

Jeanne Boyarsky and Scott Selikoff

OCP

Oracle® Certified Professional Java® SE 8 Programmer II STUDY GUIDE

EXAM 1Z0-809

Covers 100% of exam objectives, including advanced class design, functional programming, concurrency, JDBC, and much more...

Includes online interactive learning environment with:

- + 3 custom practice exams
- + More than 250 electronic flashcards
- + Searchable key term glossary

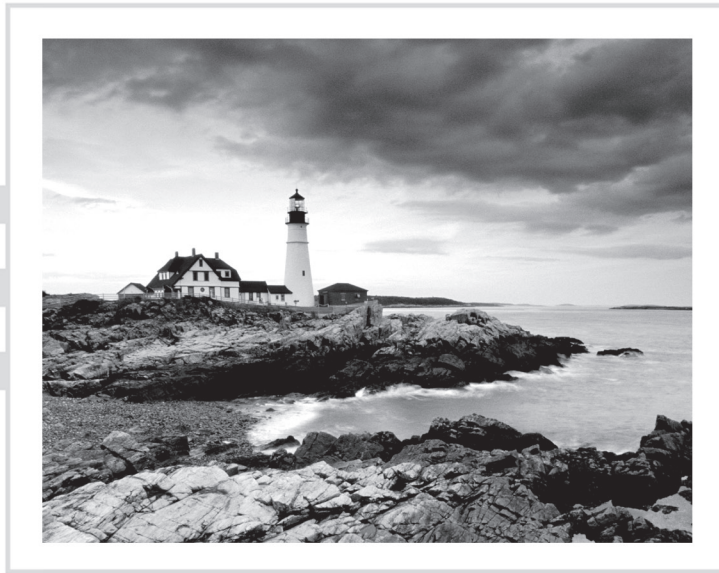


SYBEX
A Wiley Brand

OCP

Oracle® Certified Professional Java® SE 8 Programmer II

Study Guide



OCP

Oracle® Certified Professional Java® SE 8 Programmer II

Study Guide



Jeanne Boyarsky
Scott Selikoff

Senior Acquisitions Editor: Kenyon Brown
Development Editor: Gary Schwartz
Technical Editors: Ernest Friedman-Hill and Matt Dalen
Production Editor: Dassi Zeidel
Copy Editor: Linda Recktenwald
Editorial Manager: Mary Beth Wakefield
Production Manager: Kathleen Wisor
Associate Publisher: Jim Minatel
Supervising Producer: Rich Graves
Book Designers: Judy Fung and Bill Gibson
Proofreader: Josh Chase, Word One New York
Indexer: Ted Laux
Project Coordinator, Cover: Brent Savage
Cover Designer: Wiley
Cover Image: ©Getty Images Inc./Jeremy Woodhouse

Copyright © 2016 by John Wiley & Sons, Inc., Indianapolis, Indiana

Published simultaneously in Canada

ISBN: 978-1-119-06790-0

ISBN: 978-1-119-06788-7 (ebk.)

ISBN: 978-1-119-06789-4 (ebk.)

Manufactured in the United States of America

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or Web site may provide or recommendations it may make. Further, readers should be aware that Internet Web sites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at (877) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2015951679

TRADEMARKS: Wiley, the Wiley logo, and the Sybex logo are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. Oracle and Java are registered trademarks of Oracle America, Inc. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

10 9 8 7 6 5 4 3 2 1

To the programmers on FIRST robotics team FRC 694 StuyPulse
— Jeanne

*To my newborn daughters Olivia and Sophia, I love you both more and
more every day.*
— Scott

Acknowledgments

Jeanne and Scott would like to thank numerous individuals for their contribution to this book. Thank you, Gary Schwartz, for guiding us through the process and making the book better in so many ways. Thank you, Ernest Friedman-Hill, for being our Technical Editor as we wrote this book. Ernest pointed out many subtle errors in addition to the big ones. And thank you, Matt Dalen, for being our Technical Proofreader and finding the errors that managed to sneak by even Ernest. This book also wouldn't be possible without many people at Wiley, including Kenyon Brown, Dassi Zeidel, Mary Beth Wakefield, and so many others.

Jeanne would personally like to thank Chris Kreussling and Elena Felder for their feedback on early drafts of the trickier material. Elena even helped figure out a good way to explain upper bounds. Roel De Nijs reviewed the Java 8 date/time material for words that native English speakers take for granted, and he responded to lots of posts in the CodeRanch.com OCA forum on our first book. To all of the people at work and at CodeRanch.com who were so excited for me about writing this book, you made it even more exciting when we published our first book. Jeanne would like to thank the members of FIRST robotics FRC team 694 for their support. It was an awesome feeling seeing high school students pore over the book while waiting for dinner the night the hard copy arrived. Go StuyPulse! See if you can find JoeBot in this book. Finally, Jeanne would like to thank Scott for being a great co-author again.

Scott could not have reached this point without the help of a small army of people, led by his perpetually understanding wife Patti, the most wonderful mother their twin daughters could ask for. Professor Johannes Gehrke of Cornell University always believed in him and knew he would excel in his career. Jeanne's patience and striving for excellence was invaluable in creating this second book. A big thanks to fellow new father Matt Dalen, who has been a wonderful friend, sounding board, and technical reviewer over the last year. Joel McNary introduced Scott to CodeRanch.com and encouraged him to post regularly—a step that changed his life. Finally, Scott would like to thank his mother and retired teacher Barbara Selikoff, for teaching him the value of education, and his father Mark Selikoff, for instilling in him the benefits of working hard.

Finally, both Jeanne and Scott would like to give a big thank-you to the readers of our OCA 8 book. Hearing from all of you who enjoyed the book and passed the exam was great. We'd also like to thank those who pointed out errors and made suggestions for improvements in our OCA book. As of July 2015, the top three were Mushfiq Mammadov, Elena Felder, and Cédric Georges. Thank you for your attention to detail! We also would like to thank Mathias Bader, Maaïke Zijdeveld, Vincent Botteman, Edward Rance, Gabriel Jesus, Ilya Danilov, Marc ter Braak, Dominik Bauer, Saad Benbouzid, Evgeny Kapinos, Helen Colson, Alex Lord, and Kevin Abel.

About the Authors

Jeanne Boyarsky has worked as a Java developer for more than 13 years at a bank in New York City, where she develops, mentors, and conducts training. Besides being a senior moderator at CodeRanch.com in her free time, she leads the team that works on the forum's code base. Jeanne also mentors the programming division of a FIRST robotics team, where she works with students just getting started with Java.

