IQM3010

LEVEL 3 · CONTROLLED ASSESSMENT

Computer Science

Assessment 3 Applied coursework brief

EcoQuest Learning

Duration
Approximately 20 supervised hours

Total marks
100

Assessment window
Sample assessment

Assessment set
Set S of 2026

Audience
Candidates

Release timing
At coursework release

Assessment overview

All three tasks must be completed for EcoQuest Learning during approximately 20 supervised hours. A Python terminology game, a small relational database and an Excel analysis must be produced. The products share the scenario but may operate separately.

The assessment carries 100 marks and contributes one third of the module grade. Approximately 2,000 words equivalent must be submitted across the tasks, including tables, diagrams, screenshots and selected code extracts. Working files and testing evidence are also required.

Task	Product	Marks
1	Python environmental terminology game	70
2	Two-table relational clue database	15
3	Excel game test-data analysis	15

Client scenario

EcoQuest Learning develops educational games for schools. A text-based prototype is required to help GCSE pupils revise environmental science terminology. Each round displays a clue and a hidden term. Letters are guessed until the term is complete or six incorrect guesses have been made. A neutral title and remaining-attempts counter must be used.

The company also requires a database to store its science terms and clues, and a spreadsheet to summarise game-session data. The three products may operate separately; Python does not need to connect to the database or spreadsheet.

The assessment set is Set S of 2026. The centre will confirm the supervised schedule, submission deadline and submission route.

Task 1 Python environmental terminology game

70 marks. A working console-based game must be produced. Python's standard library may be used. A graphical interface and inheritance are not required.

Required game behaviour

- All 12 supplied term and clue pairs must be available for selection. The supplied file contains data only; the game must be implemented by the candidate.
- A term must be selected randomly from the unused terms in the current session. Its matching clue and one placeholder per letter must be displayed.
- Six incorrect guesses must be available in each round. One letter must be accepted at a time, ignoring case and surrounding spaces. Blank, non-letter and multiple-character inputs must be rejected without using an attempt.
- Every occurrence of a correctly guessed letter must be revealed. An attempt must be deducted only for a new incorrect letter. Repeated correct or incorrect guesses must receive a useful message without a further penalty.
- Guessed letters, the partially revealed term and remaining attempts must be displayed accurately. A round must end when the term is complete or no attempts remain. The outcome and correct term must then be displayed.
- A further round or an exit option must be offered. A term must not be selected again during the same session.
- At least one purposeful class must manage a round's state, using a constructor and methods that perform game actions. Suitable functions or methods, meaningful names and maintainable code must be used.

Illustrative behaviour

The following sequence illustrates one round for the term habitat. The actual game must select its term randomly.

Input	Required result
Start	Display the matching clue and seven placeholders; 6 incorrect guesses remain.
7, blank or AA	Invalid input; no attempt used.
T	_ _ _ _ t _ t; both occurrences of t are revealed; 6 attempts remain. Case and surrounding spaces are ignored.
z	No new letters are revealed; 5 attempts remain.
z again	Repeated incorrect guess; no further attempt is used.
A	_ a _ _ t a t; both occurrences of a are revealed; 5 attempts remain.
h, then b	h a b _ t a t; 5 attempts remain. Previously revealed letters stay visible.
i	h a b i t a t; the term is complete. Display the win and correct term; offer another round or exit.

Task 1 report and testing evidence

Analysis 10 marks

The client, intended users and purpose must be identified. Measurable success criteria, inputs, processing, outputs, validation and practical constraints must be specified. The development approach must be explained. One benefit and one limitation of a digital revision game must be considered, including its effect on access to or communication of learning information.

Design 15 marks

The solution must be decomposed into suitable components. Pseudocode or flowcharts must show the main logic, validation, game states and replay process. Class attributes, constructor, methods and data structures must be described. A test plan must specify inputs and expected results for normal, boundary and invalid cases.

Implementation 25 marks

The working program must demonstrate all required behaviour. Important implementation decisions must be explained and justified. Source files must remain readable and maintainable. Selected extracts may be included in the report, but the complete source files must also be submitted.

Testing and debugging 10 marks

Expected and actual results must be recorded with supporting evidence. Tests must cover correct and incorrect guesses, repeated letters in a term, repeated guesses, case and space handling, blank/non-letter/multiple-character input, the final permitted incorrect guess, a win, a loss, replay, exit and prevention of term reuse. Any defects, corrections and retests must be recorded. Accurate evidence of a failed test is preferable to an unsupported claim of success.

Evaluation and maintenance 10 marks

The evaluation must be approximately 500 words as part of the overall project report. It must judge the completed program against the success criteria using test results, and discuss efficiency, usability, limitations and possible improvements. Two realistic maintenance changes must be explained, including how each would be implemented and tested.

