Higher coursework assessment task

Qualifications Scotland
Teisteanasan Alba

Higher Computing Science Assignment Assessment task

This document provides information for teachers and lecturers about the coursework component of this course in terms of the skills, knowledge and understanding that are assessed. It must be read in conjunction with the course specification.

Valid for session 2025–26 only.

This assessment is given to centres in strictest confidence. You must keep it in a secure place until it is used.

This edition: January 2026 (version 1.0)

Contents

Introduction

This document contains instructions for teachers and lecturers, and instructions for candidates for the Higher Computing Science assignment. You must read it in conjunction with the course specification.

This assignment has 40 marks out of a total of 120 marks available for the course assessment.

This is one of two course assessment components. The other component is a question paper.

Instructions for teachers and lecturers

This assessment applies to the assignment for Higher Computing Science for the academic session 2025–26.

The task is valid for 2025-26 only. Once complete, you must send the assignment responses to Qualifications Scotland to be marked.

You must conduct the assignment under a high degree of supervision and control. This means:

- candidates must be supervised throughout the session(s)
- candidates must not have access to email or mobile phones
- candidates must complete their work independently — no group work is permitted
- candidates must not interact with each other
- with no interruption for targeted learning and teaching
- in a classroom environment

You can use any integrated development environments (IDEs) that enables candidates to generate evidence — this includes online IDEs. However, IDEs must have a facility that prevents candidates accessing their files and tasks outside the supervised classroom environment.

Time

Candidates have 6 hours to carry out the assignment, starting at an appropriate point in the course, after all content has been delivered. It is not anticipated that this is a continuous 6-hour session, although it can be, but conducted over several shorter sessions. This is at your discretion.

You have a responsibility to manage candidates' work, distributing it at the beginning and collecting it in at the end of each session, and storing it securely in between. This activity does not count towards the total time permitted for candidates to complete the assignment.

Candidates are prompted to print their work at appropriate stages of the tasks. They can print on an ongoing basis or save their work and print it later. Whatever approach they take, time for printing is not part of the 6 hours permitted for the assignment.

Resources

Each candidate must have access to a computer system with a high-level (textual) programming language and **either**:

- a database application or software that can create, edit and run SQL
- software that can create

This is an open-book assessment. Candidates can access resources such as programming manuals, class notes, textbooks and programs they have written throughout the course. These may be online resources.

You must not create learning and teaching tasks that make use of constructs required in the assessment task, **with the specific purpose of developing a solution that candidates can access during the assignment.**

You can provide candidates with templates, however these templates must only contain general starter code used in learning and teaching (for example, a web page that contains the HTML, title and body elements) — templates must not be tailored to this year's task.

There may be instances where restriction of network use is prohibited (for example, a local authority-managed network with specific limitations). However, it remains your professional responsibility to make every effort to meet the assessment conditions.

Reasonable assistance

The assignment consists of three independent tasks. They are designed in a way that does not require you to provide support to candidates, other than to ensure that they have access to the necessary resources. Candidates can complete the tasks in any order.

Once the assignment is complete, you must not return it to the candidate for further work to improve their mark. You must not provide feedback to candidates or offer an opinion on the perceived quality or completeness of the assignment response, at any stage.

You can provide reasonable assistance to support candidates with the following aspects of their assignments:

- printing, collating and labelling their evidence to ensure it is in the format specified by Qualifications Scotland
- ensuring any pages that do not contain candidate evidence are discarded and page order matches the order of the tasks
- ensuring candidates have all the materials and equipment required to complete the assignment — this includes any files provided by Qualifications Scotland
- ensuring candidates understand the conditions of assessment and any administrative arrangements around the submission and storage of evidence, and the provision of files
- technical support

Evidence

All candidate evidence (whether handwritten or created electronically) must be submitted to Qualifications Scotland in a paper-based format. The evidence checklist details all evidence to be gathered. You can use it to ensure you submit all evidence to Qualifications Scotland.

You should advise candidates that evidence, especially code, must be clear and legible. This is particularly important when pasting screenshots into a document

There is no need for evidence to be printed single-sided or in colour.

If evidence is handwritten, candidates must use a blue or black pen.

When packaging, ensure that:

- each assignment is accompanied by an Qualifications Scotland A4 flyleaf
- all completed task sheets and additional evidence of screenshots or printouts from development environments are included and ordered in line with the task
- sheets that do not contain candidate evidence are removed
- candidates' SCNs are on every page
- the flyleaf and candidate evidence are stapled together in the top left corner

Alteration or adaptation

The tasks are in PDF and Word formats. Each task is available as a separate file from the secure site. Word files allow candidates to word process their responses to parts of the task.

You must not adapt the assignment in any way that changes the instructions to the candidate and/or the nature and content of the tasks. However, you can make changes to font size, type and colour and to the size of diagrams for candidates with different assessment needs, for example, visual impairment.

