edForce

**Technical Training Outline**

---

**Day 1 – Programming Fundamentals**

1. Java platform overview
    - JDK vs JRE vs JVM
    - Java program execution flow
    - bytecode & class loading (high level)
2. Java program structure
    - class, main method
    - packages (intro)
    - naming conventions
3. Variables & Data types
    - primitive vs non-primitive
    - default values
    - type casting (implicit, explicit)
4. Operators
    - arithmetic
    - relational
    - logical
    - assignment
    - increment/decrement
5. Control statements
    - if, if-else, else-if
    - switch (basic)
    - for, while, do-while
    - break & continue
6. Input handling
    - Scanner class
    - command line arguments

**Functions & Modular Programming**

7. Method basics
    - Method declaration and definition
    - Method calling
    - Method signature
8. Parameters and arguments
    - Single and multiple parameters
    - Passing primitive data types
    - Order of parameters
9. Return values
    - void vs non-void methods
    - Returning computed values
    - Returning boolean for validations
10. Validation logic using methods
    - Input validation methods
11. Static vs non-static methods
    - When to use static utility methods
    - Calling static and instance methods
12. Method overloading (intro)
    - Same method name, different parameter list
    - Real-world usage in utilities
13. Scope of variables
    - Local variables
    - Method parameters

---

**Day 2 – OOP Basics in Java**

1. Why Object-Oriented design
2. Class and Object
    - Class as blueprint
    - Relationship between class and object
3. Structure of a Java class
    - Fields (attributes)
    - Methods (behaviour)
4. Creating objects
    - new keyword
    - Object reference variables
5. Encapsulation
    - private data members
    - public methods
    - data hiding
6. Getter and Setter methods
    - naming conventions
    - controlled access to data
7. Constructor (intro only)
    - default constructor
    - parameterized constructor

**Inheritance & Polymorphism**

8. Why inheritance in object oriented design
    - reuse of common behavior
    - removing duplicate code from Account
9. Inheritance basics
    - extends keyword
    - IS-A relationship

10. Method overriding
    - rules for overriding
    - @Override annotation
11. Runtime polymorphism
    - parent reference, child object

      dynamic method dispatch
12. Abstract class
    - abstract vs concrete methods
    - when to use abstract class
13. Designing base Account class
    - common properties
    - common operations

---

**Day 3 – (1 hr)**

**Exception handling**

1. Exception hierarchy (intro)
2. try – catch – finally
3. Multiple catch blocks
4. throw keyword
5. Creating custom exception class (intro)

**Programming Fundamentals and OOPs Assessment**

---

**Day 4 –**

**Collections**

1. Collection framework overview
2. List vs Set vs Map
3. ArrayList
4. Map (HashMap)

**Searching Algorithms**

1. Why searching algorithms in real systems
    - relation to in-memory collections
2. Linear Search
    - algorithm steps
    - best, worst, average case
3. Binary Search
    - pre-condition (sorted data)
    - iterative vs recursive idea
    - limitation on List of objects
4. Big-O notation (intro level)
    - O(1), O(n), O(log n)
    - search performance comparison
5. Searching in object collections
    - searching by field (accountNumber)
    - equals vs custom comparison

---

**Day 5 – (4 hrs)**

**Data Structures**

1. Why data structures are required in real applications
    - difference between algorithm and data structure
    - choosing the right structure for a use case
2. Stack – LIFO principle
    - push, pop, peek
    - real use cases (undo / rollback)
3. Queue – FIFO principle
    - offer, poll, peek
    - real use cases (request processing)
4. LinkedList as a data structure
    - node based structure
    - singly vs doubly linked list (concept only)
5. Java implementation overview
    - Stack class
    - Queue interface
    - LinkedList class as List and Queue

6. Performance idea (high level)
    - insertion and deletion in linked list vs array list

---

**Day 6 – JAVA  and Algorithms Assessments**

---

**Day 7 – File Handling**

1. Why file handling in real applications
    - difference between in-memory data and persistent data
    - limitation of collections only approach
2. Java File I/O overview
    - File vs Path (basic awareness)
    - absolute vs relative path
3. Byte streams vs character streams
    - InputStream / OutputStream
    - Reader / Writer
4. Buffered streams
    - BufferedReader
    - BufferedWriter
5. Object serialization
    - ObjectOutputStream
    - ObjectInputStream
    - Serializable interface
6. transient keyword
    - when not to persist fields
7. Exception handling in file operations
    - try-with-resources