GenAI conditions

Level 2 AI planning applies. AI may support brainstorming, outlining, planning and initial research. Submitted code, analysis, testing evidence, interpretation and evaluation must be the candidate’s own work. AI-generated final code, conclusions or unverified references must not be presented as the candidate’s work. Any permitted AI use must be acknowledged; a nil declaration is required where no AI was used.

Task 2 Simple relational clue database

15 marks. A working relational database must be created in a suitable database application. It must contain a Clues table imported from the supplied EcoQuest_Clues.csv file and a Games table. Prepare a short report with evidence that the database meets the requirements below.

Use the field names and structure shown below. The supplied CSV has no header row and uses this field order: ClueID, Word, Clue, Difficulty, Subject. The Games table must contain exactly two fields: GameID and ClueID.

Table	Required fields	Key
Clues	ClueID; Word; Clue; Difficulty; Subject	Choose a suitable primary key and explain why it is suitable.
Games	GameID; ClueID	GameID is the primary key. ClueID links a game to a record in Clues.

Creation and import

- Import all 12 records from the headerless EcoQuest_Clues.csv file into Clues, using the supplied field order. Preserve each term, clue, difficulty and subject value.
- Use suitable data types and field sizes so no supplied text is truncated. Require a value in each of the five Clues fields and use suitable validation messages.
- Choose a suitable primary key for Clues and explain why it identifies each record. Set GameID as the primary key in Games. Link Games.ClueID to Clues.ClueID and explain the purpose of this relationship.
- Restrict Subject to Ecology, Resources or Energy. Difficulty must accept integers from 1 to 5 inclusive. Reject blank values in all five Clues fields and provide suitable messages when invalid values are entered.
- Do not add, remove or alter the supplied records.

Testing evidence

Testing evidence must show that valid values are accepted and that an invalid subject, difficulty values outside 1 to 5, a non-integer difficulty and blank required values are rejected. Record the validation messages or equivalent evidence.

Submit the working database file and a short evidence report covering the Clues and Games structures, the primary key, the ClueID relationship and validation tests. Include the database testing evidence.

Task 3 Excel game test-data analysis

15 marks. EcoQuest Learning has collected simulated game-testing results for the 12 supplied terms. Use the supplied EcoQuest_Word_Test_Data.xlsx workbook to identify words whose clues or difficulty ratings may need review.

Games played is the number of rounds in which a word was attempted. Games won is the number of those rounds completed successfully. Difficulty ranges from 1, the easiest rating, to 5, the hardest rating.

Spreadsheet requirements

- Keep the supplied data and add a Win percentage column. Use a formula for every word to calculate games won as a percentage of games played. Display results to one decimal place.
- Use formulas or functions to calculate total games played, total games won and the average win percentage (the unweighted mean of the 12 per-word percentages).
- All data must be sorted by Win percentage from lowest to highest, keeping each complete row together.
- Recommendations must be recorded in an identifiable area of the workbook. Words requiring review must be identified by comparing win percentages and difficulty ratings. Recommendations must be supported by specific data, with an explanation of why the results do not establish the cause conclusively.
- One limitation of using a spreadsheet to model a real-life situation must be explained in context.

Submit the completed workbook with working formulas, recommendations and one limitation. Separate analysis and design reports are not required. Do not change the supplied data.

How the marks are allocated

Evidence	Marks
Python analysis	10
Python design	15
Python implementation	25
Python testing and debugging	10
Python evaluation and maintenance	10
Relational database	15
Spreadsheet	15
Total	100

Submission and supplied resources

Required submission

- Completed coursework cover sheet and AI-use declaration.
- One PDF report containing the required Python analysis, design, testing, evaluation and maintenance evidence, plus the short database evidence report and database testing evidence.
- Complete Python source file or files.
- The working relational database and any companion files required to open it.
- The completed Excel workbook, including formulas, recommendations and the contextual limitation.
- References and acknowledgements for external sources and any permitted AI use.

Candidate resource files

Resource	Purpose
IQM3010_A3_2026-S_EcoQuest_Terms.py	12 term and clue pairs for the game
IQM3010_A3_2026-S_EcoQuest_Clues.csv	12 headerless clue records: ClueID, Word, Clue, Difficulty, Subject
IQM3010_A3_2026-S_EcoQuest_Word_Test_Data.xlsx	12 raw game-testing records

Before submission

All working files must open correctly. The report and native files must use the centre’s candidate identification and filename convention. Personal data are not required in the game, database or spreadsheet. The source data supplied for this sample must remain unchanged. The centre will confirm the deadline and submission route.