If you are concerned that any particular adaptation changes the nature and/or the content of the task, please contact our Assessment Arrangements Team for advice as soon as possible at aarequests@qualifications.gov.scot.

Submission

Each page for submission has the number of the assignment task that it refers to, for example 1a, and contains space for candidates to complete their candidate number. Any other pages submitted, for example, prints of program listings or screenshots, must have this information added to them.

Only submit pages that contain candidate evidence. Ensure the page order matches the order of the task

Specific instructions for teachers and lecturers: 2025–26

All candidates must complete task 1 (software design and development) and **either** task 2 (database design and development) **or** task 3 (web design and development).

It is at your discretion how you approach this optionality in assessment. The task your candidates complete might be pre-determined by your progress through the course, or you may be able to let candidates choose which task to complete.

You must follow these specific instructions and ensure that candidates are aware of what you will give them at each stage in the assessment.

Print each task on single-sided paper, where applicable:

- this allows candidates to refer to information on other pages
- this helps you manage tasks that are split into more than one part

Task 1 — part A requires candidates to analyse a problem. They must submit their evidence to you before you issue part B.

Task 1 — part B is a separate section. This ensures that candidates do not access part A and change their responses. Part B requires candidates to design a section of code. They must submit their evidence to you before you issue part C.

Task 1 — part C is a separate section. This ensures that candidates do not access part B and change their responses. Part C requires candidates to implement a program following a design, test an algorithm given and evaluate their code.

Candidates must still have access to the problem description pages during parts B and C.

A CSV file (tools.csv) is provided for candidates to use in part C. You must not convert the CSV file into a different format. Candidates are assessed on their ability to implement a solution to the given task, using the specific file type provided.

Task 2 — part A requires candidates to analyse a problem and complete an entity-occurrence diagram. They must submit their evidence to you before you issue part B.

Task 2 — part B is a separate section. This ensures that candidates do not access part A and change their responses.

A Microsoft Access file (flickNest.accdb) is provided for candidates to use in part B. If your centre uses a different database management system, you can create the relational database using the CSV files or the text files provided.

If using the CSV files, you should set up all tables, fields and validation shown in the data dictionaries below. Referential integrity should also be enforced.

The text files contain SQL create and insert statements for each table. If you use the text files, you must add validation (shown in the data dictionaries below), appropriate for your version of SQL. Referential integrity should also be enforced.

The CSV files contain the data for each table. The text files contain SQL create and insert statements for each table.

In both cases, you will have to add primary keys and foreign keys as specified below. The field 'premiumAccount' in the Customer table is created as a Boolean. If required, you should set this up as an integer with '0' or '1'.

Entity: Customer

Attribute name	Key	Type	Size	Required	Validation
customerID	PK	Text	20	Yes	
password		Text	30	Yes	
forename		Text	30	Yes	
surname		Text	30	Yes	
email		Text	50	No	
premiumAccount		Boolean		No	

Entity: Purchase

Attribute name	Key	Type	Size	Required	Validation
customerID	PK/ FK	Text	20	Yes	Existing customerID from Customer entity
movieCode	PK/ FK	Text	20	Yes	Existing movieCode from Movie entity
purchaseDate		Date		Yes	
rating		Number		No	

Entity: Movie

Attribute name	Key	Type	Size	Required	Validation
movieCode	PK	Text	20	Yes	
title		Text	50	Yes	
released		Number		Yes	
duration		Number		Yes	
price		Number		Yes	
genreID	FK	Text	20	Yes	Existing genreID from Genre entity

Entity: Genre

Attribute name	Key	Type	Size	Required	Validation
genreID	PK	Text	20	Yes	
genreName		Text	30	Yes	
description		Text	250	No	

Task 2(e) requires candidates to test an SQL statement. You must provide this to candidates as part of the database. The SQL statement is already included in the MS Access file. If you use a different database management system, you should use the supplied text file ('Query for 2e.txt') to add it to the database you provide to candidates.

Task 3 — part A requires candidates to design part of a website. They must submit their evidence to you before you issue part B.

Task 3 — part B is a separate section. This ensures that candidates do not access part A and change their responses.

Candidates must still have access to the page containing the end-user and functional requirements during part B.

A folder named 'Web files' is provided. This contains the CSS, HTML and media files candidates need to complete this task. These files must not be renamed, and they must remain in the folders provided. However, the case of suffixes may be changed if the environment you work in requires them to be lower or upper case.

Instructions for candidates

This assessment applies to the assignment for Higher Computing Science.

This assignment has 40 marks out of a total of 120 marks available for the course assessment.

It assesses the following skills, knowledge and understanding:

- applying aspects of computational thinking across a range of contexts
- analysing problems within computing science across a range of contemporary contexts
- designing, implementing, testing and evaluating digital solutions (including computer programs) to problems across a range of contemporary contexts
- demonstrating skills in computer programming
- applying computing science concepts and techniques to create solutions across a range of contexts

Your teacher or lecturer will let you know if there are any specific conditions for doing this assessment.