Jeanne got her Bachelor of Arts degree in 2002 in Computer Science and her Master's in Computer Information Technology in 2005. She enjoyed getting her Master's degree in an online program while working full time. This was before online education was cool! Jeanne is also a Distinguished Toastmaster and a Scrum Master. You can find out more about Jeanne at <http://www.coderanch.com/how-to/java/BioJeanneBoyarsky>.

Scott Selikoff is a professional software consultant, author, and owner of Selikoff Solutions, LLC, which provides software development solutions to businesses in the tri-state New York City area. Skilled in a plethora of software languages and platforms, Scott specializes in database-driven systems, web-based applications, and service-oriented architectures.

A native of Toms River, New Jersey, Scott achieved his Bachelor of Arts from Cornell University in Mathematics and Computer Science in 2002, after three years of study. In 2003, he received his Master of Engineering in Computer Science, also from Cornell University.

As someone with a deep love of education, Scott has always enjoyed teaching others new concepts. He's given lectures at Cornell University and Rutgers University, as well as conferences including The Server Side Java Symposium. Scott lives in New Jersey with his loving wife, amazing twin baby girls, and two very playful dogs. You can find out more about Scott at <http://www.linkedin.com/in/selikoff>.

Jeanne and Scott are both moderators on the CodeRanch.com forums, and they can be reached there for questions and comments. They also co-author a technical blog called Down Home Country Coding at <http://www.selikoff.net>.

Contents at a Glance

<i>Introduction</i>	<i>xix</i>
<i>Assessment Test</i>	<i>xxxviii</i>
Chapter 1	Advanced Class Design 1
Chapter 2	Design Patterns and Principles 47
Chapter 3	Generics and Collections 103
Chapter 4	Functional Programming 171
Chapter 5	Dates, Strings, and Localization 233
Chapter 6	Exceptions and Assertions 283
Chapter 7	Concurrency 325
Chapter 8	IO 405
Chapter 9	NIO.2 453
Chapter 10	JDBC 505
Appendix A	Answers to Review Questions 547
Appendix B	Study Tips 575
Appendix C	Upgrading from Java 6 or Earlier 595
<i>Index</i>	<i>649</i>

Contents

Introduction

xix

Assessment Test

xxxviii

Chapter 1	Advanced Class Design	1
	Reviewing OCA Concepts	2
	Access Modifiers	2
	Overloading and Overriding	4
	Abstract Classes	5
	<i>Static</i> and <i>Final</i>	6
	Imports	6
	Using <i>instanceof</i>	7
	Understanding Virtual Method Invocation	9
	Annotating Overridden Methods	11
	Coding <i>equals</i> , <i>hashCode</i> , and <i>toString</i>	13
	<i>toString</i>	13
	<i>equals</i>	15
	<i>hashCode</i>	18
	Working with <i>Enums</i>	20
	Using <i>Enums</i> in <i>Switch</i> Statements	21
	Adding Constructors, Fields, and Methods	22
	Creating Nested Classes	24
	Member Inner Classes	25
	Local Inner Classes	27
	Anonymous Inner Classes	29
	Static Nested Classes	31
	Summary	33
	Exam Essentials	34
	Review Questions	36
Chapter 2	Design Patterns and Principles	47
	Designing an Interface	48
	Purpose of an Interface	51
	Introducing Functional Programming	52
	Defining a Functional Interface	53
	Implementing Functional Interfaces with Lambdas	55
	Applying the Predicate Interface	60
	Implementing Polymorphism	61
	Distinguishing between an Object and a Reference	63
	Casting Object References	64

Understanding Design Principles	66
Encapsulating Data	66
Creating JavaBeans	69
Applying the Is-a Relationship	71
Applying the Has-a Relationship	73
Composing Objects	74
Working with Design Patterns	75
Applying the Singleton Pattern	76
Creating Immutable Objects	82
Using the Builder Pattern	86
Creating Objects with the Factory Pattern	89
Summary	93
Exam Essentials	93
Review Questions	95
 Chapter 3	 Generics and Collections
103	103
Reviewing OCA Collections	104
Array and <i>ArrayList</i>	104
Searching and Sorting	105
Wrapper Classes and Autoboxing	106
The Diamond Operator	107
Working with Generics	108
Generic Classes	109
Generic Interfaces	112
Generic Methods	114
Interacting with Legacy Code	114
Bounds	117
Putting It All Together	122
Using Lists, Sets, Maps, and Queues	124
Common Collections Methods	125
Using the <i>List</i> Interface	127
Using the <i>Set</i> Interface	132
Using the <i>Queue</i> Interface	134
Map	138
Comparing Collection Types	140
<i>Comparator</i> vs. <i>Comparable</i>	143
<i>Comparable</i>	143
<i>Comparator</i>	146
Searching and Sorting	150
Additions in Java 8	152
Using Method References	152
Removing Conditionally	154
Updating All Elements	155

	Looping through a Collection	155
	Using New Java 8 <i>Map</i> APIs	155
	Summary	159
	Exam Essentials	161
	Review Questions	162
Chapter 4	Functional Programming	171
	Using Variables in Lambdas	172
	Working with Built-In Functional Interfaces	173
	Implementing <i>Supplier</i>	174
	Implementing <i>Consumer</i> and <i>BiConsumer</i>	175
	Implementing <i>Predicate</i> and <i>BiPredicate</i>	177
	Implementing <i>Function</i> and <i>BiFunction</i>	178
	Implementing <i>UnaryOperator</i> and <i>BinaryOperator</i>	180
	Checking Functional Interfaces	181
	Returning an <i>Optional</i>	182
	Using Streams	185
	Creating Stream Sources	188
	Using Common Terminal Operations	189
	Using Common Intermediate Operations	196
	Putting Together the Pipeline	200
	Printing a Stream	204
	Working with Primitives	205
	Creating Primitive Streams	205
	Using <i>Optional</i> with Primitive Streams	208
	Summarizing Statistics	210
	Learning the Functional Interfaces for Primitives	210
	Working with Advanced Stream Pipeline Concepts	213
	Linking Streams to the Underlying Data	213
	Chaining Optionals	214
	Collecting Results	217
	Summary	223
	Exam Essentials	224
	Review Questions	226
Chapter 5	Dates, Strings, and Localization	233
	Working with Dates and Times	234
	Creating Dates and Times	235
	Manipulating Dates and Times	241
	Working with Periods	244
	Working with Durations	247
	Accounting for Daylight Savings Time	251

