Model Curriculum

Coding

SECTOR	**:**	IT
SUB-SECTOR	**:**	PROGRAMMING
OCCUPATION	**:**	TRAINEE ASSOCIATE
REF ID	**:**	MES/NSU/Q 0901
NSQF LEVEL	**:**	4
VERSION	**:**	1.0

Certificate

Coding

TABLEOF CONTENTS

Coding

CODING

CURRICULUM/SYLLABUS

This program is aimed at training candidates for the job of a "Trainee Associate", in the "IT"Sector/Industryandaimsatbuildingthefollowingkeycompetenciesamongstthe learner.

TrainingDeliveryPlan			
ProgramName:	Coding Trainee Associate		
Qualification Pack Name&Ref.ID	MES/NSU/Q 0901		
VersionNo.	1.0	**Version Update Date**	*To be obtained from NSQF after submission of the course.*
Pre-requisites To Training	Class XII with one year of Relevant experience **OR** ITI(2yearsafter10th)withoneyearofrelevantexperience MinimumAge:18Years		
TrainingOutcome	**By the end of this program, the participants would haveachieved the following competencies:** • Understand Basic Concepts of Object Oriented Paradigms • Object Oriented Programming using C++ • Object Oriented Programming using Java • Case study implementation based on C++/Java Coding • Maintain a healthy, safe and secure working environment		

NOS and Module Details

S.N.	NOS and Module Details	Theory (Hours)	Practical (Hours)	Total Duration (Hours)
1	Basic Concepts of Object-Oriented Paradigms	36	54	90
2	Turbo C++ Software	36	54	90
3	Java Software	36	54	90
4	Case study implementation based on C++/Java Coding	36	54	90
5	Maintain Workplace Health and Safety	12	18	30
6	Employability Skills	24	36	60
	Total	**180**	**270**	**450**

Credits	Total Notional Hours	Hours per Credit
15	**450**	**1 Credit = 30 hrs(NCrF)** **ABC Complaint**

Coding

This course encompasses all National Occupational Standards (NOS) of "Coding" Qualification pack issued by "IT - SECTOR"

Sr. No.	Module	KeyLearningOutcomes	Equipment Required
1	MES/NSU/N0902: Basic Concepts of Object-Oriented Paradigms Theory Duration (hh:mm) 25:00 Practical Duration (hh:mm) 25:00	- Understand basic concepts of Object-Oriented Programming (OOP). - Identify problems in Procedural Oriented Programming. - Learn core OOP features (abstraction, encapsulation, inheritance, polymorphism). Understand OOP implementation across modern languages: C++, Java, Python (intro), and C# (intro). - Differentiate between OOP and procedural paradigms using multilingual code snippets.	• Laptop • - Whiteboard & Marker • - Projector • - UPS • Installations for Python (3.x) and .NET SDK (for C#) • - VS Code or JetBrains IDEs • - Sample source files in Python/C++/Java/C#
2	MES/NSU/N0903:Turbo C++ Software TheoryDuration (hh:mm) 35:00 PracticalDuration (hh:mm) 65:00	Explain core object-oriented programming (OOP) concepts such as encapsulation, inheritance, polymorphism, and abstraction. Write and debug C++ programs using modern compilers (GCC/Clang) and IDEs (Code::Blocks, VS Code, Eclipse CDT). Create and use classes, objects, constructors, destructors, and operator/function overloading. Apply exception handling techniques to develop error-resilient applications. Use templates to implement generic programming concepts. Compile and manage C++ projects using terminal commands and IDE features. (Optional) Utilize basic build	• Laptop • White board • Marker • Projector • GCC/Clang C++ compiler (latest stable version) • Code::Blocks / Visual Studio Code / Eclipse CDT (for C++) • UPS

