NATIONAL SKILLS QUALIFICATION

LEVEL 4

TITLE:

Programming with Java

YEAR: 2024

NATIONAL SKILLS QUALIFICATION

NSQ LEVEL 4 – Programming with Java Intermediate

GENERAL INFORMATION

QUALIFICATION PURPOSE

The is designed to equip learner to develop intermediate knowledge and skills in Java programming with a focus on more complex OOP principles, data structures, and Java APIs.

QUALIFICATION OBJECTIVES

The learner should be able to: -

i. Work with Java collections (e.g., List, Set, Map) to store and manipulate data.
ii. Write code using generics and lambda expressions for cleaner and more efficient programs.
iii. Implement multithreading to handle concurrent tasks.
iv. Manage input/output operations using Java's I/O libraries.
v. Connect Java applications to databases using JDBC.

Mandatory Units

Unit No	Reference Number	NOS Title	Credit Value	Guided Learning Hours	Remark
1	ICT/JAVA/001/L4	Health & Safety	2	20	Level 4
2	ICT/JAVA/002/L4	Teamwork	2	20	Level 4
3	ICT/JAVA/003/L4	Communication in workplace	2	20	Level 4
4	ICT/JAVA/004/L4	Java Collections Framework: List, Set, Map, Queue	4	40	Level 4
5	ICT/JAVA/005/L4	Generics and Lambda Expressions	4	40	Level 4
6	ICT/JAVA/006/L4	Advanced OOP: Inner classes, abstract classes, and interfaces	4	40	Level 4
7	ICT/JAVA/007/L4	File I/O and serialization	4	40	Level 4
8	ICT/JAVA/008/L4	Multithreading and concurrency	4	40	Level 4
9	ICT/JAVA/009/L4	Java Memory Management (Garbage Collection)	4	40	Level 4
			30	300	

NATIONAL SKILLS QUALIFICATION

LEVEL 4: PROGRAMMING WITH JAVA INTERMEDIATE

Unit 1: Health and Safety in workplace

Unit Reference Number: ICT/JAVA/001/L3

NSQ Level: 4

Credit Value: 2

Guided Learning Hours: 20

Unit Purpose: This unit aims to equip trainees with the knowledge and skills to follow health and safety regulations while working in a Java programming environment.

Unit Objectives:

Unit assessment requirements/ evidence requirements:

Assessment must be carried out in real workplace environment in which learning and human development is carried out.

Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)
5. Recognition of Prior Learning (RPL)

UNIT 01: Health and Safety

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Identify and Ergonomic Risks in a Programming Environment	1.1	Set up an ergonomic workstation with proper seating, monitor height, and keyboard placement to reduce strain and fatigue.									
	1.2	Follow guidelines for regular breaks, eye strain reduction, and posture correction to avoid repetitive strain injuries (RSI).									
	1.3	Implement the use of ergonomic tools, such as wrist supports, adjustable chairs, and standing desks where applicable.									
	1.4	Apply secure coding standards to prevent vulnerabilities such as SQL injection, buffer overflows, and cross-site scripting (XSS).									
	1.5	Demonstrate the Use version control tools (e.g., Git) to manage code changes and ensure traceability while following security protocols.									
LO 2: Manage Occupational Stress and Mental Health in a Programming Role	2.1	Identify signs of burnout and high stress during demanding programming tasks, including prolonged debugging and tight deadlines.									
	2.2	Describe ways on time management techniques to balance work tasks, reducing the risk of stress and promoting a healthy work-life balance.									
	2.3	Explain/Develop mindfulness techniques, such as meditation or relaxation exercises, during breaks to maintain mental well-being.									
LO 3: Ensure Compliance with Health and Safety Regulations in a Remote or Office-Based Programming Environment	3.1	Follow workplace health and safety guidelines related to electrical equipment, proper ventilation, and fire safety while working in office environments.									
	3.2	Ensure remote working spaces are compliant with health and safety standards, including proper lighting, minimal distractions, and a safe electrical setup.									
	3.3	Regularly assess and adjust the workspace to maintain health, safety, and comfort for sustained programming work.									

Unit 2: Teamwork

Unit Reference Number: ICT/JAVA/002/L4

NSQ Level: 4

Credit Value: 4

Guided Learning Hours: 40

Unit Purpose: This unit aims to equip trainees to learn to function as integral team members, sharing responsibilities, solving coding challenges collectively, and maintaining a productive and collaborative environment.