	Reviewing the <i>String</i> class	253
	Adding Internationalization and Localization	255
	Picking a Locale	256
	Using a Resource Bundle	258
	Formatting Numbers	267
	Formatting Dates and Times	270
	Summary	273
	Exam Essentials	274
	Review Questions	276
Chapter 6	Exceptions and Assertions	283
	Reviewing Exceptions	284
	Exceptions Terminology	284
	Categories of Exceptions	285
	Exceptions on the OCP	286
	Try Statement	288
	Throw vs. Throws	289
	Creating Custom Exceptions	289
	Using Multi-catch	291
	Using Try-With-Resources	296
	Try-With-Resources Basics	298
	AutoCloseable	300
	Suppressed Exceptions	302
	Putting It Together	305
	Rethrowing Exceptions	305
	Working with Assertions	308
	The <i>assert</i> Statement	308
	Enabling Assertions	309
	Using Assertions	310
	Summary	314
	Exam Essentials	315
	Review Questions	316
Chapter 7	Concurrency	325
	Introducing Threads	327
	Distinguishing Thread Types	328
	Understanding Thread Concurrency	328
	Introducing <i>Runnable</i>	330
	Creating a Thread	331
	Polling with Sleep	334
	Creating Threads with the <i>ExecutorService</i>	335
	Introducing the Single-Thread Executor	335

Shutting Down a Thread Executor	337
Submitting Tasks	338
Waiting for Results	340
Scheduling Tasks	345
Increasing Concurrency with Pools	348
Synchronizing Data Access	350
Protecting Data with Atomic Classes	352
Improving Access with Synchronized Blocks	354
Synchronizing Methods	356
Understanding the Cost of Synchronization	357
Using Concurrent Collections	358
Introducing Concurrent Collections	358
Understanding Memory Consistency Errors	359
Working with Concurrent Classes	360
Obtaining Synchronized Collections	365
Working with Parallel Streams	366
Creating Parallel Streams	366
Processing Tasks in Parallel	367
Processing Parallel Reductions	372
Managing Concurrent Processes	377
Creating a <i>CyclicBarrier</i>	377
Applying the Fork/Join Framework	381
Identifying Threading Problems	387
Understanding Liveness	387
Managing Race Conditions	391
Summary	392
Exam Essentials	393
Review Questions	394

Chapter 8 IO 405

Understanding Files and Directories	406
Conceptualizing the File System	406
Introducing the File Class	407
Introducing Streams	411
Stream Fundamentals	411
Stream Nomenclature	412
Common Stream Operations	418
Working with Streams	420
The <i>FileInputStream</i> and <i>FileOutputStream</i> Classes	420
The <i>FileReader</i> and <i>FileWriter</i> classes	424
The <i>ObjectInputStream</i> and <i>ObjectOutputStream</i> Classes	426
The <i>PrintStream</i> and <i>PrintWriter</i> Classes	432
Review of Stream Classes	435

	Interacting with Users	437
	The Old Way	437
	The New Way	438
	Summary	442
	Exam Essentials	443
	Review Questions	445
Chapter 9	NIO.2	453
	Introducing NIO.2	454
	Introducing <i>Path</i>	455
	Creating Paths	456
	Interacting with Paths and Files	460
	Providing Optional Arguments	461
	Using <i>Path</i> Objects	462
	Interacting with Files	471
	Understanding File Attributes	478
	Discovering Basic File Attributes	478
	Improving Access with Views	483
	Presenting the New Stream Methods	487
	Conceptualizing Directory Walking	487
	Walking a Directory	488
	Searching a Directory	490
	Listing Directory Contents	491
	Printing File Contents	492
	Comparing Legacy <i>File</i> and NIO.2 Methods	494
	Summary	495
	Exam Essentials	495
	Review Questions	497
Chapter 10	JDBC	505
	Introducing Relational Databases and SQL	507
	Identifying the Structure of a Relational Database	508
	Writing Basic SQL Statements	510
	Introducing the Interfaces of JDBC	511
	Connecting to a Database	513
	Building a JDBC URL	513
	Getting a Database <i>Connection</i>	515
	Obtaining a Statement	518
	Choosing a <i>ResultSet</i> Type	519
	Choosing a <i>ResultSet</i> Concurrency Mode	520
	Executing a <i>Statement</i>	520
	Getting Data from a <i>ResultSet</i>	524
	Reading a <i>ResultSet</i>	524

	Getting Data for a Column	526
	Scrolling <i>ResultSet</i>	529
	Closing Database Resources	533
	Dealing with Exceptions	535
	Summary	536
	Exam Essentials	537
	Review Questions	539
Appendix A	Answers to Review Questions	547
	Chapter 1: Advanced Class Design	548
	Chapter 2: Design Patterns and Principles	550
	Chapter 3: Generics and Collections	553
	Chapter 4: Functional Programming	556
	Chapter 5: Dates, Strings, and Localization	558
	Chapter 6: Exceptions and Assertions	560
	Chapter 7: Concurrency	561
	Chapter 8: IO	565
	Chapter 9: NIO.2	568
	Chapter 10: JDBC	571
Appendix B	Study Tips	575
	Studying for the Test	576
	Creating a Study Plan	576
	Comparing Previous Exams	578
	Creating and Running Sample Applications	579
	Taking the Test	582
	Understanding the Question	582
	Taking Advantage of the Exam Software	587
	Using the Provided Writing Material	587
	Choosing the Best Answer	590
	Optimizing Your Time	591
	Getting a Good Night's Rest	593
Appendix C	Upgrading from Java 6 or Earlier	595
	Enhancing the Java Language	596
	Using the Diamond Operator	596
	Creating Literals	597
	Making Decisions with <i>Switch</i>	598
	Formatting and Parsing	603
	Using <i>DecimalFormat</i> Flags	603
	Using <i>DateFormat</i>	603
	Formatting	604
	Parsing	605

