Java SE 8 Programmer I Course (40 Hours)

About this Course

Java is the foundation for virtually every type of networked application and is the global standard for developing and delivering mobile applications, games, web-based content and enterprise software. Installed on more than 5 billion devices and with more than 9 million developers worldwide, Java enables you to efficiently develop and deploy exciting applications and services. With comprehensive tools, a mature ecosystem and robust performance, Java delivers application portability across different computing environments.

The Java Associate course provides an ideal entry into an application development or software project management career using Java technologies. Throughout the course, you will learn basic knowledge of Object-Oriented Concepts, UML representation of OO concepts, the Java programming language, and general knowledge of Java Platforms and Technologies using the latest version of Java.

The course is ideal for:

- Entry-level programmers.
- Existing programmers new to the Java language.

How is the Course Structured?

Java Basics

- Installing Eclipse
- Define the scope of variables
- Define the structure of a Java class
- Create executable Java applications with a main method; run a Java program from the command line; including console output.
- Import other Java packages to make them accessible in your code
- Compare and contrast the features and components of Java such as: platform independence, object orientation, encapsulation, etc.

Working With Java Data Types

- Declare and initialize variables (including casting of primitive data types)
- Differentiate between object reference variables and primitive variables
- Know how to read or write to object fields
- Explain an Object's Lifecycle (creation, "dereference by reassignment" and garbage collection)
- Develop code that uses wrapper classes such as Boolean, Double, and Integer.

Using Operators and Decision Constructs

- Use Java operators; including parentheses to override operator precedence
- Test equality between Strings and other objects using == and equals ()

- Create if and if/else and ternary constructs
- Use a switch statement

Creating and Using Arrays

- Declare, instantiate, initialize and use a one-dimensional array
- Declare, instantiate, initialize and use multi-dimensional array

Using Loop Constructs

- Create and use while loops
- Create and use for loops including the enhanced for loop
- Create and use do/while loops
- Compare loop constructs
- Use break and continue

Working with Methods and Encapsulation

- Create methods with arguments and return values; including overloaded methods
- Apply the static keyword to methods and fields
- Create and overload constructors; including impact on default constructors
- Apply access modifiers
- Apply encapsulation principles to a class
- Determine the effect upon object references and primitive values when they are passed into methods that change the values

Working with Inheritance

- Describe inheritance and its benefits
- Develop code that demonstrates the use of polymorphism; including overriding and object type versus reference type
- Determine when casting is necessary
- Use super and this to access objects and constructors
- Use abstract classes and interfaces

Handling Exceptions

- Differentiate among checked exceptions, unchecked exceptions, and Errors
- Create a try-catch block and determine how exceptions alter normal program flow
- Describe the advantages of Exception handling
- Create and invoke a method that throws an exception
- "Recognize common exception classes (such as NullPointerException, ArithmeticExcpetion, ArrayIndexOutOfBoundsException, ClassCastException)"

Working with Selected classes from the Java API

- Manipulate data using the StringBuilder class and its methods
- Creating and manipulating Strings
- Create and manipulate calendar data using classes from java.time.LocalDateTime, java.time.LocalDate, java.time.LocalTime, java.time.format.DateTimeFormatter, java.time.Period
- Declare and use an ArrayList of a given type
- Write a simple Lambda expression that consumes a Lambda Predicate expression