		automation tools like Makefiles or CMake. Practice structured programming, coding standards, and effective debugging techniques in real-world scenarios. Understand risks of unsafe operations like buffer overflows in C++ and apply input validation and safe memory handling practices.	
3	MES/NSU/N0904: Java Software TheoryDuration (hh:mm) 35:00 PracticalDuration (hh:mm) 65:00	Understand core Java programming concepts including syntax, data types, control structures and object-oriented principles. Develop, compile, and debug Java applications using modern tools like OpenJDK and IDEs (IntelliJ IDEA, Eclipse, VS Code). Create and work with classes, objects, inheritance, polymorphism, and interfaces in Java. Implement exception handling for robust application development. Use Java collections and generics for efficient data management. Build simple GUI applications (optional extension) using Java frameworks. Apply best coding practices and debugging techniques in Java projects. Understand and use build tools and project management features available in IDEs. Understand secure coding practices such as input validation, exception handling, and avoiding common vulnerabilities (e.g., injection, buffer overflows).	• Laptop • White board • Marker • Projector • OpenJDK 17+ • IntelliJ IDEA / Eclipse / VS Code (with Java plugins) • UPS
4	MES/NSU/N0905:Case study implementation based on C++/Java Coding	Understanding of the C++ language syntax and semantics. Earn how to write well-structured	• Laptop • Whiteboard • Marker

Coding

	TheoryDuration (hh:mm) 40:00 PracticalDuration (hh:mm) 70:00	and maintainable C++ code. Develop skills to identify and fix errors in their C++ programs. Use an Integrated Development Environment (IDE) to write, compile, debug, and run Java programs. Use Javadoc to document Java code. Identify, fix, and test Java programs for correctness and efficiency. Apply Java programming skills to solve real-world problems. Write clean, well-structured, and maintainable code.	• Projector • UPS
5	MES/NSU/N0906:Maintain Workplace Health and Safety TheoryDuration (hh:mm) 20:00 PracticalDuration (hh:mm) 25:00	Understand and comply with the organization's current health, safety and security policies and procedures. Understand the safe working practices pertaining to own occupation. Understand the government norms and policies relating to health and safety including emergency procedures for illness, accidents, fires or others which may involve evacuation of the premises. Identify security signals e.g. fire alarms and places such as staircases, fire warden stations, first aid and medical rooms. Identify aspects of your workplace that could cause potential risk to own and others health and safety. Identify and recommend opportunities for improving health, safety, and security to the designated person. Identify and correct risks like illness, accidents, fires or any	• Laptop • White board • Marker • Projector • UPS • Health and Safety Signs and policy

Coding

		other natural calamity safely and within the limits of individual's authority.	
6	DGT/VSQ/N0102: Employability Skills Theory Duration (hh:mm) 20:00 Practical Duration (hh:mm) 25:00	Communicate effectively using verbal, non-verbal, and digital communication in a professional IT environment. Apply self-management, time management, and interpersonal skills in software development teams. Use digital productivity tools (Docs, Sheets, Slides, Email, GitHub, LinkedIn) for collaborative project work. Understand the freelancing model, entrepreneurship basics, and start-up culture in the IT domain. Demonstrate digital financial literacy (UPI, online banking, taxation for freelancers). Create resumes, cover letters, and professional portfolios for coding and software roles. Practice ethical behavior, green computing, and cyber hygiene in coding environments. Prepare for interviews, online assessments, and client communications. Apply digital literacy skills in a software development context, including secure coding and cyber hygiene practices.	• Laptop with internet access • Projector & whiteboard • Google Workspace/MS Office • Resume builder tools (Canva, Zety, etc.) • Access to LinkedIn, GitHub • Safety charts and digital policy guidelines

Coding

	Total Duration (hh:mm) **450:00** **Theory Duration (hh:mm)** **175:00** **Practical Duration (hh:mm)** **275:00**	**Unique Equipment Required:** White board, Marker, Projector, Laptop, UPS, Do's and Dont's instruction Charts. Turbo C++ and JAVA Software Electricity Tester, Safety and Ergonomics Charts, Fire Extinguisher, and First-Aid Kit.	