In this assessment, you have to complete two short practical tasks.

You must complete task 1 (software design and development) and **either** task 2 (database design and development) **or** task 3 (web design and development).

You may complete the tasks in any order.

Advice on how to plan your time

You have 6 hours to complete the assignment. Marks are allocated as follows:

- Task 1 — software design and development 25 marks (63% of total)

  AND EITHER
- Task 2 — database design and development 15 marks (37% of total)

  OR
- Task 3 — web design and development 15 marks (37% of total)

You can use this split as a guide when planning your time for each of the two tasks.

Advice on gathering evidence

As you complete each task, you must gather evidence as instructed.

Your evidence, especially code, must be clear and legible. This is particularly important when you paste screenshots into a document. You can print code from the software environment or copy and paste this into other packages such as notepad or Word.

Use the evidence checklist provided to make sure you submit everything necessary at the end of the assignment. Make sure you include your candidate number on all your evidence.

Evidence may take the form of printouts of code, screenshots, typed answers, handwritten answers or drawings of diagrams and designs.

You must use a blue or black pen for any handwritten answers.

Only submit pages that contain your answers. Order the pages in the same order as the task.

Advice on assistance

This is an open-book assessment. This means that you can use:

- any classroom resource as a form of reference (for example programming manuals, class notes, and textbooks) — these may be online resources
- any files you have previously created throughout the course

The tasks are designed so you can complete them independently, without any support from your teacher or lecturer. This means that you:

- cannot ask how to complete any of the tasks
- cannot access any assignment files outside the classroom

Computing Science assessment task: evidence checklist

You should complete the checklist for task 1 and **either** task 2 **or** task 3.

Task 1 — software design and development

Task	Evidence	Tick
1a	Completed task sheet identifying missing functional requirements	
1b	Completed task sheet showing completed design	
1c	Printout of your completed program code, program output and 'lateTools.csv' file	
1d(i)	Completed task sheet showing completed table	
1d(ii)	Completed task sheet explaining why the design does not produce the expected results	
1e	Completed task sheet evaluating usability	

Task 2 — database design and development

Task	Evidence	Tick
2a	Completed task sheet showing functional requirements	
2b	Completed task sheet showing completed entity-occurrence diagram	
2c	Printout of the implemented SQL statement(s)	
	Printout of the output produced	
2d	Printout of the implemented SQL statement	
	Printout of the output produced	
2e	Printout of the amended SQL statement	
	Printout of the output produced	
2f	Completed task sheet explaining the reason why the requirement cannot be met	

Task 3 — web design and development

Task	Evidence	Tick
3a	Completed task sheet showing the design of the navigation structure of the website	
3b	Printout of the edited 'rooms.html' file	
3c	Printout of the edited 'challenge.html' and/or 'styles.css' file(s), highlighting changes	
3d(i)	Printout of the edited 'contact.html' file	
3d(ii)	Completed task sheet describing how to fully test the 'Select room(s)' form element	
3e	Completed tasks sheet explaining why the website is not fit for purpose	

Please follow the steps below before handing your evidence to your teacher or lecturer:

- Check you have completed all parts of task 1 and **either** task 2 **or** task 3.
- Label any printouts and screenshots with the task number (for example 1a, 2a).
- Clearly display your candidate number on each printout.
- Discard any pages that do not contain evidence.
- Order the pages in the same order as the task.

Task 1: software design and development

Problem description

EquipYou is a company that rents out DIY tools. The manager would like a program to process data on the tools rented and calculate late fees for tools that have not been returned.

Purpose

A CSV file stores the following data about the 120 tools rented from January 2025 until January 2026 (inclusive):

- Tool name
- Tool manufacturer
- Date rented
- If it has been returned
- Late fee — a fee of 0 is stored when a tool is hired

This data will be read into parallel arrays.

A sample of the data from the file is shown below:

```
Hammer,HandyTech Solutions,03/01/2025,Yes,0
Screwdriver,BuildPro Systems,05/01/2025,Yes,0
Wrench,BrownTech,07/01/2025,Yes,0
…
Work Gloves,HandyTech Solutions,23/12/2025,No,0
Knee Pads,FixIt Co.,05/01/2026,No,0
Respirator,TitanForge,12/01/2026,No,0
```

The program will allow the user to enter a manufacturer's name. It will search for that manufacturer and display the name of each tool and the total number of tools from that manufacturer.

An example of the program's output is shown below:

```
Enter a manufacturer: BrownTech
Wrench
Oscillating Multi-Tool
Caliper
Total: 3
```

The program will calculate a late fee for tools rented in 2025 that have not been returned and write the tool name, date rented and late fee to an external CSV file.

Assumptions

- The data is formatted correctly and is error free, and stores information about 120 different tools.
- The file stores information about each tool's most recent rental.
- The date will be stored as a string.

