task	426
7.2	Single-CPU execution	427
7.3	Concurrent execution with multiple processing	427
7.4	Multi-user programming	428
7.5	Multi-programming execution	429
7.6	Single-threaded versus multithreaded executions	429
7.7	The declaration of java.lang.Thread class	431
7.8	A sample CPG	457
7.9	A simple CPG of an application program	458
7.10	Issue in a multithreading program with shared data	464
7.11	Deadlock situation in multithreading	478
7.12	Thread state transition diagram	485
7.13	Producer-Consumer problem	493
8.1	IO in Java programming	520
8.2	The hierarchy of classes and interfaces in `java.io`	521
8.3	A class hierarchy of `java.io` exception classes	524
8.4	Streams in `java.io`	526
8.5	Standard IO streams	527
8.6	Java IO streams	533
8.7	`java.io` classes for handling streams	534
8.8	Hierarchy of input stream related classes	535

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

8.9 Hierarchy of output stream related classes 537
8.10 IO programming with byte streams 540
8.11 Class hierarchy of `Reader` class 594
8.12 Class hierarchy of `Write` class 596
8.13 Character streams versus byte streams 599

9.1 Upper bound wildcard 690
9.2 Lower bound wildcard 693
9.3 Animal class hierarchy 697
9.4 Some examples of stacks 705
9.5 Stack data structure 706
9.6 Node of linked list 710
9.7 Logical view of a single linked list structure 710
9.8 Insertion operation on a single linked list 712
(a) Insertion in the front 712
(b) Insertion at the end 712
(c) Insertion at any position 712
9.9 Deletion operations on a single linked list 713
(a) Deletion in the front 713
(b) Deletion at the end 713
(c) Deletion at any position 713
9.10 Merging of two linked lists 714

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

TABLES

1.1	Difference between C, Pascal, and Java	14
1.2	Difference between C++ and Java	14
1.3	Difference between Python and Java	15
2.1	Java special character set	43
2.2	First 128 ASCII code values	44
2.3	Escape sequences for special characters	46
2.4	Separators in Java programming	47
2.5	Java keywords	48
2.6	The four integer types in Java	53
2.7	The two floating point date types in Java	58
2.8	Some code point values as per the Unicode format	63
2.9	Basic arithmetic operators in Java	65
2.10	`op=` operators	71
2.11	Increment and decrement operators	72
2.12	Bitwise operators	74
2.13	Bitwise basic Boolean operations	76
2.14	Bit shift operators	79
2.15	Logical operators in the form `op=`	81
2.16	Relational operators	84
2.17	Boolean logical operator	84
2.18	Order of precedence and associativity of operators in Java	89
2.19	Some string methods	106
2.20	Wrapper classes	107

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

4.1	Differences between method overloading and method overriding	216
4.2	Summary of access specifiers	231
4.3	The methods in `Object` class	252
4.4	Methods for to-and-from transformation	264
4.5	`java.lang.Integer` class	265
5.1	Built-in Java packages	290
5.2	Visibility control in classes and packages	300
5.3	Difference between abstract class and interface	323
5.4	Some frequently used interfaces in Java APIs	334
6.1	Java's checked exceptions	369
6.2	Java's unchecked exceptions under `RuntimeException` class	372
6.3	`throw` versus `throws` in Java	396
6.4	The methods defined in `Throwable` class in `java.lang`	398
7.1	Comparison among multiprocessing, multi-tasking, and multithreading executions	430
7.2	Fields in the class Thread	432
7.3	Constructors in the class Thread	432
7.4	Methods defined for the class java.lang.Thread	433
7.5	Constructors defined for the class java.lang.ThreadGroup	436
7.6	Methods defined for the class java.lang.ThreadGroup	436
7.7	Methods defined for the class java.lang.ThreadLocal	438
8.1	Interface in `java.io`	522
8.2	The classes in `java.io`	523
8.3	The exception classes defined in `java.io`	525
8.4	`InputStream` sub classes	536
8.5	Methods in `InputStream` sub classes	537
8.6	Output stream sub classes	538
8.7	The methods declared by `OutputStream` class	539
8.8	Fields defined by `BufferedArrayInputSream` class	546
8.9	Constructors in `FileInputSream` class	553
8.10	Additional methods defined by `DataInputStream` class	560

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

8.11	Additional methods defined by `DataOutputStream` class	563
8.12	Constructors defined by `PrintStream` class	568
8.13	Methods defined for `PrintStream` class	569
8.14	Format specifications for `printf()` method	572
8.15	Some methods for `ObjectInputStream` class	579
8.16	Some methods for `ObjectOutputStream` class	580
8.17	Methods in `PipedInputStream` class	589
8.18	Methods in `PipedOutputStream` class	590
8.19	Sub classes of the `Reader` class	594
8.20	Methods of the `Reader` class	595
8.21	Sub classes of the `Writer` class	597
8.22	Methods of the `Writer` class	598
8.23	Constructors of the `Scanner` class	621
8.24	Different `hasNext` methods defined in `Scanner` class	622
8.25	Different `next` methods defined in `Scanner` class	623
8.26	Fields of class `File`. All fields are `static` modifier.	627
8.27	Constructors of class `File`. All fields are `static` modifier.	627
8.28	Methods of class `File`. All fields are `public` modifier.	629
8.29	Different modes of accessing a random access file	640
8.30	Methods defined by `RandomAccessFile` class	641

