Unit 3 Software Development 2025

Outcome 1 Software development: programming – Template for developing an assessment task – Plan

Outcome 1

On completion of this unit the student should be able to interpret teacher-provided solution requirements and designs and use appropriate features of an object-oriented programming language to develop working software modules.

Assessment task development

Create a scenario that is a real-world example that provides students with solution requirements and designs to enable them to develop four working software modules using appropriate features of an object-oriented programming language. At least two modules must include a GUI and all modules must include testing. Key content within the assessment task should be based on the targeted key knowledge and key skills. The total number of marks for the outcome is to be out of 100.

Key knowledge	Key skills	
• emerging trends in programming using artificial intelligence, including: – using prompts to generate code – automated debugging and testing of modules – code optimisation – responsible and ethical use of artificial intelligence tools • characteristics of functional and non-functional requirements, constraints and scope • design tools for representing modules, including: – data dictionaries – mock-ups – object descriptions – input-process-output (IPO) charts – pseudocode	• interpret solution requirements and designs	Content to be included in the assessment task should introduce students to a scenario. The scenario should provide students with solution requirements and designs for four modules. A range of appropriate design tools should be used. Students are not to complete designs themselves. Software modules can be small programs that may or may not form part of a larger software solution.
• characteristics of data types, including: – text (character, string) – numeric (integer, floating point, date/time) – Boolean • characteristics of data structures, including: – one-dimensional arrays – two-dimensional arrays – records (varying data types, field index) • characteristics of data sources (plain text (TXT), delimited (CSV) and XML files), including: – structure – reasons for use	• use a range of data types, data structures and data sources	The scenario with the solution requirements and designs should enable students to determine what data types, data structures and data sources they will need to use for the software modules.

Unit 3 Software Development 2025 Outcome 1 Software development: programming – Template for developing an assessment task – Plan		
• principles of OOP, including: – abstraction – encapsulation – generalisation – inheritance • features of a programming language, including: – local and global variables, and constants – data types – instructions and control structures (sequence, selection, iteration/repetition) – arithmetic, logical and conditional operators – graphical user interfaces (GUIs) – functions and methods – classes and objects • algorithms for sorting and searching, including: – selection sort – quick sort – binary search – linear search	• use and justify appropriate features of an OOP language to develop working software modules	The scenario with the solution requirements and designs should enable students to determine the appropriate selection and use of features of an OOP language and the use of sorting and searching algorithms they will need to develop the working software modules. An appropriate OOP language is to be used by the students. Students are to justify and explain their selection of features and sorting and searching algorithms used to develop their working modules. This written justification and explanation could be included within the internal documentation or as a separate written report.
• purposes and features of naming conventions for solution elements (variables, interface controls, code structures), including: – Hungarian notation – camel casing – snake casing • validation techniques for data, including: – existence checking – type checking – range checking	• develop and apply suitable naming conventions and validation techniques within modules	Students are to apply suitable naming conventions and validation techniques when developing the software modules.
• purposes of internal documentation, including: – explaining and justifying data and code structures – code maintenance – placeholder comments for future development (stubs)	• document the functioning of modules using internal documentation	Students are to write internal documentation within their working software modules. Internal documentation should state how the modules function and describe the code involving processing and validation.
• types of errors, including: – syntax – logic – runtime (overflow, index out of range, type mismatch, divide by zero) • debugging and testing techniques for checking modules function correctly, including: – use of breakpoints – use of debugging statements – construction of relevant test data	• develop and apply suitable debugging and testing techniques using appropriate test data	They are to design a testing table that involves the testing of all validation, objects and processing, such as calculations, etc. The testing table should also include columns for the expected and actual output and show evidence of tests that work and don't work. Suitable debugging techniques should be applied to ensure all the tests of the software modules meet the solution requirements.

Unit 3 Software Development 2025 Outcome 1 Software development: programming – Template for developing an assessment task – Plan		
– test cases comparing expected and actual output in testing tables		