Grand Total Course Duration: 450 Hours, 0 Minutes

Coding

Trainer Prerequisites for Job role: "Trainee Associate" mapped to Qualification Pack: "MES/NSU/Q0901, v1.0"

S. No	Area	Details
1	Description	• Individuals at this job are responsible for Trainee Associate of software applications and interfaces as well as enhancements to existing packaged applications or pre-engineered templates. The job also involves providing support to custom applications, debugging, maintenance and documentation.
2	Personal Attributes	• This job requires the individual to work dependently and be comfortable in making decisions pertaining to his/her area of work. The individual should not come under result oriented. The individual should also be able to demonstrate skills for communication and logical thinking related to applications.
3	Minimum Educational Qualifications	• Graduation in Computer Science / Information Technology / Computer Engineering or equivalent. • Higher qualifications (Postgraduate in CS/IT, MCA, M.Tech) preferred.
4	Professional Experience:	• Minimum 3 years of industry experience in software development, programming, or IT services. • Demonstrated expertise in C++, Java, and OOP concepts. • Experience with modern tools such as IDEs (VS Code, IntelliJ, Eclipse), Git/GitHub, and project workflows.
5	Training Credentials:	• Mandatory completion of SSC/NCVET-recognized Trainer Certification / ToT (Training of Trainers) program. • Ability to deliver both theory and practical sessions, including guiding learners in projects and case studies.
6	Continuous Professional Development (CPD):	• Minimum 16 hours of annual upskilling/CPD, covering:

Coding

		• Modern Java features (streams, lambdas, modularity). • Python/C# OOP introductions. • Secure coding practices. • Cloud IDEs and collaborative platforms. • Agile tools and workflows (e.g., JIRA, Trello).
7	Soft Skills Competence:	• Effective communication and presentation skills. • Ability to mentor students on employability skills, workplace professionalism, and teamwork.
8	Domain Certification	• Certified for Job Role: "Coding" mapped to QP: "MES/NSU/Q0901", version 1.0. Minimum accepted score as per SSC guidelines is 80%.
9	Platform Certification	• Recommended that the Trainer is certified for the Job Role: "Trainee Associate", mapped to the Qualification Pack: "SSC/Q1402" with scoring of minimum 80%.

Assessor Prerequisites (for completeness)

- Must be **certified by SSC/NCVET** as an assessor.
- At least **3 years of industry/teaching experience** in programming.
- Independent from the Training Provider delivering the program.

Trainer Continuous Professional Development (CPD) Requirement

To ensure trainers remain current with evolving technologies and teaching methodologies, it is mandatory for trainers delivering the **Coding Trainee Associate** program to complete **at least 16 hours of upskilling annually**.

Suggested CPD Topics Include:

- Modern Java features (e.g., Streams API, Lambdas, Modules)
- Latest C++ standards and best practices (C++17, C++20 features)
- Integrated Development Environment (IDE) tools and productivity enhancements (e.g., IntelliJ IDEA, Visual Studio Code)
- Cloud-based IDEs and collaborative coding platforms (e.g., GitHub Codespaces, AWS Cloud9)
- Secure coding practices and cybersecurity awareness
- Emerging programming languages and frameworks relevant to OOP and coding fundamentals

Documentation: Trainers must maintain records of completed CPD hours and submit a brief report during annual trainer audits.

Assessment Criteria

Annexure: Assessment Criteria

Job Role: Coding
Qualification Pack: MES/NSU/Q0901
Sector Skill Council: IT Sector

S.No	NOS Name	Weightage	
1	MES/NSU/N0902	Basic Concepts of Object-Oriented Paradigms	10%
2	MES/NSU/N0903	Turbo C++ Software	20%
3	MES/NSU/N0904	Java Software	20%
4	MES/NSU/N0905	Case study implementation based on C++/Java Coding	35%
5	MES/NSU/N0906	Maintain Workplace Health and Safety	5%
6	DGT/VSQ/N0102	Employability Skills	10%
TOTAL			100%

Guidelines for Assessment:
1. Criteria for assessment for each Qualification Pack will be created by the Sector Skill Council. Each Performance Criteria (PC) will be assigned marks proportional to its importance in NOS. SSC will also lay down proportion of marks for Theory and Skills Practical for each PC.
2. The assessment for the theory & Practical part will be based on knowledge bank of questions created by the AA and approved by SSC.
3. Individual assessment agencies will create unique question papers for theory part for each candidate at each examination/training center (as per assessment criteria below)
4. Individual assessment agencies will create unique evaluations for skill practical for every student at each examination/training center based on these criteria.
5. To pass the Qualification Pack, every trainee should score a minimum of 70% cumulatively (Theory and Practical).
6.Final summative assessment is conducted by SSC/AB through NCVET- approved AAs, independent of the training provider. All practicals will be recorded.

