Core Java

Day 1: Java Basics, IDE, and Control Flow

Java Introduction (9:00 AM – 11:00 AM)

- Introduction to Java and its platform-independent nature
- Role of JDK, JRE, and JVM in Java program execution
- JVM architecture: class loader, memory areas, execution engine
- Structure of a Java class and execution flow from source to bytecode

Data Types, Variables & Arrays (11:15 AM – 1:00 PM)

- Java primitive and reference data types
- Variable declaration, initialization, and type casting
- Scope and lifetime of variables in different blocks
- One-dimensional and multidimensional arrays with syntax and usage

IDE Setup – IntelliJ & Operators (2:00 PM – 3:30 PM)

- Installing and setting up IntelliJ IDEA Community Edition
- Creating and running a Java project in IntelliJ
- Using arithmetic, relational, logical, and bitwise operators
- Operator precedence and use of unary, ternary operators

Control Statements & Lab (3:45 PM – 5:00 PM)

- Using if, if-else, nested if-else, and switch-case statements
- Looping with for, while, do-while and handling loop control
- Using break, continue, and return with practical examples
- Hands-on lab: Write a Java program using loops, arrays, and conditionals

Day 2: Strings and Object-Oriented Programming (OOPs)

Strings in Java (9:00 AM – 11:00 AM)

- Working with String in Java
- Common String methods – length(), charAt(), equals(), concat(), substring()
- String comparison, case conversion, and trimming methods

OOPs Concepts (11:15 AM – 1:00 PM)

- Understanding classes, objects, attributes, and methods
- Accessing members using object references
- Constructor types – default, parameterized

- Use of ‘this’ keyword

OOPs Concepts (2:00 PM – 3:30 PM)

- Inheritance – IS-A and HAS-A relationships
- Method overloading vs method overriding
- Access modifiers – public, private, protected, and default
- Static members and blocks in classes

Hands-on Lab: OOP Implementation (3:45 PM – 5:00 PM)

- Create Java classes using constructors, methods, and access modifiers
- Implement inheritance and method overloading/overriding
- Practical class-object problems using IntelliJ IDEA

Day 3: Advanced OOPs, Exceptions, and Wrappers

Encapsulation, Abstraction, Polymorphism (9:00 AM – 11:00 AM)

- Data hiding and encapsulation using private members and getters/setters
- Abstraction using abstract classes and interfaces
- Compile-time vs run-time polymorphism in Java
- Interface implementation and use in design

Wrapper Classes & Nested Classes (11:15 AM – 1:00 PM)

- Introduction to wrapper classes
- Working with Integer, Double, Character, etc.
- Static and non-static nested classes
- Anonymous, local, and inner class syntax and use cases

Packages, Interfaces & Cloning (2:00 PM – 3:30 PM)

- Creating and importing user-defined packages
- Defining and using interfaces and multiple inheritance
- Object cloning using clone()
- Final keyword with classes, methods, and variables

Hands-on Lab: Interfaces & Inheritance (3:45 PM – 5:00 PM)

- Develop Java programs using interfaces, nested classes, and polymorphism
- Implement a real-world problem using encapsulation and abstraction

Day 4: Exceptions, Multithreading & Java Collections

Exception Handling (9:00 AM – 11:00 AM)

- Types of exceptions – checked vs unchecked
- Try-catch, multiple catch blocks, nested try
- Finally block, throw, throws and exception propagation
- Custom exception creation and best practices

Multithreading Concepts (11:15 AM – 1:00 PM)

- Thread life cycle and creating threads (Thread class & Runnable)
- Thread control methods – sleep(), join(), yield(), setPriority()
- Daemon threads, thread groups, and thread pools
- Executor framework basics and garbage collection

Collections Framework (2:00 PM – 3:30 PM)

- List, Set, Map – overview and differences
- Working with ArrayList, LinkedList, HashSet, HashMap
- Using Iterator, ListIterator, Enumeration for traversal
- Comparator vs Comparable with sorting collections