Task 1: software design and development (part A)

1a Using the problem description, complete the functional requirements for the program.

(3 marks)

Input(s) ♦ chosen manufacturer
Process(es) ♦ calculate late fee
Output(s) ♦ name of each tool by chosen manufacturer ♦ total number of tools from chosen manufacturer ♦ write tool name, date rented and fee of those with late fee to external file

♦ Check your answers carefully, as you cannot return to part A after you hand it in.
♦ When you are ready, hand part A to your teacher or lecturer and collect part B.

Candidate number_____

Task 1: software design and development (part B)

The top-level design for the main steps of the program is shown below.

1	Read file into parallel arrays.	OUT: tool(), manufacturer(), dateRented(), returned(), fee()
2	Find and display the name of each tool and the total number of tools by a chosen manufacturer.	IN: tool(), manufacturer()
3	Calculate late fee for tools rented in 2025 and not returned.	IN: dateRented(), returned(), fee() OUT: fee()
4	Write the tool name, date rented and fee of any tool with a late fee to an external file.	IN: tool(), dateRented(), fee()

1b Late fees are charged for any tool rented in 2025 and not returned. The fees to be applied are shown below:

Month rented	Fee (£)
January – June	10
July - December	5

Using a design technique of your choice, design a function for step 3.

(3 marks)

- Check your answers carefully, as you cannot return to part B after you hand it in.
- When you are ready, hand part B to your teacher or lecturer and collect part C.

Candidate number_____

Task 1: software design and development (part C)

The top-level design for the main steps of the program (with partial refinements) is shown below.

1	Read file into parallel arrays.	OUT: tool(), manufacturer(), dateRented(), returned(), fee()
2	Find and display the name of each tool and the total number of tools by a chosen manufacturer.	IN: tool(), manufacturer()
3	Calculate late fee for tools rented in 2025 and not returned.	IN: dateRented(), returned(), fee() OUT: fee()
4	Write the tool name, date rented and fee of any tool with a late fee to an external file.	IN: tool(), dateRented(), fee()

Refinements

```
3.1   loop for number of tools
3.2       if tool was rented in 2025 and not returned
3.3           if the month is between January and June
3.4               set current fee to 10
3.5           else
3.6               set current fee to 5
3.7           end if
3.8       end if
3.9   end loop
3.10  return fee array

4.1   open file 'lateTools.csv' to write
4.2       loop for number of tools
4.3           if current fee is not 0
4.4               write tool name, date rented and fee to file
4.5           end if
4.6       end loop
4.7   close file 'lateTools.csv'
```


1c Using the problem description and design, implement the program in a language of your choice. Your program should:

- read data from the 'tools.csv' file to parallel arrays
- use a function to calculate the late fees for tools rented in 2025 and not returned
- use procedures to:
  - find and display the name of each tool and the total number of tools by a chosen manufacturer
  - write the tool name, date rented and fee of any tool with a late fee to an external file named 'lateTools.csv'
- be maintainable and modular
- be tested using the chosen manufacturer 'BrownTech'

(15 marks)

Expected output

```
Enter a manufacturer: BrownTech
Wrench
Oscillating Multi-Tool
Caliper
Total: 3
```

Print evidence of:

- your program code
- program outputs from your test run
- 'lateTools.csv' file

Include your candidate number on all evidence.

1d New code will be required to calculate late fees for tools rented in 2026 that have not been returned. Tools rented in the first 6 months of 2026 and not returned will be charged £10, and those rented after that and not returned will be charged £5. Tools rented in 2025 and still not returned will be charged £20, plus the late fee for 2026.

The design to calculate late fees for tools rented in 2026 is shown below.

```
3.1   loop for number of tools
3.2       if tool was rented in 2025 or 2026 and not returned
3.3           set current fee to 20
3.4           if the month is between January and June
3.5               add 10 to current fee
3.6           else
3.7               add 5 to current fee
3.8           end if
3.9       end if
3.10  end loop
3.11  return fee array
```

When tested with the data below, it was found that the design above would not produce the correct result.

Test data

```
Tool1,Manufacturer1,04/02/2025,Yes,0
Tool2,Manufacturer2,23/10/2025,No,5
Tool3,Manufacturer3,26/01/2026,No,0
Tool4,Manufacturer2,27/12/2026,No,0
```

(i) Using the design and test data above, complete the table below to show the values for each tool after each iteration.

(2 marks)

Current tool name	If tool rented in 2025 or 2026 and not returned?	Fee (£)
Tool1	False	0
Tool2		
Tool3		
Tool4		

Candidate number_____

(ii) Explain why this design does not produce the correct result.

(1 mark)

1e The program is tested using the manufacturer name 'BroTech' instead of 'BrownTech'.

Evaluate the usability of your solution when the name of a manufacturer not in the file is entered.

(1 mark)

Candidate number_____

Task 2: database design and development (part A)

FlickNest is an online movie service that allows customers to purchase movies.