Unit Objectives:

At the end of this unit, trainee should be able to:

1. Demonstrate the ability to work collaboratively with team members on Java programming tasks and projects.
2. Participate in code reviews, providing and receiving constructive feedback to improve the quality of the team's Java code.
3. Develop problem-solving strategies in collaboration with team members to address complex programming challenges.
4. Demonstrate flexibility by adapting to different roles within the team, including leadership, coding, and testing responsibilities.
5. Ensure that clear and concise communication is maintained throughout the project to avoid misunderstandings and to align on goals.

Unit assessment requirements/ evidence requirements:

Assessment must be carried out in real workplace environment in which learning and human development is carried out.

Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)
5. Recognition of Prior Learning (RPL)

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Understand Collaborating Effectively on a Java Development Team	1.1	Participate actively in team discussions, providing input on design, coding, and problem-solving decisions.									
	1.2	Share responsibilities equally, taking ownership of specific Java coding tasks and delivering them in a timely manner.									
	1.3	Use collaborative tools (e.g., Git, GitHub, or GitLab) to manage team contributions to the codebase, ensuring smooth integration and version control.									
LO 2: Contribute to Code Reviews and Provide Constructive Feedback	2.1	Review team members' code, identifying errors, suggesting improvements, and ensuring adherence to Java coding standards and best practices.									
	2.2	Provide feedback in a respectful and constructive manner, focusing on the code rather than the individual.									
	2.3	Demonstrate Accepting feedback on personal work with an open mind and a willingness to improve coding practices based on team input.									
LO 3: Solve Programming Challenges Through Team Collaboration	3.1	Work with team members to break down large programming problems into manageable tasks, assigning roles and responsibilities accordingly.									
	3.2	Contribute to brainstorming sessions where multiple solutions are evaluated and the most efficient one is selected for implementation.									
	3.3	Use pair programming or group coding sessions to troubleshoot difficult Java programming issues, leveraging collective knowledge to resolve bugs or inefficiencies.									
LO 4: Adapt to Different Team Roles and Responsibilities	4.1	Take on various team roles, such as project leader, developer, or tester, depending on the project's needs and team dynamics.									
	4.2	Provide leadership when necessary by coordinating tasks, setting milestones, and ensuring team members are on track with their coding objectives.									
	4.3	Assist teammates in their roles by offering help with coding tasks, troubleshooting bugs, or testing features they have developed.									
LO 5: Maintain Effective	5.1	Clearly communicate task progress, challenges, and solutions to team members through regular updates and meetings.									

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
Communication Within the Team	5.2	Use appropriate communication channels (e.g., Slack, emails, project management tools) to share important project information with the team.									
	5.3	Resolve any team conflicts or misunderstandings by discussing issues openly and working towards a collaborative solution.									

NATIONAL SKILLS QUALIFICATION

LEVEL 4: PROGRAMMING WITH JAVA INTERMEDIATE

Unit 3: Communication in Workplace

Unit Reference Number: ICT/JAVA/003/L4
NSQ Level: 4
Credit Value: 4
Guided Learning Hours: 40

Unit Purpose: This unit prepares trainees with advanced communication skills using modern technological tools in a professional Java programming environment.

Unit Objectives:

At the end of this unit, trainees should be able to:

1. Demonstrate the ability to use technological tools for effective communication and collaboration in a programming team.
2. Effectively use version control systems to track changes, communicate updates, and collaborate on Java codebases.
3. Participate in code review sessions using communication platforms and provide constructive feedback to team members.
4. Communicate effectively while working with geographically dispersed teams using remote work tools and techniques.
5. Use modern communication platforms to deliver professional project updates to stakeholders and team members.
6. Use collaborative tools to document and share technical solutions with the team and other departments.

Unit assessment requirements/ evidence requirements:

Assessment must be carried out in real workplace environment in which learning and human development is carried out.

Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)
5. Recognition of Prior Learning (RPL)

UNIT 03: Communication in Workplace

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Utilize Communication Tools to Collaborate	1.1	Use collaboration tools such as Slack, Microsoft Teams, or Discord to facilitate real-time communication and project discussions.									
	1.2	Demonstrate Sharing and managing project updates, documents, and tasks using project management platforms like Trello, Jira, or Asana.									
	1.3	Carry out video conferencing tools (e.g., Zoom, Google Meet) for remote meetings, ensuring professional presentation and engagement during virtual discussions.									
LO 2: Leverage on Version Control Systems to Enhance Team Communication	2.1	Demonstrate the Use of Git or GitHub to share code, track changes, and manage contributions from multiple developers within the team.									
	2.2	Construct clear and concise commit messages to communicate the purpose of code changes to the team.									
	2.3	Review and merge pull requests, ensuring effective communication during the process to resolve conflicts and maintain code quality.									
LO 3: Use Communication Tools to Facilitate Code Reviews and Feedback	3.1	Participate in real-time code reviews using GitHub or GitLab.									
	3.2	Develop a Schedule and conduct virtual code review sessions using screen-sharing tools, ensuring clear communication and collaboration during reviews.									
	3.3	Describe how to Document feedback and agreed-upon changes using shared communication tools to ensure all team members are aware of revisions.									
LO 4: Adapt Communication for Remote Java Programming Teams	4.1	Demonstrate the Use of cloud-based tools like Google Drive, Dropbox, or OneDrive to share documents.									
	4.2	Generate a Schedule and coordinate meetings across different time zones using scheduling platforms like Google Calendar or Microsoft Outlook.									
	4.3	Describe how to Maintain clear communication channels through asynchronous tools (e.g., emails, message boards) to ensure alignment on project objectives, despite time zone differences.									

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 5: Understand Java Project Updates Using Professional Communication Tools	5.1	Prepare and present Java project updates using presentation tools such as Microsoft PowerPoint, Google Slides, or Prezi during virtual meetings.									
	5.2	Explain how to Use reporting tools like Confluence or Google Docs to create and share detailed project reports with stakeholders, ensuring clarity and completeness.									
	5.3	Leverage real-time dashboards or reporting software (e.g., Jira dashboards, Trello boards) to provide stakeholders with live updates on project progress.									

NATIONAL SKILLS QUALIFICATION

LEVEL 4: Programming with Java Intermediate

Unit 4: JAVA COLLECTIONS FRAMEWORK: LIST, SET, MAP, QUEUE

Unit Reference Number: ICT/JAVA/004/L4
NSQ Level: 4
Credit Value: 4
Guided Learning Hours: 40

Unit Purpose: This unit aims to equip trainees with an in-depth understanding of the Java Collections Framework, focusing on Lists, Sets, Maps, and Queues.

Unit Objectives:
At the end of this unit, trainees should be able to:

1. Understand Collections Framework, including key interfaces such as List, Set, Map, and Queue.
2. Demonstrate practical skills in implementing and manipulating collections such as Lists, Sets, Maps, and Queues.
3. Understand how to optimize the use of Lists, Sets, Maps, and Queues to improve the performance of Java applications.
4. Demonstrate the ability to traverse and process collections using iterators and Java Streams.
5. learn how to handle concurrent modifications and ensure thread safety when working with collections in a multi-threaded environment.
6. Apply their knowledge of the Java Collections Framework to solve real-world programming problems that require data structure management.

Unit assessment requirements/ evidence requirements:
Assessment must be carried out in real workplace environment in which learning and human development is carried out.
Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)

UNIT 04: JAVA COLLECTIONS FRAMEWORK: LIST, SET, MAP, QUEUE

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Understand the Core Concepts of the Java Collections Framework	1.1	Explain the purpose and structure of the Java Collections Framework and its use in Java programming.									
	1.2	Differentiate between various collection types (List, Set, Map, Queue) and identify appropriate use cases for each.									
	1.3	Describe the hierarchical relationship between collection interfaces and classes in Java.									
LO 2: Implement and Manipulate Lists, Sets, Maps, and Queues in Java	2.1	Demonstrate Write Java programs that use ArrayList, LinkedList, and Vector to store and manage ordered data.									
	2.2	Implement a HashSet, TreeSet, or LinkedHashSet for managing unique elements and practice common operations such as adding, removing, and searching for elements.									
	2.3	Use HashMap, TreeMap, or LinkedHashMap to store key-value pairs, demonstrating the ability to retrieve, update, and remove entries efficiently.									
	2.4	Implement a PriorityQueue or LinkedList as a queue and perform operations like enqueue, dequeue, and peeking at the head of the queue.									
	2.5	Write Java programs that use ArrayList, LinkedList, and Vector to store and manage ordered data.									
LO 3: Optimize Collection-Based Solutions for Performance	3.1	Analyze the time complexity of various operations (e.g., insertion, deletion, searching) across different collection types.									
	3.2	Select the most appropriate collection type for a given problem based on performance considerations (e.g., ArrayList vs. LinkedList, HashMap vs. TreeMap).									
	3.3	Refactor existing code to optimize the use of collections, reducing memory overhead or improving runtime performance.									
LO 4: Use Iterators and Streams to Process Collections	4.1	Implement Iterator and ListIterator to traverse through Lists, Sets, and Maps, performing operations like filtering or modifying elements during iteration.									
	4.2	Use the enhanced for loop and foreach method to iterate over collections and perform common tasks such as data transformation or aggregation.									