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

Programs

1.1	Your first Java program	29
1.2	Your first Java applet program	35
1.3	An HTML file hosting an applet code	36
2.1	String literals	46
2.2	Illegal use of keywords	49
2.3	Using integer data type	56
2.4	Integers in octal form	57
2.5	Integers in hexadecimal form	57
2.6	Use of floating point data type	61
2.7	Character data type	63
2.8	Boolean data type	64
2.9	Arithmetic operations	66
2.10	% operation	67
2.11	Arithmetic operation `char` data	68
2.12	Type conversion	70
2.13	*op*= operations	71
2.14	++ and – – operators	73
2.15	Bitwise logical operations	77
2.16	Left shift operation	81
2.17	Application of bitwise operators	82
2.18	Examples of Boolean logical operators	85
2.19	Ternary operator example	88
2.20	`enum` data type	91
2.21	1D array example	95

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

2.22	Dealing with `String` objects	100
2.23	String merging	101
2.24	String equality	103
2.25	Sequence of execution	108
2.26	Block of statements	109
2.27	Simple `if` statement	112
2.28	Program for control flow graph	116
2.29	`switch` statement	118
2.30	`break` statement	119
2.31	`break` statement again	120
2.32	String in switch statement	121
2.33	`assert` statement	122
2.34	another example of `assert`	123
2.35	Numerical for-loop	125
2.36	Collection-based `for` loop	128
2.37	Collection-based for-loop with enum	128
2.38	`while` loop	129
2.39	`do-while` loop	131
2.40	Nested loop	132
2.41	`break` to exit a loop	133
2.42	`continue` statement	134
2.43	`break` with `continue`	135
3.1	Simple class declaration	150
3.2	Program with object	152
3.3	Declaration of class Box	154
3.4	Program with multiple classes	155
3.5	Nested class	157
3.6	Parameterized method	158
3.7	Class with constructor	160
3.8	Class with overloading constructor	162
3.9	Method overloading	163
3.10	Different signature in method overloading	165
3.11	Type promotion policy resolving ambiguity	165

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

3.12 Only a different return type is not method overloading! 166
3.13 Static methods can be overloaded 166
3.14 `main`() can be overloaded 167
3.15 Class and instance variables 170
3.16 Another example of class and instance variables 171
3.17 Example of instance and class methods 172
3.18 Static block 173
3.19 `this` for name–space resolution 174
3.20 Calling current class constructor 175
3.21 Returning current class instance 176
3.22 `this` as a method parameter 176
3.23 Invoking current class method 177
3.24 Argument in a constructor 177
3.25 Initializations of `final` variables 178
3.26 Analyzing a program 180
3.27 Formatted print 183
3.28 Command line argument: Example 1 184
3.29 Command line argument: Example 2 185
3.30 Command line argument: Example 3 186
3.31 Reading standard input 187
3.32 Reading using `Scanner` class 188
3.33 Example with `Scanner` class 188
3.34 Reading data using `Console` class 189
3.35 Reading data using `BufferedReader` class 190
3.36 Reading data using `DataInputStream` class 192
3.37 Factorial calculation with simple method 194
3.38 Factorial calculation with recursive method 194
3.39 Fibonacci sequence with recursive method 195
3.40 GCD calculation with recursive method 196
3.41 Tower of Hanoi puzzle 198

4.1 Single inheritance 209
4.2 Multiple single inheritance 211
4.3 Method overriding 214

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

4.4	Constructors in derived class	217
4.5	Order of constructors call	219
4.6	Case 1 of program execution	221
4.7	Case 3 of program execution	222
4.8	public access specifier	222
4.9	private access specifier	223
4.10	Accessing *private* member via public method	224
4.11	Accessing *private* member via inheritance	225
4.12	*private* constructor and singleton class	226
4.13	Accessing *protected* member in the classes in different directories	227
4.14	Accessing *protected* member in the same directory	229
4.15	*Protected* member in inheritance chain	229
4.16	Overriding *private* and *protected* members	230
4.17	Use of *super* with a variable	232
4.18	Use of *super* with method	233
4.19	Use of *super* with constructor	234
4.20	*super* with overloaded constructor	235
4.21	*super*() versus *this*()	236
4.22	Use of *final* variable	237
4.23	Use of *final* method	239
4.24	Use of *final* class	240
4.25	Reference *final* variable	241
4.26	*Abstract* class and method	242
4.27	Run-time polymorphism	246
4.28	Binding with data member	248
4.29	Arrays with different types of objects	250
4.30	Use of *getClass*() method	253
4.31	Use of *hashCode*() method	254
4.32	Use of *finalize*() method	255
4.33	Simple generic class	257
4.34	Generic class with multiple types	258
4.35	Varargs method using array	259
4.36	Varargs method using ellipsis	260
4.37	Overloading varargs	261