A database is required to store data on customers, movies, genres and customer purchases.

Potential end-users of the database have provided the following list of requirements:

- 'I would like to view genres with the word 'suspense' in the description.'
- 'I need to be able to view the number of times each movie has been purchased.'
- 'I want to see the average customer rating of each movie.'
- 'I would like to be able to find the highest rated movie by a specific director.'
- 'I would like to see the names of customers who have a premium account and the total amount of money they spent on purchases.'
- 'I would like to see the movie with the shortest duration.'

2a Assuming all the entities and attributes have been identified, use the end-user requirements above to create two additional functional requirements.

(2 marks)

Functional requirement 1
Functional requirement 2

Candidate number _____

2b The sample data in the following tables show:

- customers who purchased a movie
- genre of each movie
- movies purchased

Customer	Purchase
Customer2	Purchase2
Customer3	Purchase4
Customer1	Purchase1
Customer2	Purchase3

Movie	Genre
Movie1	Genre2
Movie3	Genre2
Movie2	Genre1

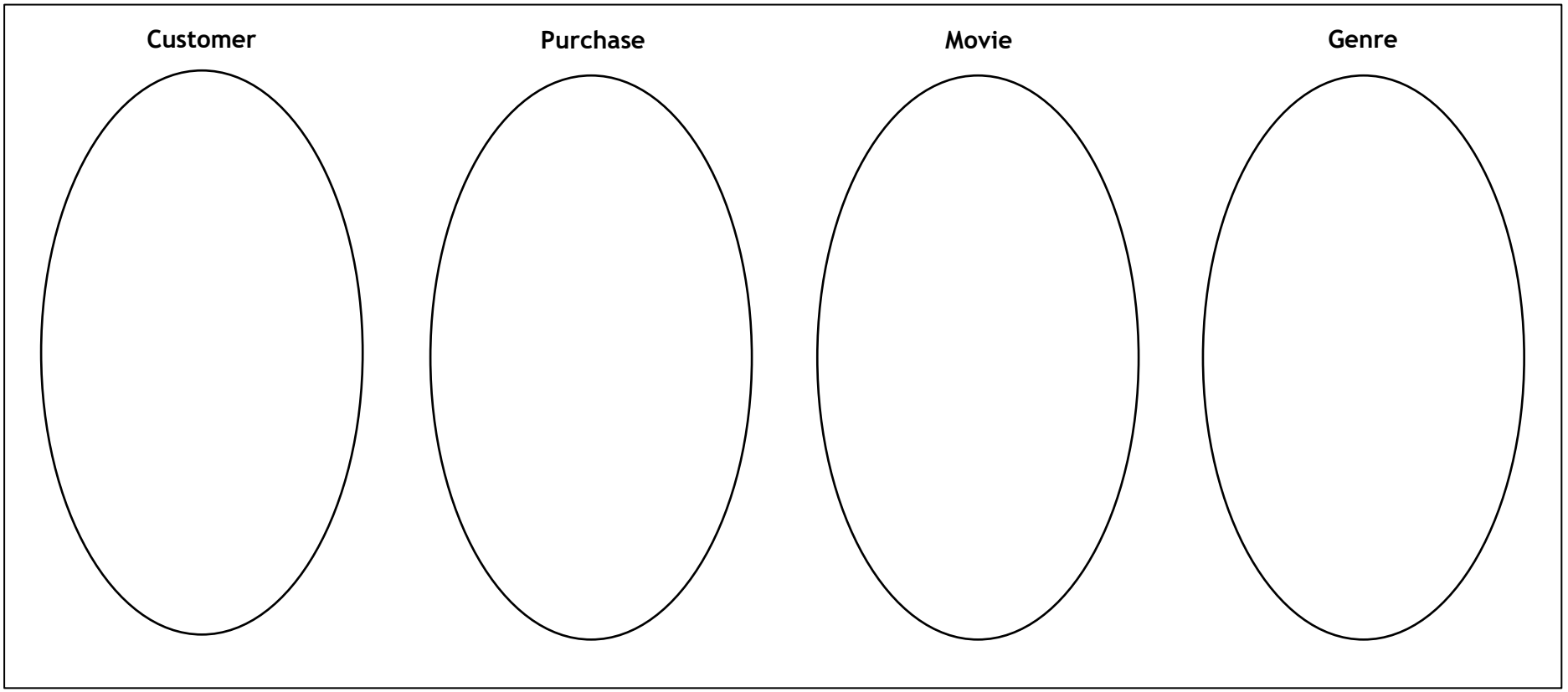
Purchase	Movie
Purchase4	Movie3
Purchase2	Movie2
Purchase3	Movie3
Purchase1	Movie1

Using this sample data, complete the entity-occurrence diagram on the following page by:

- adding the sample instances for each entity
- showing the association between the instances

(2 marks)

Entity-occurrence diagram



- Check your answers carefully, as you cannot return to part A after you hand it in.
- When you are ready, hand part A to your teacher or lecturer and collect part B.