---

**Day 8 – REST API & Spring Boot**

1. REST Architecture Basics
    - What is REST?
    - REST vs SOAP (quick comparison)
    - Resource, URI, representation
    - Stateless communication
    - Client–server model

- Idempotent vs non-idempotent methods

2. HTTP Essentials for REST

- HTTP methods: GET, POST, PUT, PATCH, DELETE
- HTTP status codes (2xx, 4xx, 5xx)
- Headers, body, content-type
- JSON as REST payload

3 Spring Boot REST Introduction

- Why Spring Boot for REST services
- Spring Boot auto-configuration
- Embedded server (Tomcat)
- Project structure
- Hands-on
- Create Spring Boot REST project
- First REST endpoint
- Test using Postman / curl

4. REST Controllers

- @RestController
- @RequestMapping
- @GetMapping, @PostMapping, etc.
- Request Parameters Handling
- @PathVariable
- @RequestParam
- @RequestBody

5. Response Handling

- Returning POJO → JSON
- ResponseEntity
- Hands-on - Build CRUD endpoints for a sample resource (Employee / Student)
- Use Path and Query parameters

---

**Day 9 – HTML Basics & CSS (4 hrs)**

1. Role of HTML in web applications
    - front-end vs back-end (connect with your Java backend track later)
2. Basic HTML structure
    - <!DOCTYPE>
    - <html>, <head>, <body>
3. Common HTML tags
    - headings, paragraphs, div, span

4. Forms in HTML
    - <form> tag
    - action & method (GET vs POST – concept only)
    - form submission flow
5. Input controls
    - text, number, email, password
    - radio, checkbox
    - select & option
    - textarea
    - button / submit
6. Table elements
    - table, thead, tbody, tr, th, td
7. CSS Basics
8. CSS Selectors
9. CSS images, bacxkground, font, text etc attributes
10. Linking of CSS with HTML Pages

---

**Day 10 – Data Structures and HTML Assessment + web page creation**

---

**Day 11 – Java + HTML Integration**

1. End-to-end data flow in web applications
    - HTML form → server → Java object → service layer
2. HTTP request and response (concept level)
    - request parameters
    - response generation
3. Form submission flow
    - GET vs POST (practical difference)
4. Mapping form data to Java objects
    - request parameters to fields
    - DTO / model object idea
5. Separation of layers
    - controller (handler)
    - service
    - model

**SQL Fundamentals**

1. What is a database and RDBMS
2. Table and row concept
3. Primary key
4. Foreign key
5. Relationship between tables
    - one-to-many (Account → Transactions)
6. CRUD operations
    - INSERT
    - SELECT
    - UPDATE
    - DELETE

---

**Day 12 – JDBC Integration (2 hrs)**

**Sub-Topics (Concepts)**

1. Why JDBC in Java applications
    - role of JDBC between Java and RDBMS
    - how it replaces file-based persistence (connect to Day-8)
2. JDBC architecture
    - Driver
    - Connection
    - Statement vs PreparedStatement
    - ResultSet
3. JDBC workflow
4. PreparedStatement
    - why not Statement

---

**Day 13 – Basics of DevOps**

**Introduction to DevOps & Culture**

1. What is DevOps?
2. Problems with traditional SDLC (Dev vs Ops silos)
3. DevOps goals:
4. Faster releases

5. Stability
6. Automation
7. Collaboration
8. DevOps lifecycle overview
9. Plan → Code → Build → Test → Release → Deploy → Operate → Monitor
10. DevOps culture and principles
11. CALMS (Culture, Automation, Lean, Measurement, Sharing)
12. DevOps vs Agile vs CI/CD

**Version Control & Collaboration (Git Basics)**

13. Role of version control in DevOps
14. Centralized vs Distributed VCS
15. Git architecture (working tree, staging, repository)
16. Basic Git workflow
17. Common commands
18. clone, add, commit, push, pull, branch, merge
19. GitHub / GitLab overview
20. Branching strategy (simple feature branch flow)

**Continuous Integration (CI)**

21. What is Continuous Integration
22. Why CI is requireq
23. CI pipeline concepts
24. Build automation basics
25. Automated testing in CI
26. CI tools overview
27. GitHub Actions
28. Jenkins
29. GitLab CI
30. Pipeline stages and triggers

---

**Module 4 – Containerization & Environment Consistency**

31. Problem of “works on my machine”

32. Introduction to containers
33. Docker fundamentals
34. Difference between VM and Container
35. Docker architecture
36. Docker image vs container
37. Basic Dockerfile concept
38. Image build and run lifecycle

---

**Day 14 – SQL and Networks Assessment**

---

**Day 15 – All Modules Assessment**

---