Assessment Artefacts:

All learners must produce the following artefacts as part of their practical assessments. These will serve as demonstrable evidence of skill attainment.

S. No	Artefact	Description	Associated Modules
1	**Small C++ Project**	A mini project (console-based or file I/O application) demonstrating OOP concepts like classes, inheritance, polymorphism, and use of pointers. Code should be compiled and submitted with inline comments and screenshots of outputs.	Module 2: OOP using C++
2	**Small Java Project**	A basic Java application implementing classes, inheritance, exception handling, and interfaces. May include GUI elements (optional). Project must include source code, execution screenshots, and exception handling implementation.	Module 3: OOP using Java
3	**Case Study – Code + Documentation**	A structured case-study-based mini project (C++ or Java) with source code, flowcharts, logic description, error handling, and final testing steps. Documented in a standard format (DOC/PDF).	Module 4: Case Study Implementation
4	**Portfolio Submission**	Consolidated folder (physical or digital) containing: - Flowcharts / algorithms - UML class diagrams - Snapshots of working code - Javadoc (for Java) or Doxygen (for C++) - Short write-up on project logic and challenges	Modules 1 to 4 + Employability Skills

Additional Notes:

- Trainers must ensure that each artefact is evaluated using a checklist and signed off.
- Portfolios will be assessed during the final evaluation by the Assessment Agency.
- Students should be encouraged to **upload their projects to GitHub** as part of **Employability Skills**.

Coding

Assessment Governance

1. **Awarding Body (AB):**
   - The Awarding Body (AB) shall be the relevant Sector Skill Council (SSC) approved under NCVET (for IT-ITeS, this is NASSCOM/IT-ITeS SSC).
   - The AB is responsible for certification and quality assurance.
2. **Assessment Agencies (AAs):**
   - Summative assessments will be conducted only through NCVET-recognized Assessment Agencies (AAs).
   - These agencies are independent of the Training Provider to ensure fairness and transparency.
3. **Assessment Methodology:**
   - **Theory Assessments:** Written/online tests covering Knowledge Units (KU).
   - **Practical Assessments:** Hands-on coding tasks, projects, and case studies aligned to Performance Criteria (PC).
   - **Weightage:** 40% theory, 60% practical, as per QP design.
4. **Video Recording:**
   - 100% of practical assessments will be video-recorded with clear audio as per SSC/NCVET requirements.
   - Recordings will be securely stored and made available for audit/verification.
5. **Passing Criteria:**
   - Minimum 70% score at the QP level is required for certification.
   - Re-assessment shall be permitted as per SSC/NCVET guidelines.
6. **Role of Training Provider:**
   - The Training Provider shall deliver training but has **no role in final certification or assessment governance**.
   - TP is responsible only for preparing candidates, maintaining infrastructure, and facilitating assessments conducted by independent AAs.
7. **Compliance Assurance:**
   - All assessment practices will follow NCVET regulations, SSC Assessment SOPs, and data privacy/security norms.
   - Continuous monitoring and audits by the SSC/AB will ensure adherence to standards.

Coding

Performance Criteria:

Job Role	Trainee Associate				
Assessment Outcomes	Assessment criteria for outcomes		Marks Allocation		
		Total Marks	Out of	Theory	Skills Practical
MES/NSU/N0902: Basic Concepts of Object Oriented Paradigms	PC1. Understand, the basic computer and internet literacy including operating a computer, describing its major components and how they work, using C++ and JAVA Programming,	100	30	10	50
	PC2.Understand, aptitude for analyzing information and making logical conclusions. In sufficient hands-on experience to be able to describe		20	10	
	PC3.Demonstrate knowledge of the foundational mathematical concepts in computing		10	10	
	PC4.Design algorithms to solve problems and convert them into source code using the appropriate Object-Oriented programming language constructs.		20	10	
	PC 5: Drawing flow charts for the pictorial representations of an algorithm to know the logical structure of a source code.		15	5	
	| PC6: Apply basic OOP concepts using Python and/or C# through small examples and exercises.		5	5	
		Total	100	50	50