Candidate number _____

Task 2: database design and development (part B)

Your teacher or lecturer will provide you with a completed relational database with the following tables.

flickNest database:

Customer	Purchase	Movie	Genre
customerID	customerID*	movieCode	genreID
password	movieCode*	title	genreName
forename	purchaseDate	released	description
surname	rating	duration	
email		price	
premiumAccount		genreID*	

2c FlickNest would like to know which customers purchased the movie with the shortest duration.

Implement the SQL statement(s) to display the forename and surname of customers who purchased the movie with the shortest duration.

The expected output is shown below.

(3 marks)

forename	surname
Ruby	Grant
Marcus	McGowan
Aleksander	Novak
Violet	Rose

Print evidence of the implemented SQL statement(s) and the output produced.

2d FlickNest would like to see the names of customers who have purchased 'Comedy' movies and the total amount of money each customer spent on these movies.

Implement an SQL statement that will display the forename and surname of customers, and the total each customer has spent on comedy movies, with the highest total spent first.

The expected output is shown below.

(5 marks)

forename	surname	Total spent(£)
Derek	Tsang	52.95
Rebecca	Shearer	32.96
Daisy	Robertson	19.99
Fatima	Hussein	9.98

Print evidence of the implemented SQL statement and the output produced.

2e A query is required to find and display the forename, surname and email address of the customers who have purchased any film released in the 1990s.

The following SQL statement is used:

```
SELECT forename, surname, email
FROM customer, purchase, movie
WHERE customer.customerID = purchase.customerID
AND movie.movieCode = purchase.movieCode
AND released = 1990;
```

A query to test the above SQL statement is provided with the database. When run, the output appears to be incorrect.

Amend the query by making the required changes to produce the correct output.

The expected output is shown below.

(2 marks)

forename	surname	email
Caleb	Evans	calev@inbox.com
Derek	Tsang	itsyaboy@snapmail.org
Isaac	Turner	turnerisaac@flashmail.net
Jasmine	Owens	jayoh@inbox.com
Marcus	McGowan	mercutio@cloudmail.co.uk
Rebecca	Shearer	becks@inbox.com
Stephanie	Ferguson	zimmy674@snapmail.org
Theo	Harrison	matthew.harrison2@cloudmail.co.uk
Violet	Rose	slayallday@snapmail.org
Xavier	Garcia	xavierxox@inbox.com

Print evidence of the amended SQL statement and the output produced.

2f When evaluating the end-user requirements from the analysis stage, it was found that the following requirement cannot be met using the current database structure.

'I would like to be able to find the highest rated movie by a specific director'

Explain, with reference to the database structure, why this requirement cannot be met.

(1 mark)

Candidate number _____

Task 3: web design and development

Breakout Lab offers escape room experiences with different themes and difficulty levels.

A web developer is commissioned to produce the website. On completing the analysis stage, the developer identifies the following end-user and functional requirements.

End-user requirements

Visitors to the Breakout Lab website want to:

- learn about the escape room experience
- find out about the different themed rooms
- find information on the different challenge levels (Novice, Amateur, Expert)
- read quotes about other visitors' experiences
- book an experience using an online form

Functional requirements

- A navigation bar with links to the main pages of the website
- A 'Home' page:
  - displaying introductory text welcoming users to the Breakout Lab website
  - displaying an image of one of the rooms
- A 'Rooms' page:
  - providing detailed descriptions of the Jungle and Carnival escape rooms
  - with hover effects revealing additional information about each room
- A 'Challenge' page:
  - displaying three challenge levels (Novice, Amateur, Expert) with related descriptions
  - with links from each challenge level to its associated page for further details
- Each challenge level page contains:
  - details about the duration, difficulty and suitability of the challenge
  - customer quotes for that specific challenge level
  - a link to the 'Contact us' page for making bookings
- A 'FAQs' page:
  - providing answers to frequently asked questions, such as 'What is an escape room?' and 'How can I book a room?'
  - with a link to the 'Contact us' page
- A 'Contact us' page including a form for customers to:
  - enter their name, email and party size
  - select a room(s)
  - select a challenge level
  - provide details of any special requirements
- The footer is consistent across all pages, displaying the text "Breakout Lab 2026"

Task 3: web design and development (part A)

3a The website will have a multi-level navigation structure, consisting of a 'Home' page with a horizontal navigation bar linking four main pages: 'Rooms', 'Challenge', 'FAQs' and 'Contact us'.

The 'Challenge' page will include links to three sub-pages: 'Novice', 'Amateur' and 'Expert'.

Design a multi-level navigation structure for this website, clearly showing the navigation bar, main pages and associated sub-pages.

(3 marks)

- Check your answers carefully, as you cannot return to part A after you hand it in.
- When you are ready, hand part A to your teacher or lecturer and collect part B.

