COMPUTER PROGRAMMING (JAVA)

TECHNICAL-PROFESSIONAL

Grade 12

Three-Term Budget Of Work (BOW)
for Learning Competencies

TECHNICAL-PROFESSIONAL
ICT SUPPORT AND COMPUTER PROGRAMMING TECHNOLOGIES
Grade 12

Three-Term Budget Of Work
For Learning Competencies

Please note that each Tech-Pro elective is completed within one year if delivered at Grade 12.

Elective	**Computer Programming (Java)**
Prerequisite	**None**
Content Standard	The learners demonstrate an understanding of Java programming, including basic structure, syntax, data types, variables, and comments. They also develop skills in creating algorithms, flowcharts, and pseudocode to plan and visualize their code before implementation.
Performance Standard	The learners perform procedures in setting up the Integrated Development Environment (IDE) and in creating algorithms, flowcharts, and pseudocode to plan and visualize their code.

Week	Learning Competency	Suggested Activities
1	• Discuss the fundamental concepts of Java	**Java Concepts Quick Map** o Learners identify the key Java concepts such as program structure, syntax, data types, variables, and comments. o Make a simple concept map showing how these elements appear in a basic Java program. **Java Program Walk-Through** o Learners explain the fundamental parts of a simple Java program. o Through a given a sample code, briefly describe the purpose of each component (class, main method, variables, comments).

Week	Learning Competency	Suggested Activities
	• Perform the steps in setting up the development environment	**Java Tools Setup Check** o Install the Java Development Kit (JDK), open the IDE, and check if Java runs correctly by viewing the version or welcome screen. **First Project Test Run** o Learners confirm that the development environment works properly. o Learners create a new Java project in the IDE and run a simple sample program to see if it executes without errors.
	• Discuss Java basic structures and syntax	**Parts of a Java Program** o Learners identify the basic parts of a Java program. o Learners are given a simple Java code and label the class name, main method, variables, and comments, then briefly explain what each part does. **Syntax Spot-the-Error** o Learners recognize correct and incorrect Java syntax. o Learners review short Java code samples and point out missing semicolons, brackets, or wrong keywords, then correct them.
	• Develop algorithms for designing a program or system	**Everyday Steps to Algorithm** o Learners learn how an algorithm works using a familiar

Week	Learning Competency	Suggested Activities
		activity. o Learners choose a simple daily task (e.g., logging in to a computer) and write the step-by-step instructions in correct order as an algorithm. **Problem-to-Algorithm Challenge** o Learners create an algorithm to solve a simple programming problem. o Given a basic problem (e.g., display the sum of two numbers), learners list the steps needed to solve it using clear and logical sequence.
2	• Develop flowcharts for designing a program or system	**From Steps to Flowchart** o Learners convert simple steps into a flowchart. o Learners take a given step-by-step solution (algorithm) and draw the correct flowchart using basic symbols (start, process, decision, end). **Flowchart a Simple Problem** o Learners design a flowchart for a basic programming problem. o Given a simple task (e.g., display the larger of two numbers), learners draw a flowchart showing the logical flow from start to end
	• Develop pseudocodes for designing a program or system	**Algorithm to Pseudocode** o Learners translate a simple solution into pseudocode. o Given a basic problem (e.g., display the sum of two

Week	Learning Competency	Suggested Activities
		numbers), learners write the steps in plain-English pseudocode using correct sequence and keywords. **Pseudocode Fix-It Task** o Learners improve a given pseudocode to make it logical and complete. o Learners review a short pseudocode with missing or incorrect steps, then revise it so the logic is clear and ready for coding.
Content Standard	The learners demonstrate an understanding of Java Operators, Input/Output (I/O) classes, flow control structures, and arrays to solve real-world problems.	
Performance Standard	The learners create a well-structured program using Java Operators, input and output classes, flow control structures, and arrays.	
2	• Apply operators in writing a Java program	**Relational Operators** o Learners write a Java program that calculates total expenses and averages using arithmetic operators. o Learners apply relational and logical operators to compare expenses with a set budget (e.g., over/under budget). **Conditional Salary Checker** o Learners develop a program that assigns salary values and applies arithmetic operators to compute allowances or deductions. o Learners use relational and logical operators to determine eligibility for benefits based on salary conditions.
3	• Apply Java I/O classes to read and write data files in Java programs	**Learner's Record File Reader** o Learners develop a program that assigns salary values and applies arithmetic operators to compute allowances or deductions.