Coding

Assessment Outcomes	Assessment criteria for outcomes		Marks Allocation		
		Total Marks	Out of	Theory	Skills Practical
MES/NSU/N0903: Turbo C++ Software	PC1: Demonstrate a strong understanding of C++ syntax, data types, control structures, and function definitions	100	10	5	50
	PC2: Proficiency in OOP principles, including classes, objects, inheritance, polymorphism, and encapsulation		25	15	
	PC3: Able to manage memory effectively, including understanding pointers, dynamic memory allocation, and avoiding memory leaks		20	10	
	PC4: Ability to read and write data to files using C++ streams.		20	5	
	PC5: Implementing robust error handling mechanisms, such as exceptions or error codes		25	15	
		Total	**100**	**50**	**50**

Coding

Assessment Outcomes	Assessment criteria for outcomes		Marks Allocation		
		Total Marks	Out of	Theory	Skills Practical
MES/NSU/N0904: Java Software	PC1: Demonstrate understanding and correct usage of Java syntax, including data types, variables, operators, control flow statements (if, else, for, while), and methods.	100	25	15	50
	PC2: Understand and apply OOP principles like classes, objects, and their relationships, encapsulation, inheritance, polymorphism, and abstraction.		25	10	
	PC3: Utilize interfaces and abstract classes appropriately.		10	5	
	PC4: Identify and handle exceptions effectively using try-catch blocks.		20	10	
	PC5: Understand and implement multithreading and Applet concepts.		15	5	
	PC6: Apply digital literacy skills in a software development context, including secure coding and cyber hygiene practices.		5	5	
		Total	**100**	**50**	**50**

Coding

Assessment Outcomes	Assessment criteria for outcomes		Marks Allocation		
		Total Marks	Out of	Theory	Skills Practical
MES/NSU/N0905: Case study implementation based on C++/Java Coding	PC1: Discover C++ / JAVA based project ideas for beginners to improve coding skills with hands on experience	100	5	5	50
	PC2: Writing clean, well-documented, and maintainable C++ / JAVA code.		5	5	
	PC3: Ability to break down complex problems into smaller, manageable parts and develop effective C++ / JAVA solutions.		10	5	
	PC4: Ability to work on small to medium-sized C++ / JAVA projects, from requirements gathering to implementation and testing.		50	25	
	PC5: Apply Java or C++ programming skills to solve real-world problems. Write clean, well-structured, and maintainable code.		30	10	
		Total	**100**	**50**	**50**

Assessment Outcomes	Assessment criteria for outcomes		Marks Allocation		
		Total Marks	Out of	Theory	Skills Practical
MES/NSU/N0906: Maintain Workplace Health and Safety	PC1: understand and comply with the organizations current health, safety and security policies and procedures	100	20	10	50
	PC2: understand the safe working practices pertaining to own occupation		20	10	
	PC3: understand the government norms and policies relating to health and safety including emergency procedures for illness, accidents, fires or others which may involve evacuation of the premises		20	10	
	PC4: identify the people responsible for health and safety in the workplace, including those to contact in case of an emergency		20	10	
	PC5: follow organizations emergency procedures for accidents, fires or any other natural calamity in case of a hazard		20	10	
		Total	**100**	**50**	**50**

Coding

Assessment Outcomes	Assessment criteria for outcomes		Marks Allocation		
		Total Marks	Out of	Theory	Skills Practical
DGT/VSQ/N0102 Employability Skills	PC1: Communicate effectively using verbal, non-verbal and written methods.	100	20	10	50
	PC2: Demonstrate time management, goal-setting, and personal development.		20	10	
	PC3Apply digital literacy skills in a software development context.		20	10	
	PC4: Demonstrate financial awareness and use digital finance tools securely.		20	10	
	PC5: Show awareness of entrepreneurship, freelancing, and start-up basics.		15	5	
	PC6 : Demonstrate understanding of basic secure coding principles and apply them in code samples.		5	5	
		Total	**100**	**50**	**50**

Coding

Performance Criteria (PC) and Knowledge & Understanding per Module

Module 1: MES/NSU/N0902 Basic Concepts of Object-Oriented Paradigms

Objectives:

- Introduce the fundamentals of Object-Oriented Programming (OOP).
- Compare OOP with procedural paradigms.
- Build understanding of how multiple programming languages implement OOP.