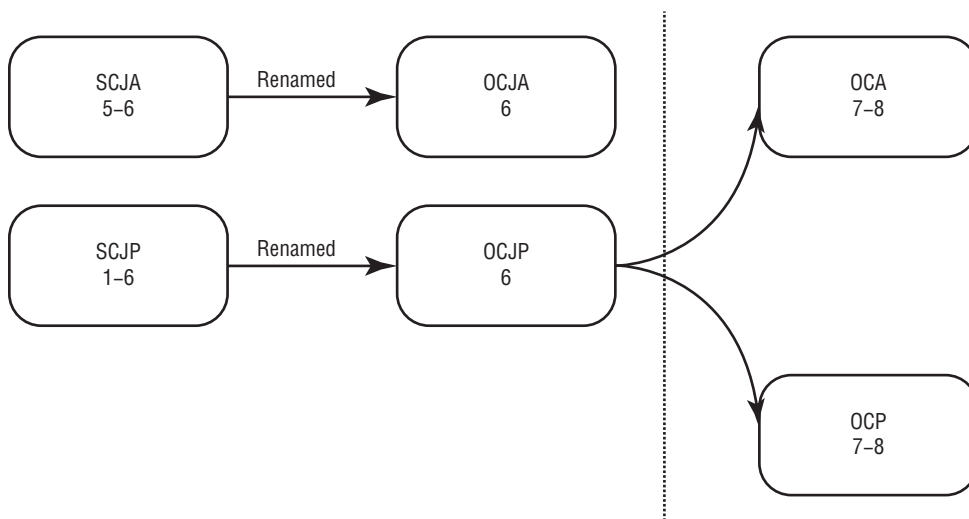
Custom Date Formats	606
Applying Locks	607
Understanding the <i>Lock</i> Framework	607
Using a <i>ReentrantLock</i>	610
Understanding Read/Write Locks	614
Using a <i>ReentrantReadWriteLock</i>	615
Working with Directories	617
Traversing Directories	618
Monitoring a Directory for Changes	625
Summary	633
Exam Essentials	633
Review Questions	635
Answers to Practice Test	645
<i>Index</i>	649

Introduction

Java recently celebrated its 20th birthday, since it was “born” in 1995. As with anything 20 years old, there is a good amount of history and variation between different versions of Java. Over the years, the certification exams have changed to cover different topics. The names of the exams have even changed. This book covers the Java 8 OCP exam along with the upgrade exams to Java 8.

If you read about the exam on the web, you may see information about the older names for the exam. The name changes are shown in Figure I.1. Here’s what happened. Back when Sun Microsystems owned Java, they used to have two exams. The SCJA (Sun Certified Java Associate) was meant for new programmers and the SCJP (Sun Certified Java Programmer) was meant for those who wanted broader knowledge. When Oracle bought Sun Microsystems, they renamed all of the exams from Sun to Oracle, giving us the OCJA (Oracle Certified Java Associate) and OCJP (Oracle Certified Java Programmer).

FIGURE I.1 Names for the exam



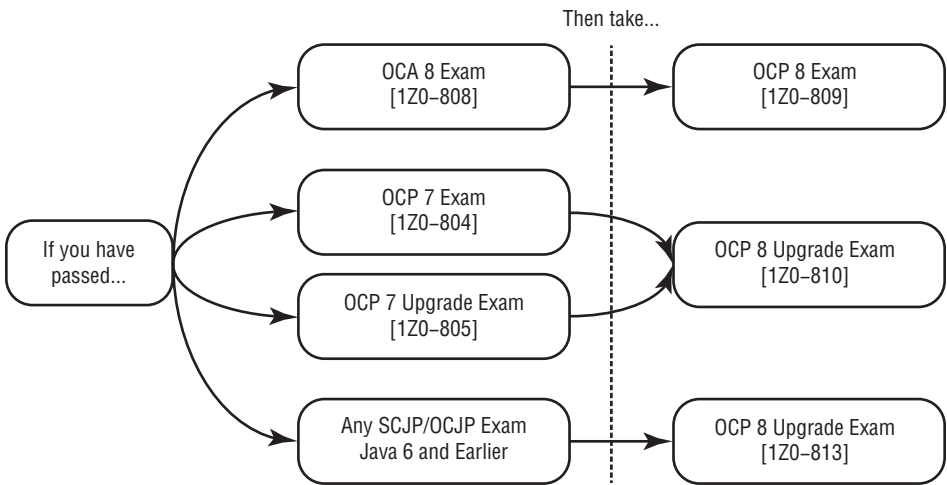
Then Oracle made two strategic decisions with Java 7. They decided to stop updating the OCJA exam. They also decided to cover more in the programmer space, and they split it into two exams. Now you first take the OCAJP (Oracle Certified Associate Java Programmer), also known as Java Programmer I or OCA. Then you take the OCPJP (Oracle Certified Professional Java Programmer), also known as Java Programmer II or OCP, and that’s what this book is about. Most people refer to the current exams as OCA 8 and OCP 8.

Oracle also has upgrade exams in case you took an older version of the SCJP or OCPJP and you want to upgrade. While most people refer to them as the Java 8 upgrade exam, there are really two exams, and you choose the correct one based on the certification you currently hold. Table I.1 describes the exams that this book covers, while Figure I.2 helps you decide what exam to take next, assuming that you have passed a prior Java certification exam. Our book is designed to help you prepare for any of these three exams, all of which result in you being OCP 8 certified.

TABLE I.1 Exams this book covers

Exam Code	Name	Who Should Take
1Z0-809	Java Programmer II	Holders of the OCA 8 certification
1Z0-810	Upgrade Java SE 7 to Java SE 8 OCP Programmer	Holders of the OCPJP 7 certification
1Z0-813	Upgrade to Java SE 8 OCP (Java SE 6 and all prior versions)	Holders of any of the following certifications: ▪ SCJP/OCJP 6 ▪ SCJP/OCJP 5 ▪ SCJP 1.4 ▪ Any older SCJP certs

FIGURE I.2 Exam prerequisites



We try to keep the history to a minimum in this book. There are some places on the exam where you need to know both an old way and a new way of doing things. When that happens, we will be sure to tell you what version of Java introduced which way. We will also let you know about topics that are not on the exam anymore, in case you see questions on them in the older free online mock exams.

The OCP Exam

All you need to do to become an Oracle Certified Professional for Java 8 is to pass an exam! That's it.

Oracle has a tendency to fiddle with the length of the exam and the passing score once it comes out. Since it's pretty much guaranteed that whatever we tell you here will become obsolete, we will instead give you a feel for the range. The OCP exam has varied between 60 and 90 questions since it was first introduced. The score to pass the exam has varied between 60 percent and 80 percent. The time allowed to take the exam has varied from two hours to two-and-a-half hours.

Oracle has a tendency to tweak the exam objectives over time as well. They do make minor additions and deletions from what is covered on the exam. For example, serialization has been added and removed from the objectives many times over the life of the OCP. As of this writing, it is on the exam.

While there will likely be minor changes to the scope of the exam, they certainly aren't a secret. We've created a book page on our blog: <http://www.selikoff.net/ocp>. If there are any changes to the topics on the exam after this book is published, we will note them there.