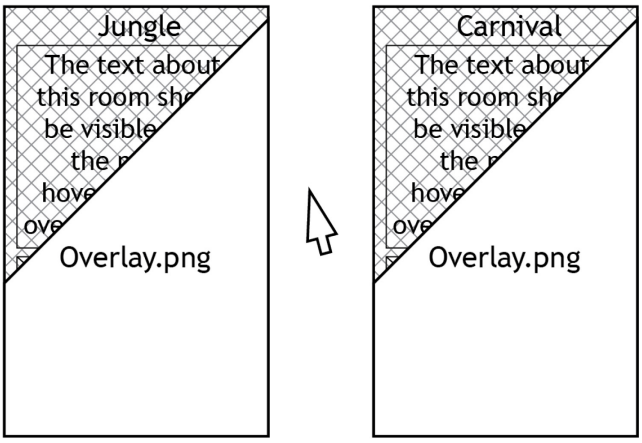
Candidate number_____

Task 3: web design and development (part B)

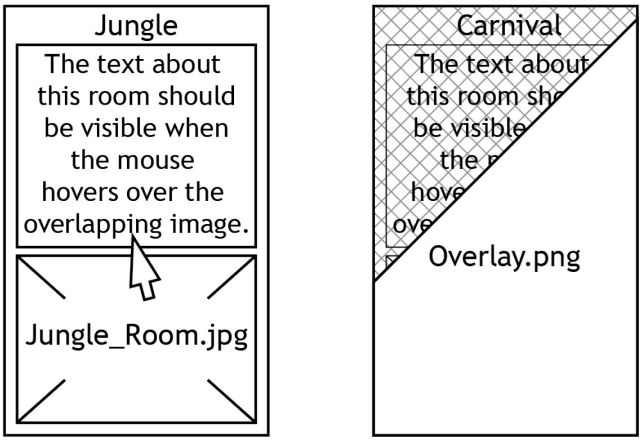
Your teacher or lecturer will provide you with a copy of the incomplete Breakout Lab website.

Open the 'home.html' file in a browser. Examine the home page and each of the other pages on the website.

3b The information for each room on the 'Rooms' page is partially obscured by the image 'Overlay.png'.



When the mouse hovers over the overlay image, it should be hidden to reveal the underlying information and the image of the room, as illustrated in the wireframes below.



Open the 'rooms.html' file in a suitable editor.

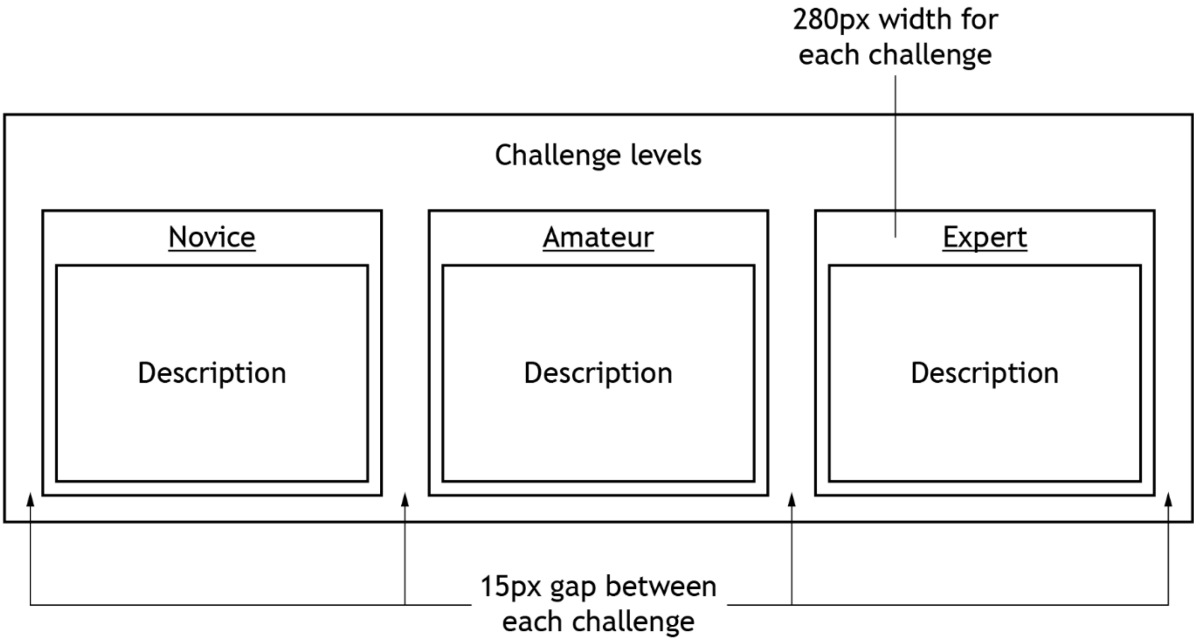
Edit the 'rooms.html' file by adding events to:

- hide the overlay image when the mouse moves over it
- show the overlay image when the mouse moves away from it

(3 marks)

Print evidence of the edited 'rooms.html' file.