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE

Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

4.38	Varargs method with Object	263
4.39	Type conversion with wrapper classes	266
4.40	Call-by-value example	267
4.41	Wrapper class approach	267
4.42	Call by reference in Java	268
4.43	Swap test 1: a working case	269
4.44	Swap test 2: a failed case	270
4.45	Swap test 3: object wrapping method	271
5.1	Creating a package	294
5.2	Accessing a class of a package	294
5.3	Static import	295
5.4	Package location checking	296
5.5	Problem with multiple inheritance	302
5.6	Interface declaration	304
5.7	Implementing an interface	306
5.8	Partial implementation of an interface	308
5.9	Multiple implementation	309
5.10	An interface inherits another interface	312
5.11	A class inherits a class and interface	313
5.12	Interfaces with same method name	314
5.13	Interfaces with same method name	315
5.14	Interfaces with same variable name	316
5.15	Explicit reference of variables	316
5.16	Method overriding with interface	317
5.17	`Static` method in interface	318
5.18	Explicit reference of an interface's member	319
5.19	Default method in interface	320
5.20	Nested Interface	321
5.21	Constants in Java programs	324
5.22	Implementation of an interface and dynamic binding	326
5.23	Inheritance from two interfaces	330
5.24	Class implements two interfaces	331
5.25	Class extends a class and implements interfaces	332

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

5.26	`Iterator` interface	334
5.27	`Cloneable` interface	335
5.28	`Serializable` interface	335
5.29	`Serializable` interface	336
6.1	Compilation errors	362
6.2	Run-time error	363
6.3	Exception throwing	366
6.4	Checked exception	369
6.5	Remedy for checked exception	370
6.6	Unchecked exception	371
6.7	Program with exception	374
6.8	Program with exception handling mechanism	375
6.9	Call stack in JVM	376
6.10	Call stack searching by JVM	378
6.11	Robust with exception handler	379
6.12	Multiple occurrences of same exception	380
6.13	Multiple occurrences of multiple exceptions	381
6.14	Scope of variables declared inside a try block	382
6.15	Multiple exceptions in a program	384
6.16	Handling multiple exceptions in Program 6.15	384
6.17	Ordering of exception hierarchy in multiple catch blocks	385
6.18	`try-catch-finally` construct	387
6.19	`try-finally` construct	388
6.20	Execution of `finally` block is a must	388
6.21	Execution of `finally` block with loop elements	389
6.22	A simple program with multiple exceptions	391
6.23	Solution with nested `try` blocks	391
6.24	`throw` statement for explicit throwing	393
6.25	Usefulness of `throws`	394
6.26	Remedy with `throws`	395
6.27	`throws` with multiple exceptions	395
6.28	User-defined exception	399
6.29	Another user-defined exception	399

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-1-009-21191-8 — Joy with Java
Debasis Samanta , Monalisa Sarma
Frontmatter
More Information

6.30 An application program 400
6.31 Chained exception 403
6.32 Chained exception application 404
6.33 Multiple exceptions caught by a single catch block 405
6.34 Rethrow feature 406
6.35 File handling 408
6.36 `try-with-resources` in file handling 410

7.1 A simple Java program and its thread 430
7.2 Creating the main thread 438
7.3 Creating a thread and its execution 440
7.4 Execution of multiple threads 441
7.5 Execution of multiple threads of different tasks 442
7.6 Creating a thread using Runnable interface 445
7.7 Execution of multiple threads using Runnable interface 446
7.8 Anomaly in multithread programming 448
7.9 Setting priority to threads 450
7.10 Execution of threads with different priorities 452
7.11 Pre-empting and time-slicing in thread executions 455
7.12a Definitions of threads for the CPG 459
7.12b Main program for the CPG 462
7.13 Execution of program with non-serialization issue 465
7.14 Synchronization of methods 467
7.15 Partial synchronization 471
7.16 Modification of Program 7.15 with the synchronized block construct 475
7.17 Multithreading program with deadlock situation 478
7.18 Multithreading program without deadlock situation 481
7.19 Different states of a thread 485
7.20 State transition of threads 487
7.21 Controlling threads' execution using suspend() and resume() 491
7.22 Producer-Consumer problem 493
7.23 Thread group with threads 497
7.24 Thread group with threads and common run() 499
7.25 Running a daemon thread 501
7.26 Incorrect setting of a thread as daemon thread 504

© in this web service Cambridge University Press & Assessment www.cambridge.org