That book page also contains a link to the official exam page, so you can check the length and passing score that Oracle has chosen for the moment. Finally, all known errata and links to discussion can be found at <http://www.coderanch.com>.

Scheduling the Exam

Pearson VUE administers the exam, and it can be taken at any Pearson VUE testing center. To find a testing center or to register for the exam, go to <http://pearsonvue.com>. Choose IT and then Oracle. If you haven't been to the test center before, we recommend visiting in advance. Some testing centers are nice and professionally run. Others stick you in a closet with lots of people talking around you. You don't want to be taking the test with someone complaining about his or her broken laptop nearby!

At this time, you can reschedule the exam without penalty until up to 24 hours in advance. This means that you can register for a convenient time slot well in advance knowing that you can delay taking the exam if you aren't ready by that time. Rescheduling is easy and can be done completely on the Pearson VUE website. This may change, so check the rules before paying.

The Day of the Exam

When you go to take the exam, remember to bring two forms of ID, including one that is government issued. See Pearson's list of what is an acceptable ID at <http://www.pearsonvue.com/policies/1S.pdf>. Try not to bring too much extra with you, because it will not be allowed into the exam room. While you will be allowed to check your belongings, it is better to leave extra items at home or in your car.

You will not be allowed to bring paper, your phone, and so on into the exam room with you. Some centers are stricter than others. At one center, tissues were even taken away from us! Most centers allow you to keep your ID and money. They watch you taking the exam, though, so don't even think about writing notes on money. Some centers place these articles in a locker and give you the key, whereas others just throw them in an administrator's desk drawer. Suffice it to say, if you have something that you really don't want to lose, we recommend that you leave it at home.

The exam center will give you writing materials to use during the exam. These are to be used as scratch paper during the exam to figure out answers and to keep track of your thought process. The exam center will dispose of them at the end. Notice how we said "writing materials" rather than "pen and paper." Actually getting pen and paper is rare. Most centers provide a small erasable board and a dry erase marker. Before going into the exam room, we recommend testing that the marker writes and erases.

As we alluded to earlier, some exam centers are more professionally run than others, so we recommend visiting your local exam center before scheduling the exam if you have never been there before. Some exam centers also have problems keeping the temperature at a comfortable level. Regardless of whether it is winter or summer, when you take the exam, we strongly recommend that you dress in layers, such as a long-sleeve shirt or sweatshirt over a short-sleeve shirt. This way, you can add/remove layers of clothing to adjust for your own comfort.

Some exam centers are located in quiet suburban areas while others are near busy city streets with noisy traffic. Furthermore, you might get lucky and be the only person in your exam room the day you show up, or you might be unlucky and have 10 other people in the room coming and going at different times. If you are someone who gets easily distracted by noise and other people moving around, we recommend that you bring a pair of earplugs for the exam. Some exam centers will even offer you a pair of sterile earplugs if you ask. Double-check with your test administrator before using your own, so that they don't think you're trying to cheat!

While many exam centers permit bathroom breaks during the exam with permission, very few allow you to bring drinks inside. Since these exams are at least two hours long, make sure that you are well hydrated before you arrive. Just be aware that if you do need to use the facilities, your exam clock will not be paused.

Finally, if you have any issues like it being unbearably hot, cold, or noisy in your exam room, you should contact Oracle after you finish taking the exam to let them know

the quality of the particular testing center was poor. Some exam centers have shown improvement after receiving such reports.

Finding Out Your Score

In the past, you would find out whether you passed or not right after finishing the exam. Now you have to wait nervously until you can check your score online.

If you go to the Pearson VUE website, it will just show a status of “Taken” rather than your result. Oracle uses a separate system for scores. You’ll need to go to <http://certview.oracle.com> to find out whether you passed and your score. It doesn’t update immediately upon taking the test, but we haven’t heard of it taking more than an hour. In addition to your score, you’ll also see objectives for which you got a question wrong and instructions on how to get a hardcopy certificate.

At some point, you’ll get an electronic certificate, and some more time after that, you’ll receive a printed certificate. Sound vague? It is. The times reported to receive certificates vary widely.

Exam Questions

The OCP exam consists of multiple-choice questions. There are typically four to six possible answers for each question. If a question has more than one correct answer, the question specifically states exactly how many correct answers there are. This book does not do that. We say “choose all that apply” if there might be more than one correct answer to make the questions harder. This means that the questions in this book are generally harder than those on the exam. The idea is to give you more practice so that you can spot the correct answer more easily on the real exam.

You can right-click questions to cross out answers. This lets you mark answers as incorrect as you go so that you have less to think about as you read. It also helps you remember what you’ve eliminated when you go back to questions.

The exam uses two different formats for identifying line numbers. We use both approaches in this book to get you prepared. The first approach is a comment at the end of a line such as this:

```
list.stream()  
    .map(s-> s.length())    // k1  
    .forEach(System.out::println);
```

One or more answer choices will refer to k1. With this approach, imports will be provided for any class definitions. For code snippets, you can assume that all necessary

surrounding code is implied. The other approach is placing line numbers at the beginning of each line, like so:

```
4:    list.stream()  
5:        .map(s-> s.length())  
6:        .forEach(System.out::println);
```

With this approach, the line numbers often begin with numbers higher than 1. This is to indicate that you are looking at a code snippet rather than a complete class.

If you read about older versions of the exam online, you might see references to drag-and-drop questions. These questions had you do a puzzle on how to complete a piece of code. There was also a bug in the exam software that caused your answers to be lost if you reviewed them. Luckily, these are no longer on the exam.

Getting Started

We recommend reading Appendix B, “Study Tips,” before diving into the technical material in this book. Knowing how to approach studying will help you make better use of your study time.

Next, make sure that you have downloaded version 8 of the JDK. If you learned Java some time ago, you might have version 7 or even earlier. Many of the examples won’t even compile in Java 7.

Also, please check our book page to make sure that Oracle hasn’t changed the objectives. For example, if Oracle changed their mind on whether to include serialization yet again, you’d want to know that before studying. We will post any updates that you should know about at <http://www.selikoff.net/ocp>.

Getting Help

Both of the authors are moderators at CodeRanch.com, a very large and active programming forum that is very friendly toward Java beginners. It has a forum just for this exam called “SCJP/OCPJP.” It also has a forum called “Java in General” for non-exam-specific questions. As you read the book, feel free to ask your questions in either of those forums. It could be that you are having trouble compiling a class or that you are just plain confused about something. You’ll get an answer from a knowledgeable Java programmer. It might even be one of us.