3c The wireframe for the ‘Challenge’ page is shown below.



Open the ‘challenge.html’ file in a browser.

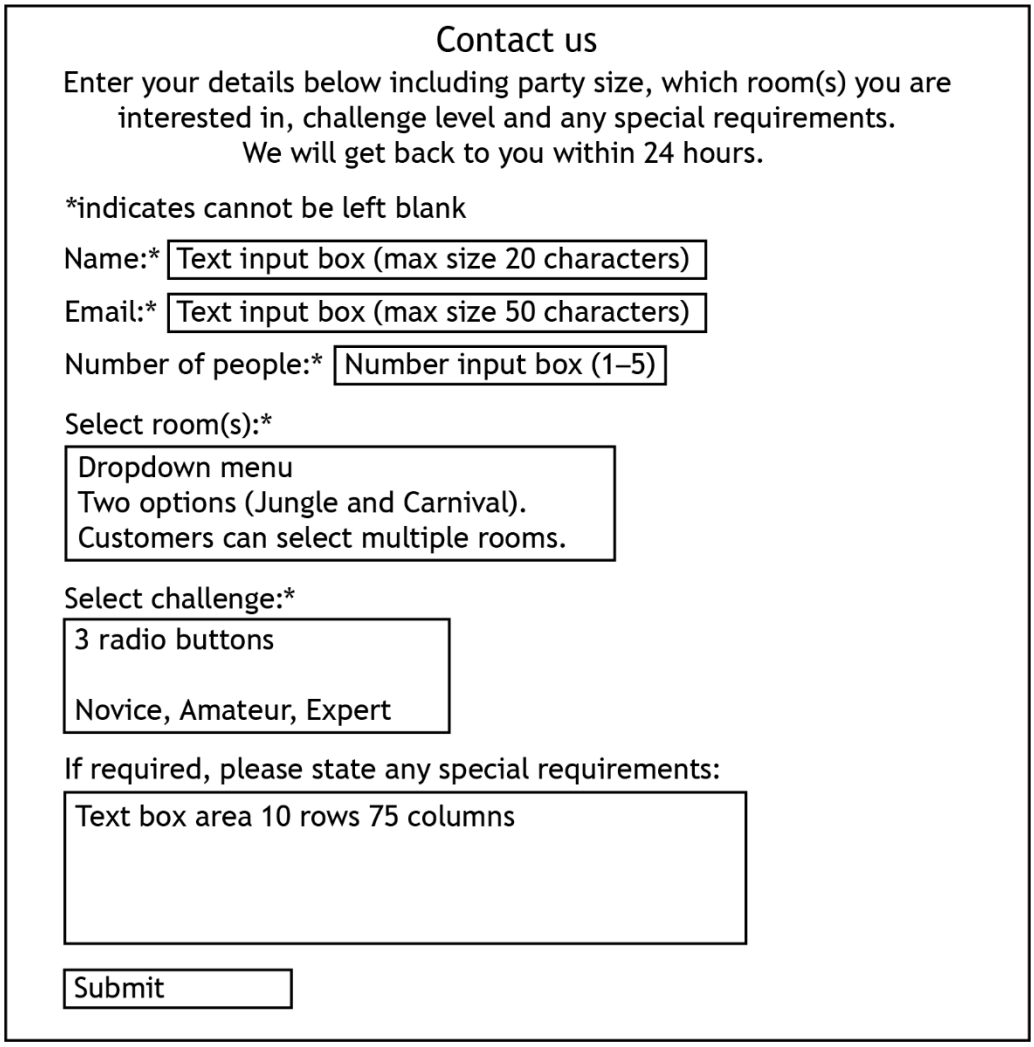
The three challenge levels have not been displayed side-by-side, as shown in the wireframe above.

Edit the code to make the required changes to match the wireframe. You may choose to edit the ‘challenge.html’, ‘styles.css’ files or both.

(2 marks)

Print evidence of any files you have edited to make the changes, highlighting the changes you have made.

3d The final design for the form on the 'Contact us' page is shown below.



(i) Edit the 'contact.html' file to implement the form, ensuring all validation is included and the layout matches the above design.

(4 marks)

Print evidence of the edited 'contact.html' file.

(ii) Describe how you would fully test the 'Select room(s)' form element.

(1 mark)

3e State two reasons why the Breakout Lab website is not fit for purpose.

(2 marks)

Candidate number_____

Administrative information

Published: January 2026 (version 1.0)

History of changes

Version	Description of change	Date

Security and confidentiality

This document can be used by Qualifications Scotland approved centres for the assessment of National Courses and not for any other purpose.

This document may only be downloaded from Qualifications Scotland's designated secure website by authorised personnel.