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
	4.3	Apply Java Streams to perform declarative operations on collections (e.g., map, filter, collect) and optimize data processing pipelines.									
LO 5: Handle Concurrent Modifications and Synchronization of Collections	5.1	Identify common pitfalls related to concurrent modifications in collections and implement solutions to avoid exceptions such as ConcurrentModificationException.									
	5.2	Use Collections.synchronizedList, ConcurrentHashMap, and other thread-safe collection types to manage shared data in a concurrent environment.									
	5.3	Implement proper synchronization techniques, such as using locks or concurrent classes, to ensure that collections are modified safely when accessed by multiple threads.									
LO 6: Apply Java Collections in Real-World Programming Scenarios	6.1	implement a Java application that effectively uses a combination of Lists, Sets, Maps, and Queues to manage complex data.									
	6.2	Integrate collections into larger programming solutions.									
	6.3	Test and debug Java applications to ensure that collections are correctly implemented and that data integrity is maintained.									

NATIONAL SKILLS QUALIFICATION

LEVEL 4: PROGRAMMING WITH JAVA INTERMEDIATE

Unit 5: Generics and Lambda Expressions in Java

Unit Reference Number: ICT/JAVA/005/L4
NSQ Level: 4
Credit Value: 4
Guided Learning Hours: 40

Unit Purpose: This unit aims to equip trainees with comprehensive knowledge and practical skills in using Generics and Lambda Expressions in Java.

Unit Objectives:
At the end of this unit, the trainees will be able to:

1. Demonstrate a foundational understanding of Generics in Java, including their purpose and how to implement them in code.
2. Develop and use Generics to write code that can handle multiple types while avoiding redundancy and ensuring flexibility.
3. Use Lambda Expressions to simplify code logic, particularly in functional programming scenarios like filtering, mapping, and reducing data.
4. Combine Generics and Lambda Expressions to solve real-world programming challenges efficiently.

Unit assessment requirements/ evidence requirements:
Assessment must be carried out in real workplace environment in which learning and human development is carried out.
Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)
5. Recognition of Prior Learning (RPL)

UNIT 05: GENERICS AND LAMBDA EXPRESSIONS IN JAVA

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Understand the Purpose and Structure of Generics in Java	1.1	Describe Generics in Java									
	1.2	Explain the purpose of Generics in Java and how they enhance type safety and reusability in code.									
	1.3	Differentiate between bounded and unbounded Generics, as well as wildcards (? extends, ? super), and understand their use cases.									
	1.4	Implement simple generic classes and methods to handle various data types without compromising type safety.									
LO 2: Use Generics to Develop Flexible and Reusable Code	2.1	Create generic methods and classes to perform operations on different data types, such as sorting or filtering data.									
	2.2	Apply bounded types to restrict the permissible types for Generics, ensuring only valid objects are passed into generic classes or methods.									
	2.3	Refactor existing Java code to incorporate Generics, reducing redundancy and improving maintainability.									
LO 3: Implement Lambda Expressions for Functional Programming in Java	3.1	Explain the syntax and structure of Lambda Expressions, including their use in replacing anonymous inner classes.									
	3.2	Implement Lambda Expressions to streamline common operations such as filtering, sorting, and transforming collections.									
	3.3	Use Lambda Expressions in conjunction with functional interfaces such as Predicate, Function, Supplier, and Consumer to implement concise, functional code.									
LO 4: Apply Generics and Lambda Expressions in Real-World Java Applications	4.1	Implement a Java application that uses Generics to manage multiple data types, while Lambda Expressions are used for operations like filtering and mapping.									
	4.2	Construct reusable, type-safe methods using Generics, while integrating Lambda Expressions to simplify the flow of data processing in the application.									
	4.3	Carry out Test and debug Java applications, ensuring that the use of Generics and Lambda Expressions leads to more concise, efficient, and maintainable code.									