Who Should Buy This Book

If you want to become Oracle Certified Java Programmer, this book is definitely for you. If you want to acquire a solid foundation in Java, and your goal is to prepare for the exam, this book is also for you. You’ll find clear explanations of the concepts you need to grasp

and plenty of help to achieve the high level of professional competency you need in order to succeed in your chosen field.

This book is intended to be understandable to anyone who knows basic Java. Since the exam has a prerequisite of the Associate exam, we assume that you have a good handle on that much Java. We don't assume that you know the Java 8–specific parts of the Associate exam, since some readers are taking the upgrade exam and are new to Java 8.

This book is for anyone from high school students to those beginning their programming journey to experienced professionals who need a review for the certification.

How This Book Is Organized

This book consists of 10 chapters plus supplementary information: a glossary, this introduction, four appendices, and the assessment test after this introduction. You might have noticed that there are more than 10 exam objectives. We split up what you need to know to make it easy to learn and remember. Each chapter begins with a list of the objectives that are covered in that chapter.

Java 8 lambdas and functional programming streams are prevalent throughout the exam and appear in questions on many topics. You have to know this topic as well as you knew loops on the OCA exam. We've made sure to use them in many chapters so that you will be ready. For example, Chapter 2 reviews functional interfaces and Chapter 3 introduces method references. Chapter 4 covers the Streams API in detail. Later chapters use “the new approach” to writing code often so that you keep using it and become fluent.

The chapters are organized as follows:

Chapter 1: Advanced Class Design This chapter covers inheritance, including abstract classes and the `final` keyword. It also discusses inner classes and enums.

Chapter 2: Design Patterns and Principles This chapter teaches you best practices for designing and writing applications that lead to code that is easier to understand and more maintainable.

Chapter 3: Generics and Collections This chapter goes beyond `ArrayList` and shows `Sets`, `Maps`, and `Queues`. It also shows new methods in Java 8 on these classes.

Chapter 4: Functional Programming This chapter explains lambdas and stream pipelines in detail. It also covers the built-in functional interfaces and the `Optional` class.

Chapter 5: Dates, Strings, and Localization This chapter shows the improved date and time classes in Java 8. It also covers more advanced `String` concepts that you need to know and how to make your application work in multiple languages.

Chapter 6: Exceptions and Assertions This chapter shows more advanced syntax for exceptions than what appears on the OCA. It also covers how to use assertions to verify the state of your program.

Chapter 7: Concurrency This chapter introduces the concept of thread management, and it teaches you how to write multi-threaded programs using the Concurrency API.

Chapter 8: IO This chapter introduces you to managing files and directories using the `java.io` API. It also instructs you on how to read and write file data using I/O streams.

Chapter 9: NIO.2 This chapter shows you how to manage files and directories using the newer NIO.2 API. It includes techniques for reading and writing file attributes, as well as traversing and searching for files using lambdas and streams.

Chapter 10: JDBC This chapter provides the basics of working with databases in Java including different types of `ResultSet`s.

Appendix A: Answers to Review Questions This appendix lists the answers to the Review Questions along with explanations.

Appendix B: Study Tips This appendix covers how to approach studying for the exam.

Appendix C: Upgrading from Java 6 or Earlier This appendix covers topics that are on the upgrade exam for those holding a Java 6 or earlier programmer certification. These topics are not on the main OCP 8 exam, nor are they on the upgrade exam for holders of the OCP 7 certification.

At the end of each chapter, you'll find a few elements that you can use to prepare for the exam:

Summary This section reviews the most important topics that were covered in the chapter, and it serves as a good review.

Exam Essentials This section summarizes the main points that were covered in the chapter. You should be able to convey the information requested.

Review Questions Each chapter concludes with at least 20 Review Questions. You should answer these questions and check your answers against the ones provided after the questions. If you can't answer at least 80 percent of these questions correctly, go back and review the chapter, or at least review those sections that seem to be giving you difficulty.



The Review Questions, Assessment Test, and other testing elements included in this book are *not* derived from the real exam questions, so don't memorize the answers to these questions and assume that doing so will enable you to pass the exam. You should learn the underlying topic, as described in the text of the book. This will let you answer the questions provided with this book *and* pass the exam. Learning the underlying topic is also the approach that will serve you best in the workplace—the ultimate goal of a certification.

To get the most out of this book, you should read each chapter from start to finish before going to the end-of-chapter elements. They are most useful for checking and

reinforcing your understanding. Even if you're already familiar with a topic, you should at least skim the chapter. There are a number of subtleties to Java that you may not encounter even when working with Java for years.

Interactive Online Learning Environment and Test Bank

The interactive online learning environment that accompanies *OCP Oracle Certified Professional SE 8 Programmer II: Exam 1Z0-809* provides a test bank with study tools to help you prepare for the certification exam, and it increases your chances of passing it the first time! The test bank includes the following:

Sample Tests All of the questions in this book are provided, including the Assessment Test, which you'll find at the end of this introduction, and the Chapter Tests, which include the Review Questions at the end of each chapter. In addition, there are three Practice Exams—180 questions in total! Use these questions to test your knowledge of the study guide material. The online test bank runs on multiple devices.

Flashcards Over 250 questions are provided in digital flashcard format (a question followed by a single correct answer). You can use the flashcards to reinforce your learning and provide last-minute test prep before the exam.

Other Study Tools Several bonus study tools are included:

Glossary A glossary of key terms from this book and their definitions is available as a fully searchable PDF.

Nashorn Materials Early drafts of the exam objectives had Nashorn on the exam for using JavaScript with Java. Since it isn't on the exam anymore, this topic isn't in the printed book. The appendix is available in the bonus contents in case you want to learn about this topic, independent of the exam.



Go to <http://sybextestbanks.wiley.com> to register and gain access to this interactive online learning environment and test bank with study tools.

Conventions Used in This Book

This book uses certain typographic styles in order to help you quickly identify important information and to avoid confusion over the meaning of words such as on-screen prompts. In particular, look for the following styles:

- *Italicized text* indicates key terms that are described at length for the first time in a chapter. (Italics are also used for emphasis.)
- A monospaced font indicates code or command-line text.
- *Italicized monospaced text* indicates a variable.

In addition to these text conventions, which can apply to individual words or entire paragraphs, a few conventions highlight segments of text:



A note indicates information that's useful or interesting. It is often something to which you should pay special attention for the exam.

Sidebars

A sidebar is like a note but longer. The information in a sidebar is useful, but it doesn't fit into the main flow of the text.