Hands-on Lab: Exception & Collections (3:45 PM – 5:00 PM)

- Implement multithreading using Runnable and ExecutorService
- Create programs using ArrayList, HashSet, and sorting with Comparator

Day 5: Java 8+, JDBC, Streams, and Project

Java 8 Features & Streams (9:00 AM – 11:00 AM)

- Lambda expressions and method references
- Functional interfaces and default/static methods in interfaces
- Stream API – filter, map, reduce, forEach, collect
- Optional class, Base64 encoding/decoding, Collectors class

JDBC Concepts (11:15 AM – 1:00 PM)

- JDBC overview and connectivity architecture
- Types of JDBC drivers and setup in IntelliJ
- JDBC classes and interfaces: DriverManager, Connection, Statement
- Performing CRUD operations with a MySQL

Java 11 & 17 Features + Memory Model (2:00 PM – 5:00 PM – Final Project)

- Enhancements in Java 11 and 17 – var, HttpClient, switch expressions
- Java Memory Model – Heap, Stack, Method Area, Garbage Collection
- Overview of Concurrency API with basics of CompletableFuture
- **Final Project Lab:** Build a Java console-based application using OOP, Collections, Exception Handling, Multithreading, and JDBC

Unit Testing

Day 1: JUnit 5 Fundamentals and Testing Techniques

JUnit 5 Introduction & Architecture (9:00 AM – 11:00 AM)

- Overview of unit testing and role of JUnit in Java
- Understanding JUnit 5 architecture – Jupiter, Platform, Vintage
- Difference between JUnit 4 and JUnit 5
- Environment setup in IntelliJ IDEA with JUnit 5 dependencies

Creating Test Cases & Annotations (11:15 AM – 1:00 PM)

- Writing basic test cases using @Test in JUnit 5
- Using key annotations – @BeforeAll, @AfterAll, @BeforeEach, @AfterEach
- Using assertEquals, assertTrue, assertNotNull, and other assertions

Advanced Features in JUnit 5 (2:00 PM – 3:30 PM)

- Using assumptions with assumeTrue, assumeFalse, assumingThat
- Testing exceptions using assertThrows and assertDoesNotThrow
- Using @RepeatedTest and @ParameterizedTest for test repetition

Tagging, Filtering & Hands-on Lab (3:45 PM – 5:00 PM)

- Tagging test cases using @Tag for selective execution
- Filtering tests using includeTags and excludeTags
- Combining tests using TestSuite and hierarchical test structure
- Hands-on lab: Create a reusable test suite with multiple test classes

Day 2: Mockito Framework and Test Integration

Mockito Overview & Setup (9:00 AM – 11:00 AM)

- Introduction to mocking and need for Mockito
- Setting up Mockito in IntelliJ using Maven
- Creating mock objects using mock() and @Mock annotation
- Injecting mocks using @InjectMocks and initializing with MockitoAnnotations

JUnit & Mockito Integration (11:15 AM – 1:00 PM)

- Writing test cases with both JUnit 5 and Mockito
- Adding behavior using when().thenReturn() pattern
- Verifying behavior using verify() and times() methods
- Testing multiple calls and varying responses using thenReturn() chaining

Advanced Mockito Features (2:00 PM – 3:30 PM)

- Handling exceptions using thenThrow() and doThrow()
- Creating ordered verifications using InOrder object
- Using callbacks with Answer interface to handle dynamic stubbing
- Resetting mocks with Mockito.reset() for test reusability

BDD, Timeouts & Hands-on Lab (3:45 PM – 5:00 PM)

- Writing tests in Behavior Driven Development (BDD) style using BDDMockito
- Using timeouts with verify(mock, timeout(100)) to ensure timely execution
- Final hands-on lab: Build a layered app (Service + DAO) and write test cases using JUnit + Mockito
- Recap and best practices for writing clean and maintainable unit tests