NATIONAL SKILLS QUALIFICATION

LEVEL 6: PROGRAMMING WITH JAVA INTERMEDIATE

Unit 6: ADVANCED OOP: INNER CLASSES, ABSTRACT CLASSES, AND INTERFACES

Unit Reference Number:
NSQ Level: 4
Credit Value: 4
Guided Learning Hours: 40

Unit Purpose: This unit aims to equip trainees with the knowledge and skills required to work effectively with Inner Classes, Abstract Classes, and Interfaces in Java.

Unit Objectives:

Trainees should be able to:

1. To develop modular, reusable, and maintainable code structures that align with industry standards in Java programming.
2. Know how to manipulate Inner Classes, Abstract Classes, and Interfaces in Java.
3. To work effectively with programming constructs, and will be able to build flexible, modular, and reusable code that meets industry standards.

Unit assessment requirements/ evidence requirements:

Assessment must be carried out in real workplace environment in which learning and human development is carried out.

Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)
5. Recognition of Prior Learning (RPL)

UNIT 6: ADVANCED OOP: INNER CLASSES, ABSTRACT CLASSES, AND INTERFACES

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Understand and Implement Inner Classes for Better Code Organization	1.1	Differentiate between the four types of inner classes: static nested classes, non-static inner classes, local classes, and anonymous classes.									
	1.2	Implement static nested classes and non-static inner classes to encapsulate logically related code inside outer classes.									
	1.3	Write Java programs using anonymous classes to handle concise event-driven programming or single-method implementations, such as in GUI applications.									
	1.4	Refactor existing code to encapsulate functionality more effectively using inner classes.									
LO 2: Work with Abstract Classes to Achieve Code Generalization and Reusability	2.1	Explain the purpose of abstract classes in enforcing a base structure for subclasses, distinguishing between abstract and concrete methods.									
	2.2	Create abstract classes and implement them in Java programs to define a common template for related classes while leaving specific behavior to subclasses.									
	2.3	Use abstract methods in abstract classes to enforce implementation in subclasses, ensuring consistency in behavior across class hierarchies.									
	2.4	Refactor existing Java code to introduce abstract classes, simplifying maintenance by reducing redundancy and code duplication.									
LO 3: Utilize Interfaces to Define Contracts and Implement Multiple Inheritance	3.1	Define and implement interfaces that specify methods without bodies, ensuring that implementing classes follow a common contract.									
	3.2	Implement multiple interfaces in a single class to simulate multiple inheritance, achieving more flexible and modular designs.									
	3.3	Utilize default and static methods in interfaces where necessary to provide additional functionality without breaking the contract of the interface.									
	3.4	Write and refactor Java programs to incorporate interfaces, ensuring that classes adhere to clearly defined contracts while remaining adaptable to change.									

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 4: Integrate Inner Classes, Abstract Classes, and Interfaces in Real-World Java Applications	4.1	Develop Java applications that use a combination of inner classes to modularize tightly related functions, such as event handling or utility logic.									
	4.2	Implement abstract classes to serve as a base for similar classes, ensuring that specific behavior is delegated to concrete subclasses.									
	4.3	Design Java applications using interfaces to enable flexible component interaction and ensure that classes adhere to common behaviors or contracts.									
	4.4	Test and debug Java programs to verify correct implementation of inner classes, abstract classes, and interfaces, ensuring modularity, reusability, and maintainability.									

NATIONAL SKILLS QUALIFICATION

LEVEL 4: PROGRAMMING WITH JAVA INTERMEDIATE

Unit 7: File Input/Output Serialization

Unit Reference Number: ICT/JAVA/007/L4
NSQ Level: 4
Credit Value: 4
Guided Learning Hours: 40

Unit Purpose: This unit is aimed at to providing trainees with skills and hands-on experience in File Input/Output (I/O) operations and Serialization.

Unit Objectives:
Trainees will be proficient in managing file-based data and preserving object states for future use.