Real World Scenario

Real World Scenario

A real world scenario is a type of sidebar that describes a task or an example that's particularly grounded in the real world. This is something that is useful in the real world but is not going to show up on the exam.

OCP Exam Objectives

This book has been written to cover every objective on the OCP 8 exam along with both upgrade exams.

OCP 8 (1Z0–809)

The following table provides a breakdown of this book's exam coverage for the OCP 8 (1Z0–809) exam, showing you the chapter where each objective or sub-objective is covered:

Exam Objective	Chapter
Java Class Design	
Implement encapsulation	2
Implement inheritance including visibility modifiers and composition	1,2
Implement polymorphism	2

Exam Objective	Chapter
Override hashCode, equals, and toString methods from Object class	1
Create and use singleton classes and immutable classes	2
Develop code that uses static keyword on initialize blocks, variables, methods, and classes	1
Advanced Java Class Design	
Develop code that uses abstract classes and methods	1
Develop code that uses final keyword	1
Create inner classes including static inner class, local class, nested class, and anonymous inner class	1
Use enumerated types including methods and constructors in an enum type	1
Develop code that declares, implements and/or extends interfaces and use the @Override annotation.	1,2
Create and use Lambda expressions	2,3
Generics and Collections	
Create and use a generic class	3
Create and use ArrayList, TreeSet, TreeMap, and ArrayDeque objects	3
Use java.util.Comparator and java.lang.Comparable interfaces	3
Collections Streams and Filters	4
Iterate using forEach methods of Streams and List	3,4
Describe Stream interface and Stream pipeline	4
Filter a collection by using lambda expressions	3
Use method references with Streams	3,4
Lambda Built-in Functional Interfaces	
Use the built-in interfaces included in the java.util.function package such as Predicate, Consumer, Function, and Supplier	2,4

Exam Objective	Chapter
Develop code that uses primitive versions of functional interfaces	4
Develop code that uses binary versions of functional interfaces	4
Develop code that uses the UnaryOperator interface	4
Java Stream API	
Develop code to extract data from an object using peek() and map() methods including primitive versions of the map() method	4
Search for data by using search methods of the Stream classes including findFirst, findAny, anyMatch, allMatch, noneMatch	4
Develop code that uses the Optional class	4
Develop code that uses Stream data methods and calculation methods	4
Sort a collection using Stream API	4
Save results to a collection using the collect method and group/partition data using the Collectors class	4
Use of merge() and flatMap() methods of the Stream API	3,4
Exceptions and Assertions	
Use try-catch and throw statements	6
Use catch, multi-catch, and finally clauses	6
Use Autoclose resources with a try-with-resources statement	6
Create custom exceptions and AutoCloseable resources	6
Test invariants by using assertions	6
Use Java SE 8 Date/Time API	
Create and manage date-based and time-based events including a combination of date and time into a single object using LocalDate, LocalTime, LocalDateTime, Instant, Period, and Duration	5
Work with dates and times across time zones and manage changes resulting from daylight savings including Format date and times values	5

Exam Objective	Chapter
Define and create and manage date-based and time-based events using Instant, Period, Duration, and TemporalUnit	5
Java I/O Fundamentals	
Read and write data from the console	8
Use BufferedReader, BufferedWriter, File, FileReader, FileWriter, FileInputStream, FileOutputStream, ObjectOutputStream, ObjectInputStream, and PrintWriter in the java.io package.	8
Java File I/O (NIO.2)	
Use Path interface to operate on file and directory paths	9
Use Files class to check, read, delete, copy, move, manage metadata of a file or directory	9
Use Stream API with NIO.2	9
Java Concurrency	
Create worker threads using Runnable, Callable, and use an ExecutorService to concurrently execute tasks	7
Identify potential threading problems among deadlock, starvation, livelock, and race conditions	7
Use synchronized keyword and java.util.concurrent.atomic package to control the order of thread execution	7
Use java.util.concurrent collections and classes including CyclicBarrier and CopyOnWriteArrayList	7
Use parallel Fork/Join Framework	7
Use parallel Streams including reduction, decomposition, merging processes, pipelines and performance.	7
Building Database Applications with JDBC	
Describe the interfaces that make up the core of the JDBC API including the Driver, Connection, Statement, and ResultSet interfaces and their relationship to provider implementations	10
Identify the components required to connect to a database using the DriverManager class (including the JDBC URL)	10

Exam Objective	Chapter
Submit queries and read results from the database (including creating statements, returning result sets, iterating through the results, and properly closing result sets, statements, and connections)	10
Localization	
Read and set the locale by using the Locale object	5
Create and read a Properties file	5
Build a resource bundle for each locale and load a resource bundle in an application	5

Upgrade from Java 7 (1Z0–810)

This table shows the chapter where each objective or sub-objective is covered for the upgrade exam from Java 7 to Java 8 OCP (1Z0–810):

Exam Objective	Chapter
Lambda Expressions	
Describe and develop code that uses Java inner classes, including nested class, static class, local class, and anonymous classes	1
Describe and write functional interfaces	2
Describe a lambda expression; refactor the code that uses an anonymous inner class to use a lambda expression; describe type inference and target typing	4
Using Built-in Lambda Types	
Describe the interfaces of the java.util.function package	4
Develop code that uses the Function interface	4
Develop code that uses the Consumer interface	4
Develop code that uses the Supplier interface	4
Develop code that uses the UnaryOperator interface	4
Develop code that uses the Predicate interface	4

Exam Objective	Chapter
Develop code that uses the primitive and binary variations of the base interfaces of the <code>java.util.function</code> package	4
Develop code that uses a method reference, including refactoring a lambda expression to a method reference	3,4
Filtering Collections with Lambdas	
Develop code that iterates a collection by using the <code>forEach()</code> method and method chaining	3
Describe the <code>Stream</code> interface and pipelines	4
Filter a collection by using lambda expressions	3
Identify the lambda operations that are lazy	4
Collection Operations with Lambda	
Develop code to extract data from an object by using the <code>map()</code> method	3
Search for data by using methods such as <code>findFirst()</code> , <code>findAny()</code> , <code>anyMatch()</code> , <code>allMatch()</code> , and <code>noneMatch()</code>	4
Describe the unique characteristics of the <code>Optional</code> class	4
Perform calculations by using Java Stream methods, such as <code>count()</code> , <code>max()</code> , <code>min()</code> , <code>average()</code> , and <code>sum()</code>	4
Sort a collection by using lambda expressions	4
Develop code that uses the <code>Stream.collect()</code> method and <code>Collectors</code> class methods, such as <code>averagingDouble()</code> , <code>groupingBy()</code> , <code>joining()</code> , and <code>partitioningBy()</code>	4
Parallel Streams	
Develop code that uses parallel streams	7
Implement decomposition and reduction in streams	4,7
Lambda Cookbook	
Develop code that uses Java SE 8 collection improvements, including <code>Collection.removeIf()</code> , <code>List.replaceAll()</code> , <code>Map.computeIfAbsent()</code> , and <code>Map.computeIfPresent()</code> methods	3