Spring, Spring MVC & Spring Boot

Day 1: Spring Core Fundamentals & Configuration

Spring Framework Overview (9:00 AM – 11:00 AM)

- Overview of the Spring ecosystem and architecture
- Installing and configuring Spring in IntelliJ IDEA
- Exploring major Spring modules and their use cases
- Understanding Inversion of Control (IoC) vs Dependency Injection (DI)

Spring Bean Management (11:15 AM – 1:00 PM)

- IoC Containers – BeanFactory and ApplicationContext
- Creating and configuring beans using XML and annotations
- Bean scopes, lifecycle, and destruction callbacks
- Injecting collections, inner beans, and using autowiring strategies

Spring Java & Annotation-Based Configuration (2:00 PM – 3:30 PM)

- Differences between XML-based and annotation-based configuration
- Using @Component, @Autowired, @Qualifier annotations
- Java-based configuration using @Configuration and @Bean
- Working with @Value for property injection and custom events in Spring

AOP & JDBC Integration in Spring (3:45 PM – 5:00 PM)

- Introduction to Aspect-Oriented Programming in Spring
- Spring JDBC template for database interaction
- Declarative transaction management using @Transactional

Day 2: Spring MVC Deep-Dive

Spring MVC Architecture (9:00 AM – 11:00 AM)

- Introduction to Spring Web MVC and DispatcherServlet role
- Understanding the complete flow of MVC architecture
- Controllers, ViewResolvers, and HandlerMappings
- Using @RequestMapping and @RequestParam annotations

Advanced MVC Concepts (11:15 AM – 1:00 PM)

- Difference between @Controller and @RestController
- Exception handling using @ControllerAdvice and @ExceptionHandler
- Understanding stereotypes: @Component, @Service, @Repository

Spring REST Integration & Data JPA (2:00 PM – 3:30 PM)

- Creating REST controllers using Spring MVC
- Calling Spring REST services from MVC controllers
- Introduction to Spring Data JPA and repository interfaces
- CRUD operations using JpaRepository and custom query methods

Security & Logging Basics (3:45 PM – 5:00 PM)

- Introduction to Spring Security and authentication
- OAuth vs OAuth2 overview and differences
- Logging with Log4j and integration in Spring
- Hands-on lab: Build secure Spring MVC app with basic REST

Day 3: Spring Boot Essentials & REST API Development

Spring Boot Introduction (9:00 AM – 11:00 AM)

- Spring Boot architecture and key features
- Difference between Spring, Spring MVC, and Spring Boot
- Using Spring Initializr to bootstrap projects in IntelliJ IDEA
- Spring Boot Hello World example using Maven

Spring Boot Components & Auto Configuration (11:15 AM – 1:00 PM)

- Key annotations: @SpringBootApplication, @EnableAutoConfiguration
- Understanding Spring Boot Starters and Starter Parent
- Application.properties vs YAML configuration
- Externalized configuration using @ConfigurationProperties

Spring Boot RESTful Services (2:00 PM – 3:30 PM)

- Creating REST APIs with Spring Boot controllers
- Input validation using @Valid and BindingResult
- Global exception handling using @ControllerAdvice
- Adding authentication using Spring Boot Security and OAuth2

Spring Boot Advanced Config (3:45 PM – 5:00 PM)

- Profile-based configuration
- Logging using SLF4j, Logback, and Log4j2 in Spring Boot
- Caching using Spring Boot with EhCache
- Hands-on lab: Create CRUD REST API with Spring Boot & Security

Day 4: REST Architecture, Testing, and Swagger Integration

REST API Concepts (9:00 AM – 11:00 AM)

- Overview of REST Architecture & REST maturity model
- HTTP methods, status codes, headers
- Designing resource-based REST APIs
- REST API error/response handling best practices

Testing REST and OAuth2 Security (11:15 AM – 1:00 PM)

