Appendix B

Exam Topics: Java SE 8 Programmer I

Please note that all information presented in this appendix was valid as of January 2016. It is imperative to visit the exam website mentioned in this appendix regularly while you are preparing for the exam, as Oracle is known to change the exam objectives intermittently.

Exam Name: *Java SE 8 Programmer I*	Duration: *150 minutes*
Exam Number: *1Z0-808*	Number of Questions: *77*
Associated Certification:	Passing Score: *65%*
Oracle Certified Associate,	Exam Price: *$245*
Java SE 8 Programmer	

The *Java SE 8 Programmer I* exam is required to qualify as *Oracle Certified Associate, Java SE 8 Programmer* (OCAJP8). Pertinent information about this exam can be found at:

```
http://education.oracle.com/pls/web_prod-plq-dad/
db_pages.getpage?page_id=5001&get_params=p_exam_id:1Z0-808
```

The web page also provides links to the exam topics defined by Oracle. The topics are organized in *sections*, and each section is *reproduced verbatim* in this appendix. For each section, we have provided references to where in the book the exam topics in the section are covered. In addition, the extensive index at the end of the book can be used to look up specific topics.

General information about taking the exam can be found in Appendix A. Oracle has also specified certain important assumptions about the exam questions, which can found in Appendix A, p. 511.

Section 1: Java Basics		
[1.1]	Define the scope of variables	*§2.4, p. 44* *§4.4, p. 114*
[1.2]	Define the structure of a Java class	*§1.2, p. 2* *§3.1, p. 48*
[1.3]	Create executable Java applications with a main method; run a Java program from the command line; including console output	*§1.10, p. 16* *§4.3, p. 107*
[1.4]	Import other Java packages to make them accessible in your code	*§4.2, p. 97*
[1.5]	Compare and contrast the features and components of Java such as: platform independence, object orientation, encapsulation, etc.	*§1.12, p. 21*
Section 2: Working with Java Data Types		
[2.1]	Declare and initialize variables (including casting of primitive data types)	*§2.3, p. 40* *§2.4, p. 42* *§5.6, p. 158*
[2.2]	Differentiate between object reference variables and primitive variables	*§2.3, p. 40*
[2.3]	Know how to read or write to object fields	*§1.3, p. 4*
[2.4]	Explain an Object's Lifecycle (creation, “dereference by reassignment” and garbage collection)	*§9.1, p. 384* *§9.2, p. 384* *§9.3, p. 386*
[2.5]	Develop code that uses wrapper classes such as Boolean, Double, and Integer	*§8.3, p. 346*
Section 3: Using Operators and Decision Constructs		
[3.1]	Use Java operators; including parentheses to override operator precedence	*§5.3, p. 150* *§5.4, p. 152* *§5.6–§5.17, p. 158*
[3.2]	Test equality between Strings and other objects using == and equals()	*§5.12, p. 181* *§8.4, p. 357* *§8.4, p. 363*
[3.3]	Create if and if/else and ternary constructs	*§5.16, p. 194* *§6.2, p. 200*
[3.4]	Use a switch statement	*§6.2, p. 203*

Section 4: Creating and Using Arrays		
[4.1]	Declare, instantiate, initialize, and use a one-dimensional array	*§3.4, p. 58*
[4.2]	Declare, instantiate, initialize, and use multi-dimensional array	*§3.4, p. 64*
Section 5: Using Loop Constructs		
[5.1]	Create and use while loops	*§6.3, p. 213*
[5.2]	Create and use for loops including the enhanced for loop	*§6.3, p. 215* *§6.3, p. 217*
[5.3]	Create and use do/while loops	*§6.3, p. 214*
[5.4]	Compare loop constructs	*§6.3, p. 213*
[5.5]	Use break and continue	*§6.4, p. 219*
Section 6: Working with Methods and Encapsulation		
[6.1]	Create methods with arguments and return values; including overloaded methods	*§3.2, p. 49* *§3.5, p. 72* *§6.4, p. 224* *§7.2, p. 273*
[6.2]	Apply the static keyword to methods and fields	*§4.8, p. 132*
[6.3]	Create and overload constructors; including impact on default constructors	*§3.3, p. 53*
[6.4]	Apply access modifiers	*§4.5, p. 118* *§4.7, p. 123*
[6.5]	Apply encapsulation principles to a class	*§7.14, p. 334*
[6.6]	Determine the effect upon object references and primitive values when they are passed into methods that change the values	*§3.5, p. 72* *§5.2, p. 147*

Section 7: Working with Inheritance		
[7.1]	Describe inheritance and its benefits	*§7.1, p. 264* *§7.13, p. 331*
[7.2]	Develop code that demonstrates the use of polymorphism; including overriding and object type versus reference type	*§7.2, p. 268* *§7.12, p. 329*
[7.3]	Determine when casting is necessary	*§7.11, p. 320*
[7.4]	Use super and this to access objects and constructors	*§3.2, p. 50* *§7.4, p. 276* *§7.5, p. 282*
[7.5]	Use abstract classes and interfaces	*§4.6, p. 120* *§7.6, p. 290*
Section 8: Handling Exceptions		
[8.1]	Differentiate among checked exceptions, unchecked exceptions, and Errors	*§6.6, p. 237*
[8.2]	Create a try-catch block and determine how exceptions alter normal program flow	*§6.7, p. 238*
[8.3]	Describe the advantages of exception handling	*§6.10, p. 254*
[8.4]	Create and invoke a method that throws an exception	*§6.8, p. 249* *§6.9, p. 251*
[8.5]	Recognize common exception classes (such as NullPointerException, ArithmeticException, ArrayIndexOutOfBoundsException, ClassCastException)	*§6.6, p. 233*
Section 9: Working with Selected Classes from the Java API		
[9.1]	Manipulate data using the StringBuilder class and its methods	*§8.5, p. 374*
[9.2]	Creating and manipulating Strings	*§8.4, p. 357*
[9.3]	Create and manipulate calendar data using classes from java.time.LocalDateTime, java.time.LocalDate, java.time.LocalTime, java.time.format.DateTimeFormatter, java.time.Period	*§11.1, p. 462* *§11.2, p. 462* *§11.3, p. 476* *§11.4, p. 486*
[9.4]	Declare and use an ArrayList of a given type	*§10.1, p. 414*
[9.5]	Write a simple Lambda expression that consumes a Lambda Predicate expression	*§10.2, p. 433*