Exam Objective	Chapter
Develop code that uses Java SE 8 I/O improvements, including Files.find(), Files.walk(), and lines() methods	9
Use the merge() and flatMap() methods	3,4
Develop code that creates a stream by using the Arrays.stream() and IntStream.range() methods	
Method Enhancements	
Add static methods to interfaces	2
Define and use a default method of an interface and describe the inheritance rules for the default method	2
Use Java SE 8 Date/Time API	
Create and manage date- and time-based events, including a combination of date and time in a single object, by using LocalDate, LocalTime, LocalDateTime, Instant, Period, and Duration	5
Work with dates and times across time zones and manage changes resulting from daylight savings, including Format date and times values	5
Define, create, and manage date- and time-based events using Instant, Period, Duration, and TemporalUnit	5

Upgrade from Java 6 or lower (1Z0–813)

If you are studying for the 1Z0–813 exam, you *must* read Appendix C. It covers topics that are on your exam but not the other exams covered by the book.

This table shows the chapter where each objective or sub-objective is covered for the upgrade exam from a Java 6 or lower certification to Java 8 OCP (1Z0–813).

Exam Objective	Chapter
Language Enhancements	
Develop code that uses String objects in the switch statement, binary literals, and numeric literals, including underscores in literals	Appendix C
Develop code that uses try-with-resources statements, including using classes that implement the AutoCloseable interface	6

Exam Objective	Chapter
Develop code that handles multiple Exception types in a single catch block	6
Use static and default methods of an interface including inheritance rules for a default method	2
Concurrency	
Use collections from the <code>java.util.concurrent</code> package with a focus on the advantages over and differences from the traditional <code>java.util</code> collections	7
Use <code>Lock</code> , <code>ReadWriteLock</code> , and <code>ReentrantLock</code> classes in the <code>java.util.concurrent.locks</code> and <code>java.util.concurrent.atomic</code> packages to support lock-free thread-safe programming on single variables	Appendix C
Use <code>Executor</code> , <code>ExecutorService</code> , <code>Executors</code> , <code>Callable</code> , and <code>Future</code> to execute tasks using thread pools	7
Use the parallel Fork/Join Framework	7
Localization	
Describe the advantages of localizing an application and developing code that defines, reads, and sets the locale with a <code>Locale</code> object	5
Build a resource bundle for a locale and call a resource bundle from an application	5
Create and manage date- and time-based events by using <code>LocalDate</code> , <code>LocalTime</code> , <code>LocalDateTime</code> , <code>Instant</code> , <code>Period</code> , and <code>Duration</code> , including a combination of date and time in a single object	5
Format dates, numbers, and currency values for localization with the <code>NumberFormat</code> and <code>DateFormat</code> classes, including number and date format patterns	5, Appendix C
Work with dates and times across time zones and manage changes resulting from daylight savings	5
Java File I/O (NIO.2)	
Operate on file and directory paths by using the <code>Path</code> class	9
Check, delete, copy, or move a file or directory by using the <code>Files</code> class	9
Recursively access a directory tree by using the <code>DirectoryStream</code> and <code>FileVisitor</code> interfaces	Appendix C

Exam Objective	Chapter
Find a file by using the PathMatcher interface, and use Java SE 8 I/O improvements, including Files.find(), Files.walk(), and lines() methods	9
Observe the changes in a directory by using the WatchService interface	Appendix C
Lambda	
Define and write functional interfaces and describe the interfaces of the java.util.function package	3,4
Describe a lambda expression; refactor the code that uses an anonymous inner class to use a lambda expression; describe type inference and target typing	3,4
Develop code that uses the built-in interfaces included in the java.util.function package, such as Function, Consumer, Supplier, UnaryOperator, Predicate, and Optional APIs, including the primitive and binary variations of the interfaces	4
Develop code that uses a method reference, including refactoring a lambda expression to a method reference	3
Java Collections	
Develop code that uses diamond with generic declarations	Appendix C
Develop code that iterates a collection, filters a collection, and sorts a collection by using lambda expressions	3
Search for data by using methods, such as findFirst(), findAny(), anyMatch(), allMatch(), and noneMatch()	4
Perform calculations on Java Streams by using count, max, min, average, and sum methods and save results to a collection by using the collect method and Collector class, including the averagingDouble, groupingBy, joining, partitioningBy methods	4
Develop code that uses Java SE 8 collection improvements, including the Collection.removeIf(), List.replaceAll(), Map.computeIfAbsent(), and Map.computeIfPresent() methods	3
Develop code that uses the merge(), flatMap(), and map() methods on Java Streams	4

Exam Objective	Chapter
Java Streams	
Describe the Stream interface and pipelines; create a stream by using the Arrays.stream() and IntStream.range() methods; identify the lambda operations that are lazy	4
Develop code that uses parallel streams, including decomposition operation and reduction operation in streams	7

Assessment Test

1. What is the result of executing the following application? (Choose all that apply.)

```
import java.util.concurrent.*;
import java.util.stream.*;
public class BabyPandaBathManager {
    public static void await(CyclicBarrier cb) {
        try {
            cb.await();
        } catch (InterruptedException | BrokenBarrierException e) {
            // Handle exception
        }
    }
    public static void main(String[] args) {
        final CyclicBarrier cb = new CyclicBarrier(3,()-> System.out.
            println("Clean!")); // u1
        ExecutorService service = Executors.newScheduledThreadPool(2);
        IntStream.iterate(1, i-> 1) // u2
            .limit(12)
            .forEach(i-> service.submit( // u3
                ()-> await(cb))); // u4
        service.shutdown();
    }
}
```

- A. It outputs Clean! at least once.
 - B. It outputs Clean! four times.
 - C. The code will not compile because of line u1.
 - D. The code will not compile because of line u2.
 - E. The code will not compile because of line u3.
 - F. The code will not compile because of line u4.
 - G. It compiles but throws an exception at runtime.
 - H. It compiles but waits forever at runtime.
2. What is the result of the following program?
- ```
1: public abstract class Message {
2: public String recipient;
3: public abstract final void sendMessage();
4: public static void main(String[] args) {
```