

Course Code	
Course Name	Core Java, Junit, Apache Maven
Duration (in days)	7.5
Pre-requisites	None

Proficiency Level	OOPS using Java with Data Structures
Target Audience	Campus Hires

Learning Outcome

At the end of the program, participants will be able to learn:

- Design and build robust, object-oriented applications.
- Organize complex data using Java collections.
- Deal with exceptions in a Java program.
- Appreciate new features in Java 8 such Lambdas & Streams API.
- Work with Data structures such as Stacks & Queues.
- Use appropriate sorting & searching technique in each situation.
- Access Oracle / MySQL database using JDBC.
- Use the JUnit testing framework and become fluent in writing assertions to verify correct program behaviour.
- Solid understanding of Garbage Collection and its Algorithms.

Day-wise Session Plan

Day	Unit	Objective(s)	Hours
1	Introduction	[REFRESHER] <ul style="list-style-type: none"> • Introduction to Java • JVM Architecture • Installation of Java • Configuring SDE Eclipse • Understanding how JRE and JVM works 	1
1	Wrapper Classes	[REFRESHER] <ul style="list-style-type: none"> • Class Structure • Java Keywords • Primitive data types • Creating primitive variables • Using operators • Using if-else and switch statements • Iterating with loops: while, do-while, for, enhanced for • Wrapper Classes and Autoboxing concepts 	1

1	Object Oriented Programmings	[REFRESHER] <ul style="list-style-type: none"> Working on Constructors Achieving Encapsulation Code Reusability via Inheritance Achieving Polymorphism Working on methods of java.lang.Object class Object Casting Passing Objects as Arguments Abstraction via Abstract Classes and Interfaces Diamond Problem using Interfaces. Creating Static Classes and Static Methods 	1
1	Arrays API	[REFRESHER] <ul style="list-style-type: none"> Single-Dimensional Array Multi-Dimensional Arrays Array of Objects Arrays utility class 	0.5
1	String Classes	[REFRESHER] <ul style="list-style-type: none"> String Class StringBuffer class StringBuilder class Introduction to Regex (Regular Expression) 	0.5
1	Working with Exceptions	[REFRESHER] <ul style="list-style-type: none"> Defining the purpose of Java exceptions Using the try and throw Statements. Using the catch, multi-catch, and finally clauses Autoclose resources with a try-with-resources statement Recognizing common exception classes and categories Creating custom exceptions 	1
1,2	Design Patterns	<ul style="list-style-type: none"> Singleton Design Pattern Factory Design Pattern Abstract Factory Design Pattern Builder Design Pattern Template Method Design Pattern Bridge Design Pattern Proxy Design Pattern Creating Immutable classes 	8

2	Java 8 Features	<ul style="list-style-type: none"> • Motivation for Lambdas • Lambda Expression Overview • Lambda Expressions and Functional Interfaces • Method References 	2
2,3	Working with the Date/Time API	<ul style="list-style-type: none"> • The Date/Time API (JSR 310) • Use of LocalDate/LocalTime/LocalDateTime Instances • Dates and Times across Time Zones • Formatting Dates 	4
3	Generic Classes	<ul style="list-style-type: none"> • Inheritance with Generic Types • Wildcard Parameter Types (bounded & unbounded) 	2
3,4	Collections Framework	<p>[REFRESHER]</p> <ul style="list-style-type: none"> • Collections Overview • Using the type inference diamond to create an object • Creating a collection by using generics • Implementing an ArrayList, LinkedList, Vector • Implementing a HashSet, TreeSet • Implementing a HashMap, TreeMap, Hashtable • Ordering collections – Comparable & Comparator • Utility Classes - Collections and Arrays • Stream API • java.util.function Package – Predicate, Consumer, Function, and Supplier • Stream Operations • Stream map method • FindFirst and Lazy Operations • Sorting a Stream 	6
4	Working with Stacks	<ul style="list-style-type: none"> • Introducing the Stack • Stack Using Arrays - Is Empty, Is Full, and Size • Stack Using Arrays - Push • Stack Using Arrays - Pop • Stack Using Arrays - Peek • Stack Using Linked Lists - Push • Stack Using Linked Lists - Pop and Peek 	4

4,5	Working with Queues	<ul style="list-style-type: none"> Introducing the Queue Queue Using Arrays - Is Full, Is Empty, and Size Queue Using Arrays - Enqueue, Dequeue Queue Using Arrays - O(N) Enqueue and Peek Circular Queue - Is Full, Is Empty, and Enqueue Circular Queue - Dequeue and Peek Queue Using Linked Lists - Enqueue, Dequeue and Peek 	4
5	Sorting & Searching Algorithms	<ul style="list-style-type: none"> Sorting Algorithms and Trade-offs Implementing Selection Sort Implementing Bubble Sort Implementing Insertion Sort Implementing Merge Sort Implementing Quick Sort Implementing Linear Search Implementing Binary Search 	8
6	Input & Output Streams	[REFRESHER] <ul style="list-style-type: none"> Describing the basics of input and output in Java Read and write data from the console. Using streams to read and write files. Writing and read objects using Serializable. 	1
6	Multi-Threading	[REFRESHER] <ul style="list-style-type: none"> Describing operating system task scheduling Creating worker threads using Runnable and Callable Thread Life Cycle Synchronization in Threads InterThread Communication Avoiding common multithreading pitfalls Schedulers, Timers 	2
6	JDBC API	[REFRESHER] <ul style="list-style-type: none"> Defining the layout of the JDBC API Connecting to a database by using a JDBC driver Submitting queries and get results from the Database. 	1

		<ul style="list-style-type: none"> Specifying JDBC driver information externally Performing CRUD operations using the JDBC API – Statement & PreparedStatement Metadata using ResultSetMetaData and DatabaseMetaData 	
6	Performing Unit Testing using JUnit4	[REFRESHER] <ul style="list-style-type: none"> Overview Tests, Assertions, and Fixtures Writing and Running Tests Assertions Test Fixtures, @Before and @After, @BeforeClass and @AfterClass Test cases for Exception and Timeout Parameterized Tests Test Suites 	1
7	Deployment and Application Enhancement	<ul style="list-style-type: none"> Packages Creating a JAR File Client/Server Architecture Running a JAR File from the Command Line 	2
7	Memory Management	<ul style="list-style-type: none"> Garbage Collection API Make an object eligible for GC. Requesting JVM to run Garbage Collector How and when to use Finalization Types of JVM Garbage Collectors 	2
7	Maven Repositories & Dependency Management	[REFRESHER] <ul style="list-style-type: none"> Setting up Maven Navigating a Project Structure The POM File Building, Testing, and Packaging a Project Overview of Dependency Management and Repository Maven Lifecycles and Phases Configuring and Using Plugins Developing a Basic Plugin Built in Archetypes Generating a Web Project Maven Build Profiles Working with Build Profiles 	4
8	Running Tests & Generating Reports	<ul style="list-style-type: none"> Overview of Testing 	4

		<ul style="list-style-type: none"> • Adding Test-scoped Dependencies • Running Tests • Generating Test Reports • Using the Site Lifecycle • Customized Site Configuration • Using the Javadoc Plugin • Integrating Maven with Eclipse 	
		Total	60