Performance Criteria (PC):

- PC1: Define and explain core OOP concepts (abstraction, encapsulation, inheritance, polymorphism).
- PC2: Identify limitations of procedural programming.
- PC3: Compare OOP implementations in C++, Java, Python, and C#.
- PC4: Create flowcharts and simple UML diagrams representing OOP structure.

Knowledge:

- Principles of OOP across languages.
- Differences between procedural vs object-oriented paradigms.
- Syntax familiarity with C++, Java, Python (intro), and C# (intro).

Skills:

- Analyze and interpret OOP-based code snippets.
- Write pseudocode applying OOP concepts.
- Create basic flowcharts/UML diagrams.

Learning Outcomes:

- Explain OOP principles (encapsulation, inheritance, polymorphism, abstraction).
- Apply problem-solving techniques using OOP methodology.
- Create simple UML diagrams (class, object, sequence).
- Differentiate between procedural and object-oriented paradigms.

Coding

Module 2: MES/NSU/N0903 Turbo C++ Software

Objectives:

- Deepen understanding of C++ for object-oriented development.
- Develop robust, secure, and modular C++ programs.

Performance Criteria (PC):

- PC1: Use core C++ features: classes, objects, inheritance, polymorphism, encapsulation.
- PC2: Handle memory management and pointer operations securely.
- PC3: Implement file I/O, error handling, and generic programming using templates.
- PC4: Use modern IDEs and compilers for C++ development.
- PC5: (Optional) Apply build automation using Makefile or CMake.

Knowledge:

- Syntax and semantics of C++ language.
- OOP-specific constructs and memory management.
- Error handling and template programming.

Skills:

- Create and debug structured C++ code in Code::Blocks/VS Code.
- Compile projects using GCC/Clang and build tools.
- Practice debugging and exception handling.

Learning Outcomes:

- Develop C++ programs using control structures, functions, and arrays.
- Implement OOP features in C++: classes, constructors, destructors, overloading.
- Apply advanced C++ concepts: pointers, dynamic memory, file handling.
- Create a small C++ project demonstrating modular programming and documentation.

Coding

Module 3: MES/NSU/N0904 Java Software

Objectives:

- Build Java applications using modern OOP concepts and best practices.
- Learn secure coding, GUI basics, and debugging in Java.

Performance Criteria (PC):

- PC1: Understand Java syntax, data types, control structures, and class structure.
- PC2: Apply OOP principles including inheritance, interfaces, and polymorphism.
- PC3: Implement exception handling and input validation.
- PC4: Use collections and generics in data management.
- PC5: Understand GUI development basics using JavaFX/Swing (optional).
- PC6: Demonstrate secure coding practices (e.g., validation, no hardcoded credentials).

Knowledge:

- Java syntax, OOP features, standard libraries.
- Collections, multithreading, and exception hierarchy.

Skills:

- Develop modular Java applications in IntelliJ/Eclipse/VS Code.
- Debug and document Java programs using Javadoc.
- Follow secure and standard coding guidelines.

Learning Outcomes:

- Write Java programs using OOP features (inheritance, interfaces, polymorphism).
- Apply exception handling and multithreading in Java applications.
- Develop GUI-based applications using AWT/Swing/JavaFX.
- Integrate Java programs with external files and libraries.
- Create a Java mini-project demonstrating real-world problem solving.

Coding

Module 4: MES/NSU/N0905 Case Study Implementation (C++/Java)

Objectives:

- Apply knowledge of Java/C++ to real-world mini projects.
- Emphasize structured code, documentation, and testing.

Performance Criteria (PC):

- PC1: Analyze a problem and design a solution using C++ or Java.
- PC2: Break down project into components and implement using modular code.
- PC3: Apply debugging and exception handling.
- PC4: Document logic using comments, flowcharts, and output screenshots.
- PC5: Present project with clarity, justifying design decisions and logic used.

Knowledge:

- Software development life cycle (SDLC) concepts.
- Language-specific implementation nuances (C++ vs Java).

Skills:

- Write clean and maintainable code.
- Use IDEs for project development.
- Perform unit testing and basic validations.