Unit assessment requirements/ evidence requirements:
Assessment must be carried out in real workplace environment in which learning and human development is carried out.
Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)

UNIT 7: File Input/Output Serialization

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Understand the Basics of File I/O Operations in Java	1.1	Explain the purpose and structure of Java's java.io package, focusing on I/O streams such as FileInputStream, FileOutputStream, BufferedReader, and BufferedWriter.									
	1.2	Implement basic file reading and writing operations using file streams to manage text files.									
	1.3	Use file handling classes like File to manipulate file and directory paths.									
	1.4	Handle file exceptions using try-with-resources to ensure proper resource management and avoid memory leaks.									
LO 2: Implement Efficient File I/O Using Buffered Streams	2.1	Use BufferedReader and BufferedWriter to enhance the efficiency of file reading and writing operations by reducing the number of I/O interactions.									
	2.2	Compare the performance differences between unbuffered and buffered I/O operations and understand when to use each.									
	2.3	Implement file copying programs using buffered streams to handle large file transfers efficiently.									
	2.4	Manage exceptions effectively during file I/O operations, ensuring that file handles and streams are closed properly.									
LO 3: Work with Serialization to Save and Restore Object States	3.1	Define the concept of serialization in Java.									
	3.2	Explain the role of the Serializable interface in object serialization.									
	3.3	Implement object serialization using ObjectOutputStream and ObjectInputStream to write objects to and read them from files.									
	3.4	Demonstrate how to serialize complex objects, including those containing references to other objects (object graphs).									
LO 4: Apply File I/O and Serialization in Real-World Java Applications	4.1	Develop a Java application that reads and writes structured data to files using efficient file I/O operations, ensuring the correct format and structure of the data.									
	4.2	Implement a program that serializes and deserializes objects to save application state between runs or transfer data between systems.									
	4.3	Test and debug Java applications to ensure that file handling, serialization, and deserialization are performed correctly and data integrity is maintained.									

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
	4.4	Demonstrate the ability to work with different file formats (e.g., text files, binary files) and manage file permissions and exceptions during file operations.									

NATIONAL SKILLS QUALIFICATION

LEVEL 8: PROGRAMMING WITH JAVA INTERMEDIATE

UNIT 8: MULTITHREADING AND CONCURRENCY IN JAVA

Unit Reference Number: ICT/JAVA/008/L4
NSQ Level: 4
Credit Value: 4
Guided Learning Hours: 40

Unit Purpose: This unit will equip trainees with the foundational knowledge and practical skills necessary to understand and implement multithreading and concurrency in Java.

Unit Objectives:
Trainees will be able to use these concepts essentially for creating applications that can perform multiple tasks simultaneously, improving performance and responsiveness.

Unit assessment requirements/ evidence requirements:
Assessment must be carried out in real workplace environment in which learning and human development is carried out.
Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)

UNIT 8: MULTITHREADING AND CONCURRENCY IN JAVA

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
LO 1: Understand the Basics of Multithreading in Java	1.1	Explain the concept of multithreading and how it improves the performance of applications by executing multiple tasks concurrently.									
	1.2	Create and manage threads using both the Thread class and the Runnable interface.									
	1.3	Write Java programs that initiate, start, and control the lifecycle of threads, including starting, pausing, and stopping threads.									
	1.4	Understand thread priorities and demonstrate how to adjust the execution order of threads using thread priorities.									
LO 2: Implement Thread Synchronization to Avoid Data Inconsistency	2.1	Explain the concept of thread synchronization and the issues that arise from unsynchronized access to shared resources (e.g., race conditions).									
	2.2	Use synchronized methods and blocks to control access to shared resources, ensuring data consistency in multithreaded environments.									
	2.3	Demonstrate how to avoid deadlock situations by carefully controlling thread access to multiple resources.									
	2.4	Implement inter-thread communication using wait(), notify(), and notifyAll() methods to coordinate the execution of threads.									
LO 3: Work with Java Concurrency Utilities for Advanced Thread Management	3.1	Define Executor Framework in Java									
	3.2	Explain the role of the Executor framework and its advantages over manually managing threads.									
	3.3	Implement a thread pool using the ExecutorService to manage a group of threads efficiently, improving application performance.									
	3.4	Use Callable and Future to run tasks concurrently and retrieve results after completion.									
	3.5	Demonstrate how to properly shut down the ExecutorService to ensure all threads terminate gracefully and resources are released.									
	3.6	Use the java.util.concurrent package to manage thread execution, such as using ExecutorService to create thread pools and manage tasks.									