Week	Learning Competency	Suggested Activities
		o Learners use relational and logical operators to determine eligibility for benefits based on salary conditions. **Transaction Log Write** o Learners develop a program that accepts user input for transaction details. o Learners write the collected data into a file using FileWriter/BufferedWriter.
	• Apply Java Flow Control Structures in writing a Java program	**Menu-Driven Program Using Selection Statements** o Learners design a menu-based program using if-else or switch statements. o Learners implement user input processing to execute selected operations. **Loop-Based Number Analyzer** o Learners write a program that accepts multiple numbers using loops (for/while). o Learners compute totals, counts, or averages based on loop processing.
	• Apply Java Arrays in writing a Java program	**Learner's Score Manager Using Arrays** o Learners create an array to store multiple student scores. o Learners perform operations such as displaying scores and computing averages. **Inventory Tracker with Array Storage** o Learners use arrays to store product names and quantities.

Week	Learning Competency	Suggested Activities
		o Learners update and display inventory data using array elements.
	• Apply sorting algorithms in a Java program that organizes customer transaction data stored in an array	**Customer Transaction Sorter (Ascending and Descending)** o Learners write a Java program that applies a basic sorting algorithm (e.g., bubble sort) to arrange customer transaction amounts stored in an array. o Learners test the program using different sets of transaction data and verify the correctness of the sorted results (ascending and descending). **Transaction History Organizer** o Learners develop a Java program that sorts customer transaction records (IDs or amounts) stored in an array to generate an organized transaction history report. o Learners display the sorted transaction records in a formatted output
Suggested Performance Tasks	**Individual Activities:** • **Set Up for Coding Success** ○ Configure and set up an Integrated Development Environment (IDE) with the required tools and settings for coding tasks. ○ • **Plan Before You Code** ○ Create an algorithm, flowchart, and pseudocode to visualize the solution to a given programming problem.	

Week	Learning Competency	Suggested Activities
	Group Activities: • **IDE Setup Squad** ○ Collaboratively install and configure an IDE, ensuring all group members can build and run a sample program. • **Code Design Challenge** ○ Work as a team to develop and present an algorithm, flowchart, and pseudocode for a given problem-solving scenario.	
Content Standard	The learners demonstrate an understanding of object-oriented Java programming principles, procedures to manage errors with exception handling, and the Java Collections Framework for efficient data management.	
Performance Standard	The learners create a well-structured and efficient program using Java flow control structures, Java arrays, and Java Object-Oriented Programming (OOP) language.	
4	• Discuss Object-Oriented Programming (OOP) Language in Java	**OOP Concept Mapping in Java** o Learners create a concept map explaining classes, objects, and OOP principles. o Learners present their concept map and explain each OOP concept using Java examples. **Real-World Object Identification Activity** o Learners identify real-world entities and describe their classes, attributes, and methods. o Learners convert one identified entity into a simple Java class structure.

Week	Learning Competency	Suggested Activities
	• Apply OOP principles in a Java program	**Cass Design for a Simple Banking System** o Learners write Java classes applying encapsulation and inheritance to model basic bank accounts. o Learners test the class by creating sample objects and displaying account details. **Polymorphism Demonstration Program** o Learners implement method overriding in Java to demonstrate polymorphism using a parent and child class. o Learners run the program and compare outputs of parent and child methods.
5	• Apply exception handling in the Java program	**Input Validation with Try-Catch Blocks** o Learners develop a Java program that handles invalid user inputs using try-catch statements. o Learners test the program using invalid inputs and observe error handling behavior. **Custom Exception Creation Activity** o Learners create and apply a user-defined exception to handle logical errors in a Java program o Learners integrate the custom exception into a program and trigger it under defined conditions.
	• Apply Java Collections Framework in writing Java programs	**Learner List Management Using Array List** o Learners use ArrayList to store, add, remove, and display student records in a Java program.

Week	Learning Competency	Suggested Activities
		o Learners implement a search function to locate a specific student record. **Unique Data Organizer Using Set and Map** o Learners implement Set or Map collections to manage unique IDs or key-value data. o Learners display data entries and explain how duplicates are handled.
6	• Apply the java.io package and java file operations in a Java program	**Text File Creator Using java.io Classes** o Learners create and save text files using java.io classes. o Learners input sample data (e.g., name and age) and save it into the created file. **File Update and Append Activity** o Learners develop a Java program that appends new data to an existing file using Java file operations. o Learners verify that new entries are appended without overwriting existing data.
	• Apply reading and writing classes in writing a Java program	**Buffered File Reading Exercise** o Learners read data from a text file using BufferedReader and display the output in the console.