Learning Outcomes:

- Plan and design a coding project using OOP methodology.
- Apply C++ and/or Java programming skills to solve real-world problems.
- Use Git/GitHub for version control and collaboration.
- Document project using flowcharts, UML diagrams, and API documentation (Javadoc/Doxygen).
- Demonstrate teamwork, problem-solving, and presentation skills in project execution.

Coding

Module 5: MES/NSU/N0906 Maintain Workplace Health and Safety

Objectives:

- Promote a culture of safety, security, and health compliance at the workplace.

Performance Criteria (PC):

- PC1: Understand organizational safety protocols.
- PC2: Recognize emergency signals and procedures.
- PC3: Follow safe ergonomic practices.
- PC4: Identify potential hazards and report appropriately.
- PC5: Participate in emergency preparedness drills.

Knowledge:

- Health & Safety Acts, workplace policies.
- Types of risks: fire, electrical, physical injury.

Skills:

- Use emergency tools (fire extinguisher, first aid kit).
- Demonstrate safe workstation setup.
- Comply with safety norms.

Learning Outcomes:

- Apply ergonomic practices to reduce strain and fatigue.
- Follow fire safety, electrical safety, and evacuation procedures.
- Demonstrate cyber hygiene: safe browsing, password management, malware awareness.
- Show awareness of workplace policies on equality, diversity, and harassment.

Coding

Module 6: DGT/VSQ/N0102 Employability Skills

Objectives:

- Build essential soft skills and digital readiness for the IT workplace.

Performance Criteria (PC):

- PC1: Communicate effectively in professional settings.
- PC2: Use digital productivity tools (Docs, Sheets, Email, GitHub, LinkedIn).
- PC3: Demonstrate time and self-management.
- PC4: Understand freelancing and entrepreneurship models.
- PC5: Practice secure coding and online financial safety.

Knowledge:

- Soft skills, team dynamics, freelancing basics.
- Digital tools, cybersecurity, green computing.

Skills:

- Create resumes, cover letters, and personal portfolios.
- Participate in interviews and group discussions.
- Use digital tools for project collaboration.

Learning Outcomes:

- Communicate effectively in oral and written forms in workplace contexts.
- Use digital productivity tools (Word, Excel, PowerPoint, email, online collaboration).
- Prepare resume, cover letter, and perform confidently in interviews.
- Demonstrate teamwork, leadership, and adaptability in group activities.
- Practice financial literacy (budgeting, digital payments) and workplace ethics.

Coding

Element	Details
Objectives	Introduce the fundamentals of Object-Oriented Programming (OOP) and compare it with procedural paradigms.
Performance Criteria (PC)	- PC1: Define core OOP concepts (abstraction, encapsulation, inheritance, polymorphism). - PC2: Identify limitations of procedural programming. - PC3: Compare OOP implementations in C++, Java, Python, and C#.
Knowledge	- Principles of OOP- Syntax familiarity with C++, Java, Python (intro), and C# (intro)- Differences between procedural vs object-oriented paradigms
Skills	- Analyze code snippets- Create flowcharts/UML diagrams- Write pseudocode using OOP logic
Learning Outcomes	- Explain and apply key OOP concepts- Differentiate between OOP and procedural logic using real code- Recognize how different languages implement OOP

Coding

Career Pathways

This program prepares candidates for entry-level roles in the IT software development domain. Upon successful completion and gaining relevant experience, candidates can progress along the following career trajectory:

Current Role	Next Level Role	Typical Experience Required	Key Skills to Develop
Trainee Associate	Junior Programmer	6 months to 1 year	Core programming, debugging, version control
Junior Programmer	Software Developer	1 to 3 years	Advanced coding, software design, OOP mastery
Software Developer	Senior Developer	3 to 5 years	System architecture, code reviews, mentoring
Senior Developer	Software Architect	5+ years	High-level system design, project leadership

---

Additional Notes:

- **Trainee Associate:** Focus on learning foundational programming skills, debugging, and basic project work.
- **Junior Programmer:** Gains responsibility for small features or bug fixes in real projects.
- **Software Developer:** Handles full module development, code optimization, and integration.
- **Senior Developer:** Leads teams, designs complex modules, and ensures code quality.
- **Software Architect:** Defines system architecture, technology standards, and guides overall software strategy.