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
	3.7	Implement thread-safe collections (e.g., ConcurrentHashMap) to avoid concurrency issues while accessing shared data structures.									
	3.8	Use synchronization utilities like CountDownLatch, CyclicBarrier, and Semaphore to coordinate complex multithreaded operations.									
	3.9	Demonstrate how to use Future and Callable interfaces to handle results from asynchronous computations in multithreaded applications.									
LO 4: Apply Multithreading and Concurrency in Real-World Java Applications	4.1	Use concurrent data structures from the java.util.concurrent package, such as ConcurrentHashMap, to avoid data inconsistencies in multi-threaded environments.									
	4.2	Detect and avoid deadlock situations by properly managing locks and synchronizing threads in a structured way.									
	4.3	Implement atomic operations using classes like AtomicInteger to perform thread-safe, lock-free operations.									
	4.5	Develop a Java application that spawns multiple threads to perform parallel tasks, such as file processing or network requests, ensuring resource efficiency.									
	4.6	Implement a multithreaded Java program that synchronizes access to shared resources, ensuring data integrity across all threads.									
	4.7	Use the ExecutorService to manage a pool of threads that handle large-scale data processing tasks, optimizing resource usage and application throughput.									
	4.8	Test and debug multithreaded Java applications to ensure thread safety, correct synchronization, and the elimination of race conditions and deadlocks.									

NATIONAL SKILLS QUALIFICATION

LEVEL 4: PROGRAMMING WITH JAVA INTERMEDIATE

UNIT 9: JAVA MEMORY MANAGEMENT (GARBAGE COLLECTION)

Unit Reference Number: ICT/JAVA/009/L4
NSQ Level: 4
Credit Value: 4
Guided Learning Hours: 40

Unit Purpose: This Unit aims to equip Trainees with knowledge and skills to explore practical techniques for optimizing garbage collection and tuning memory settings, making them proficient in ensuring efficient memory usage in real-world Java applications.

Unit Objectives:
At the end of this unit, trainees will be able to:

1. Understand how the JVM allocates and reclaims memory.
2. Learn how garbage collection works.
3. Learn how to prevent memory-related issues such as leaks.

Unit assessment requirements/ evidence requirements:
Assessment must be carried out in real workplace environment in which learning and human development is carried out.
Assessment methods to be used include:

1. Direct Observation/oral questions (DO)
2. Question and Answer (QA)
3. Witness Testimony (WT)
4. Assignment (ASS)
5. Recognition of Prior Learning (RPL)

UNIT 9: JAVA MEMORY MANAGEMENT (GARBAGE COLLECTION)

LEARNING OBJECTIVE (LO) **The learner will:**		**PERFORMANCE CRITERIA** **The learner can:**	**Evidence Type**					**Evidence Ref. Page No.**			
LO 1: Understand Java's Memory Management Model	1.1	Explain the structure of Java memory areas, including the heap, stack, and method area, and how they are used to manage application data.									
	1.2	Identify and differentiate between memory allocated for primitive data types and objects in Java.									
	1.3	Describe the role of the Java Virtual Machine (JVM) in memory management and its interaction with different memory regions.									
	1.4	Explain the process of object creation and lifecycle in the heap, including how objects are referenced and dereferenced.									
LO 2: Work with the JVM Garbage Collection Process	2.1	Explain the concept of garbage collection and why it is necessary to manage memory in Java.									
	2.2	Identify the different types of garbage collectors (e.g., Serial, Parallel, CMS, G1) and explain their benefits and use cases.									
	2.3	Understand how the JVM tracks object references and triggers garbage collection based on reference counting or reachability analysis.									
	2.4	Implement a program to demonstrate how unreachable objects are automatically collected by the JVM to free memory.									
LO 3: Tune and Optimize Garbage Collection for Performance	3.1	Configure the JVM's garbage collection behavior using command-line options (e.g., -XX:+UseG1GC, -Xms, -Xmx).									
	3.2	Monitor and analyze garbage collection logs to identify performance bottlenecks or memory inefficiencies.									
	3.3	Use profiling tools like VisualVM or JConsole to observe heap usage and garbage collection cycles in a running Java application.									
	3.4	Optimize garbage collection behavior for specific applications by adjusting heap sizes, GC algorithms, and other tuning parameters.									
LO 4: Prevent Memory Leaks and Manage Object References	4.1	Identify common causes of memory leaks in Java applications, such as improper use of static references or failing to remove objects from collections.									
	4.2	Implement best practices for managing object references, including the use of									