Week	Learning Competency	Suggested Activities
		o Learners modify the output format (e.g., labeled or numbered lines) for better readability. **Data Writing Using FileWriter and BufferedWriter** o Learners write structured data to a file using Java writing classes. o Learners organize data into a structured format (e.g., records with labels).
	• Apply threads in creating a Java program using both the Thread class and Runnable interface	**Thread Creation Using Thread Class** o Learners create a simple multithreaded Java program by extending the Thread class. o Learners observe and describe the execution sequence of multiple threads. **Runnable-Based Multitasking Program** o Learners implement the Runnable interface to run multiple tasks concurrently in a Java program. o Learners compare the output when running single vs. multiple concurrent tasks.
Suggested Performance Tasks	**Individual Learning Activities:** • **Code It Smart** Develop a Java program applying flow control structures and arrays to solve a given problem efficiently.	

Week	Learning Competency	Suggested Activities
	• **Object-Oriented Builder** Create a Java program using classes and objects that demonstrates proper application of OOP principles. **Group Activities:** • **Java Program Design Challenge** Collaboratively design and implement a structured Java program integrating control structures, arrays, and OOP concepts. • **Code Showcase** Work as a team to develop, test, and present an optimized Java program demonstrating logical flow and organized code design.	
Content Standard	The learners showcase their grasp of Java's multithreading principles, graphical user interface (GUI) development, and efficient component structuring to build interactive and responsive applications. They incorporate market analysis to ensure their solutions meet industry demands and customer needs.	
Performance Standard	The learners will develop a creative GUI-based Java application that demonstrates innovative solutions to real-world problems	
7	• Create a Java program that effectively integrates CountDownLatch, CyclicBarrier, Semaphore	**Traffic Control Simulation** o Learners design a program where multiple threads request access to a shared resource using Semaphore. o Learners modify the program to simulate limited access (e.g., only 2 threads at a time) and observe behavior. **Team Task Synchronization**

Week	Learning Competency	Suggested Activities
		o Learners create a program where threads complete tasks and wait using CountDownLatch before continuing. o Learners simulate group coordination by synchronizing threads using CyclicBarrier.
	• Create simple GUI applications by using JFrame, JLabel, JButton, and JTextField while handling events with Action Listeners	**Simple Login Form** o Learners create a GUI login interface using JFrame, JLabel, JTextField, and JButton. o Learners add an ActionListener to validate correct/incorrect input and display messages. **Basic Calculator App** o Learners design a calculator GUI with input fields and operation buttons. o Learners implement button actions to perform addition, subtraction, multiplication, and division.
7	• Create user interfaces in Java using layout managers such as FlowLayout, BorderLayout, and GridLayout, while effectively nesting components within multiple Jpanels	**Form Layout Design** o Learners construct a registration form using GridLayout for structured input fields. o Learners group related components using nested JPanels for better organization. **GUI Dashboard Layout** o Learners create a window divided into sections using BorderLayout**.** o Learners add buttons arranged using FlowLayout within a panel.

Week	Learning Competency	Suggested Activities
8	• Create interactive Java applications by leveraging advanced UI components and handling complex user interactions to enhance user experience and interface responsiveness	**Interactive Quiz Application** o Learners create a simple quiz interface with multiple-choice questions. o Learners implement event handling to check answers and display scores. **Real-Time Form Validator** o Learners design a form that captures user input (e.g., email, password). o Learners implement real-time validation feedback using event listeners.
9	• Develop business plans for a system proposal	**Problem-Solution Mapping** o Learners analyze a real-world problem and describe it briefly. o Learners propose a system-based solution and outline key features. **Pitch Deck Development** o Learners prepare a short presentation of their system proposal. o Learners present the target users, benefits, and feasibility of their idea.
10	• Create designs of a GUI Java application showcasing creativity to solve real-world problems	**Prototype Sketching** o Learners draw a basic GUI layout (paper or digital) addressing a real-world problem. o Learners label components (buttons, fields, panels) and describe their functions.

Week	Learning Competency	Suggested Activities
		GUI Concept Presentation o Learners present their GUI design to peers. o Learners explain layout choices, features, and how the interface solves the problem.
Suggested Performance Tasks	**Individual Learning Activity : My Java Workspace Setup** ○ Learners independently set up the Java IDE and prepare their workspace for coding. ○ Learners install the Java Development Kit (JDK), open and configure the IDE, then create and save a new Java project to confirm the setup is ready for coding. **Group Activity : Code Planning Team Lab** ○ Learners work in groups to plan a simple program before coding. ○ In small groups, learners select a simple problem, agree on the solution steps, and create an algorithm, flowchart, and pseudocode to clearly visualize the program logic.	

** Teachers may assign learning competencies per week depending on the needs of the learners.*