- Writing integration tests for REST APIs using Spring Boot
- Using TestRestTemplate and MockMvc for testing
- Securing REST APIs with Spring Security + OAuth2
- Implementing token-based authorization and role-based access

Swagger Tools Introduction (2:00 PM – 3:30 PM)

- Overview of SwaggerHub, Swagger UI, and Swagger Editor
- Generating server stub using Swagger Codegen
- Auto-generating Swagger docs from Spring Boot annotations
- Testing APIs using Swagger Inspector

Swagger REST Integration & Hands-on Lab (3:45 PM – 5:00 PM)

- Integrating Swagger with existing REST APIs
- Creating and testing API specs in Swagger Editor
- Generating REST clients from Swagger spec
- Final hands-on project: Secure REST API with full Swagger integration

MongoDB with Spring Boot

Day 1: MongoDB with Spring Boot & Cloud Deployment

MongoDB Fundamentals (9:00 AM – 11:00 AM)

- Introduction to MongoDB and its NoSQL advantages
- Installing MongoDB on local and using MongoDB Atlas
- Understanding Documents, Collections, and BSON
- MongoDB Compass for GUI-based operations

CRUD Operations & Spring Boot Integration (11:15 AM – 1:00 PM)

- Performing CRUD operations using Mongo Shell
- Connecting Spring Boot app to MongoDB
- Defining documents and repositories using `@Document` and `MongoRepository`
- Implementing CRUD operations in Spring Boot

Advanced Mongo Operations & Cloud Deployment (2:00 PM – 3:30 PM)

- Performing advanced queries using Spring Data MongoDB
- Query annotations and custom repository methods
- Deploying Spring Boot MongoDB app to Heroku or Render Cloud Platform
- Connecting MongoDB Atlas cluster to deployed app

Hands-on Lab (3:45 PM – 5:00 PM)

- Deploying Mongo-based Spring Boot app to Heroku or Render Cloud Platform
- Lab: Build and deploy a full MongoDB CRUD app using Spring Boot + MongoDB Atlas

ORM & Hibernate with JPA

Day 1: ORM & Hibernate with Spring Boot

ORM Fundamentals & Hibernate Basics (9:00 AM – 11:00 AM)

- Introduction to Object-Relational Mapping (ORM)
- Overview of ORM frameworks and why Hibernate
- Hibernate architecture and core components
- Setting up JPA/Hibernate in Spring boot Project

Hibernate Configuration & CRUD (11:15 AM – 1:00 PM)

- Configuration and annotation-based setup
- Session, Transaction, and SessionFactory concepts
- Writing CRUD operations with JPA/Hibernate

Spring Data JPA Integration (2:00 PM – 3:30 PM)

- Using Spring Boot with Spring Data JPA
- Creating `@Entity` classes and repositories
- Configuration of DataSource and application.properties
- Spring Boot JDBC integration examples

Hands-on Lab: Spring Boot + Hibernate (3:45 PM – 5:00 PM)

- Build a Spring Boot application using Hibernate ORM
- Perform CRUD using JPA Repository.

Kafka

Day 1: Apache Kafka Essentials to Cluster Setup

Kafka Introduction & Installation (9:00 AM – 11:00 AM)

- What is Kafka and why it's used for real-time messaging
- Kafka architecture and message flow
- Core components: Broker, Producer, Consumer, Zookeeper
- Installing and configuring Kafka locally

Message Publishing & Consuming (11:15 AM – 1:00 PM)

- Creating and managing topics, partitions, offsets
- Kafka producers and message keys
- Consumers and consumer groups
- Kafka delivery semantics and offset management

Kafka Internals and Replication (2:00 PM – 3:30 PM)

- Topic replication, and leader election
- Kafka physical storage and message retention
- Partitioning strategies

Kafka with Docker + Lab (3:45 PM – 5:00 PM)

- Setting up a Kafka cluster using Docker
- Lab: Create Kafka Producer and Consumer using Spring Boot + Dockerized Kafka