LEARNING OBJECTIVE (LO) The learner will:		PERFORMANCE CRITERIA The learner can:	Evidence Type					Evidence Ref. Page No.			
		WeakReference, SoftReference, and PhantomReference.									
	4.3	Ensure objects are eligible for garbage collection by managing object lifecycles properly, avoiding unnecessary retention of references.									
	4.4	Debug and fix memory leaks using memory profiling tools like jmap, jhat, or Eclipse MAT.									
LO 5: Understand the Impact of Object Finalization and Cleanup	5.1	Explain the role of the finalize() method in object cleanup and why it is discouraged in modern Java programming.									
	5.2	Implement a simple finalize() method to observe its behavior and understand how it fits into the garbage collection cycle.									
	5.3	Explain the limitations of finalization and the importance of using proper resource management techniques such as try-with-resources.									
	5.4	Explain the rationale on best practices for handling resource cleanup, avoiding reliance on the garbage collector for releasing critical resources (e.g., file handles, network connections).									

PARTICIPANT FOR CRITIQUE WORKSHOP

S/N	Full Name	Organization	Address	Email	Telephone
1	OBIAHU, Okechukwu Othniel	Oando Energy Resources Nigeria Ltd.	No 43 NDDC Road 11, Rumukwurusi Pipeline, Rivers State	othnielobiahu@yahoo.com	08038869114
2	FASINA, Felicia ltse	NBTE	Plot B Bida Road, NBTE, Kaduna	feliciasina@gmail.com	08036570850
3	ABDULLAHI, Lawal	KAD ICT HUB	No 47 Kanta Road Off Independence Way, Kaduna State	ocplawal@gmail.com	08035169089
4	YOUNG-HARRY, Constance Soye	Ministry of Education Rivers State	Road 12, House 14 Trans Amadi Gardens Port Harcourt, Rivers State	constanceyoungharry@gmail.com	08032684914
5	MUHAMMAD, BILYAMINU MUSA	NBTE	PLOT B, Bida Road, Kaduna	mahogany@gmail.com	09036071291
6	Muhammad Bello Aliyu	CPN	1321 Adesoji Aderemi Street, Gudu District, Apo Abuja FCT	mbacaspet@gmail.com	08039176984
7	BENJAMIN, Prince Chukwudindu	CPN	1321 Adesoji Aderemi Street, Gudu District, Apo Abuja FCT	Pco.benjamin@gmail.com	08132850544
8	Amoo, Taofeek	CPN	1321 Adesoji Aderemi Street, Gudu District, Apo Abuja FCT	taofeekamoo@gmail.com	08053370334
9	Olatunji Abibat	CPN	1321 Adesoji Aderemi Street, Gudu District, Apo Abuja FCT	adehabb@gmail.com	08054263602

10	Linda Ngbeken	CPN	1321 Adesoji Aderemi Street, Gudu District, Apo Abuja FCT	excel4all2000@yahoo.com	08128219274

PARTICIPANT FOR VALIDATION WORKSHOP

S/N	Full Name	Organization	Address	Email	Telephone
1	OBIAHU, Okechukwu Othniel	Oando Energy Resources Nigeria Ltd.	No 43 NDDC Road 11, Rumukwurusi Pipeline, Rivers State	othnielobiahu@yahoo.com	08038869114
3	ABDULLAHI, Lawal	KAD ICT HUB	No 47 Kanta Road Off Independence Way, Kaduna State	ocplawal@gmail.com	08035169089
4	YOUNG-HARRY, Constance Soye	Ministry of Education Rivers State	Road 12, House 14 Trans Amadi Gardens Port Harcourt, Rivers State	constanceyoungharry@gmail.com	08032684914
5	Dr. Musa Hatim Koko		PLOT B, Bida Road, Kaduna	Hatimlion@gmail.com	08039606948
6	MUHAMMAD, BILYAMINU MUSA	NBTE	PLOT B, Bida Road, Kaduna	mahogany@gmail.com	09036071291
7	Muhammad Bello Aliyu	CPN	1321 Adesoji Aderemi Street, Gudu District, Apo Abuja FCT	mbacaspet@gmail.com	08039176984
8	BENJAMIN, Prince Chukwudindu	CPN	1321 Adesoji Aderemi Street, Gudu District, Apo Abuja FCT	Pco.benjamin@gmail.com	08132850544