1Z0-809

Java SE 8 Programmer II

Exam Summary – Syllabus – Questions

Table of Contents

Introduction to 1Z0-809 Exam on Java SE 8 Programmer II

You can use this document to collect all the information about Java SE 8 Programmer II (1Z0-809) certification. The Oracle 1Z0-809 certification is mainly targeted to those candidates who are from enterprise software development background and want to flourish their career with Oracle Certified Professional Java SE 8 Programmer (OCPJP) credential. The Java SE 8 Programmer II certification exam validates your understanding of the Oracle Java technology and sets the stage for your future progression.

Oracle 1Z0-809 Certification Details:

Exam Name	Java SE 8 Programmer II
Exam Code	1Z0-809
Exam Product Version	Java SE
Exam Price	USD $245 (Pricing may vary by country or by localized currency)
Duration	150 Mins
Number of Questions	85
Passing Score	65%
Validated Against	This exam has been validated against SE 8.
Format	Multiple Choice
Recommended Training	Java SE 8 Programming
Schedule Exam	Pearson VUE - Oracle
Recommended Practice	**1Z0-809 Online Practice Exam**

Oracle 1Z0-809 Exam Syllabus:

Java Class Design	- Implement encapsulation - Implement inheritance including visibility modifiers and composition - Implement polymorphism - Override hashCode, equals, and toString methods from Object class - Create and use singleton classes and immutable classes - Develop code that uses static keyword on initialize blocks, variables, methods, and classes
Advanced Java Class Design	- Develop code that uses abstract classes and methods - Develop code that uses final keyword - Create inner classes including static inner class, local class, nested class, and anonymous inner class - Use enumerated types including methods, and constructors in an enum type - Develop code that declares, implements and/or extends interfaces and use the atOverride annotation. - Create and use Lambda expressions
Generics and Collections	- Create and use a generic class - Create and use ArrayList, TreeSet, TreeMap, and ArrayDeque objects - Use java.util.Comparator and java.lang.Comparable interfaces - Collections Streams and Filters - Iterate using forEach methods of Streams and List - Describe Stream interface and Stream pipeline - Filter a collection by using lambda expressions - Use method references with Streams
Lambda Built-in Functional Interfaces	- Use the built-in interfaces included in the java.util.function package such as Predicate, Consumer, Function, and Supplier - Develop code that uses primitive versions of functional interfaces - Develop code that uses binary versions of functional interfaces - Develop code that uses the UnaryOperator interface
Java Stream API	- Develop code to extract data from an object using peek() and map() methods including primitive versions of the map() method - Search for data by using search methods of the Stream classes including findFirst, findAny, anyMatch, allMatch, noneMatch - Develop code that uses the Optional class - Develop code that uses Stream data methods and calculation methods - Sort a collection using Stream API - Save results to a collection using the collect method and group/partition data using the Collectors class - Use flatMap() methods in the Stream API

Exceptions and Assertions	- Use try-catch and throw statements - Use catch, multi-catch, and finally clauses - Use Autoclose resources with a try-with-resources statement - Create custom exceptions and Auto-closeable resources - Test invariants by using assertions
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 - Work with dates and times across timezones and manage changes resulting from daylight savings including Format date and times values - Define and create and manage date-based and time-based events using Instant, Period, Duration, and TemporalUnit
Java I/O Fundamentals	- Read and write data from the console - Use BufferedReader, BufferedWriter, File, FileReader, FileWriter, FileInputStream, FileOutputStream, ObjectOutputStream, ObjectInputStream, and PrintWriter in the java.iopackage.
Java File I/O (NIO.2)	- Use Path interface to operate on file and directory paths - Use Files class to check, read, delete, copy, move, manage metadata of a file or directory - Use Stream API with NIO.2
Java Concurrency	- Create worker threads using Runnable, Callable and use an ExecutorService to concurrently execute tasks - Identify potential threading problems among deadlock, starvation, livelock, and race conditions - Use synchronized keyword and java.util.concurrent.atomic package to control the order of thread execution - Use java.util.concurrent collections and classes including CyclicBarrier and CopyOnWriteArrayList - Use parallel Fork/Join Framework - Use parallel Streams including reduction, decomposition, merging processes, pipelines and performance.
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 - Identify the components required to connect to a database using the DriverManager class including the JDBC URL - 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
Localization	- Read and set the locale by using the Locale object - Create and read a Properties file - Build a resource bundle for each locale and load a resource bundle in an application

1Z0-809 Sample Questions:

01. Which of the following will print current time?
a) System.out.print(new LocalTime().now());
b) System.out.print(new LocalTime());
c) System.out.print(LocalTime.now());
d) System.out.print(LocalTime.today());
e) None of the above.

02. Which of the following can be used to create a MonthDay instance?
a) new MonthDay(10,28);
b) MonthDay.get(10,28);
c) MonthDay.of(10,28);
d) MonthDay.of(28,Month.OCTOBER);
e) None of above.

03. Which of the following statements are true, assuming a and b are String objects?
a) If a.equals(b) is true, a.hashCode() == b.hashCode() is always true.
b) If a.equals(b) is true, a.hashCode() == b.hashCode() is sometimes but not always true.
c) If a.equals(b) is false, a.hashCode() == b.hashCode() can never be true.
d) If a.equals(b) is false, a.hashCode() == b.hashCode() can sometimes be true.

04. What are some of the properties of using the singleton pattern?
a) Singleton object can be replaced with encapsulated setter method.
b) Requires constructor of singleton class to be private.
c) Singleton object must be named instance.
d) Singleton object may be private or protected.
e) Ensures that there is only one instance of an object in memory.
f) Requires a public static method to retrieve the instance of the singleton.

05. Which of the answer choices make sense to implement with a lambda?
a) Comparable interface
b) Comparator interface
c) remove method on a Collection
d) removeAll method on a Collection
e) removeIf method on a Collection

06. Which of the following is equivalent to this code?
UnaryOperator<Integer> u = x -> x * x;
a) BiFunction<Integer> f = x -> x*x;
b) BiFunction<Integer, Integer> f = x -> x*x;
c) BinaryOperator<Integer, Integer> f = x -> x*x;
d) Function<Integer> f = x -> x*x;
e) Function f = x -> x*x;
f) None of the above

07. Which of the following are stored in a Period object?
a) Year
b) Month

c) Day
d) Hour
e) Minute
f) Second

08. Fill in the blank: A class that implements _____ may be in a try-with-resource statement.
a) AutoCloseable
b) Closeable
c) Exception
d) RuntimeException
e) Serializable

09. Assuming / is the root directory, which of the following are true statements?
a) /home/parrot is an absolute path.
b) /home/parrot is a directory.
c) /home/parrot is a relative path.
d) The path pointed to from a File object must exist.
e) The parent of the path pointed to by a File object must exist.

10. Which NIO.2 method is most similar to the legacy java.io.File.listFiles() method?
a) Path.listFiles()
b) Files.walk()
c) Files.find()
d) Files.files()
e) Files.list()
f) Files.lines()

Answers to 1Z0-809 Exam Questions:

QUESTION: 01 Answer: c	QUESTION: 02 Answer: c	QUESTION: 03 Answer: a	QUESTION: 04 Answer: b, e, f	QUESTION: 05 Answer: b, e
QUESTION: 06 Answer: e	QUESTION: 07 Answer: a, b, c	QUESTION: 08 Answer: a, b	QUESTION: 09 Answer: a	QUESTION: 10 Answer: e

Note: If you find any typo or data entry error in these sample questions, we request you to update us by commenting on this page or write an email on feedback@oraclestudy.com