Higher
Coursework
Assessment Task

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 2021-22 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 2022 (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 2021–22.

The task is valid for 2021–22 only. Once complete, you must send the assignment responses to SQA 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

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 and software that can create, edit and run SQL
- software that can create, edit and run HTML, CSS and JavaScript

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.**

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 SQA
- ensuring candidates have all the materials and equipment required to complete the assignment — this includes any files provided by SQA
- 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 created manually or electronically) must be submitted to SQA 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 SQA.

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.

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 possible at aarequests@sqa.org.uk.

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 name and candidate number. Any other pages submitted, for example, prints of program listings or screenshots, must have this information added to them.

Specific instructions for teachers and lecturers: 2021–22

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, as this:

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

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

A text file 'mammals.txt' is provided for candidates to use in this part of the task.

Task 1 — 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 problem description page during part B.

Task 2 — part A requires candidates to analyse and design a database. 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 (WestFifeWalkers.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 for part B using the CSV files provided.

If your centre uses an SQL server, you can create the database using the text files provided. These files contain SQL create and insert statements for each table.

Specific instructions for database setup
The 'WestFifeWalkers' database includes table names, field names, primary keys and foreign keys.

You do not need to add validation to any of the fields in the database tables.

WestFifeWalkers database			
Planner	**Route**	**Walk**	**Walker**
plannerNo	routeID	walkID	walkerNo
forename	distance	routeID*	forename
surname	woodName	walkerNo*	surname
telNo	footwear	walkDate	street
regYear	description		town
	difficulty		postcode
	plannerNo*		telNo

Task 2d 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 2d.txt) to add it to the database you provide to candidates.

Task 3 — part A requires candidates to analyse and design 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.

A folder (Walkers Website) has been provided. This contains the CSS, HTML and the image files candidates need to complete this task. These files must not be renamed and they must remain in the folders provided.

Candidates **do not** need to print completed web pages in colour.

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 may complete the tasks in any order.

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

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 in each task.

Your evidence, especially code, must be clear and legible. This is particularly important when you paste screenshots into a document.

Use the evidence checklist provided to make sure you submit everything necessary at the end of the assignment. Ensure your name and candidate number is included on all your evidence.

Evidence may take the form of printouts of code/screenshots/typed answers, hand-written answers or drawings of diagrams/designs.

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		
Part A evidence		
1a	Completed task sheet stating two assumptions	☐
1b	Completed task sheet showing a design of the function to convert the user's inputs to strings that start with an upper-case character	☐
Part B evidence		
1c	Printout of your completed program code	☐
	Printout of your output showing the number of sightings for each date in the text file	☐
1d	Completed task sheet describing how a watchpoint could be used	☐
1e	Completed task sheet evaluating efficiency and maintainability	☐

Task 2		
Part A evidence		
2a	Completed entity-occurrence diagram showing the entity names, instances and associated relationships	☐
Part B evidence		
2b	Printout of the SQL statement(s) to display the planners and the number of times their route(s) has been walked	☐
	Printout of the output produced	☐
2c	Printout of the SQL statement(s) to find all walkers who have walked the current longest route	☐
	Printout of the output produced	☐
2d	Printout of the amended SQL statement and the output produced	☐
2e	Completed task sheet identifying a functional requirement that cannot be implemented	☐

Task 3		
Part A evidence		
3a	Completed task sheet showing two end-user requirements and one functional requirement	☐
3b	Completed task sheet showing the completed wireframe	☐
Part B evidence		
3c and 3d	Printout of edited 'dean.html' file and 'styles.css' file showing code	☐
	Printout of the 'Dean' page as viewed in a browser	☐
	Printout of 'gallery.html' file showing code	☐
	Printout of 'Gallery' page as viewed in a browser	☐
3e(i)	Completed task sheet with description of testing	☐
3e(ii)	Completed task sheet evaluating the fitness for purpose of the form	☐

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

- Check you have completed all parts of tasks 1, and either task 2 or task 3.
- Label any printouts/screenshots with the task number (for example 1a, 2a).
- Clearly display your name and candidate number on each printout.

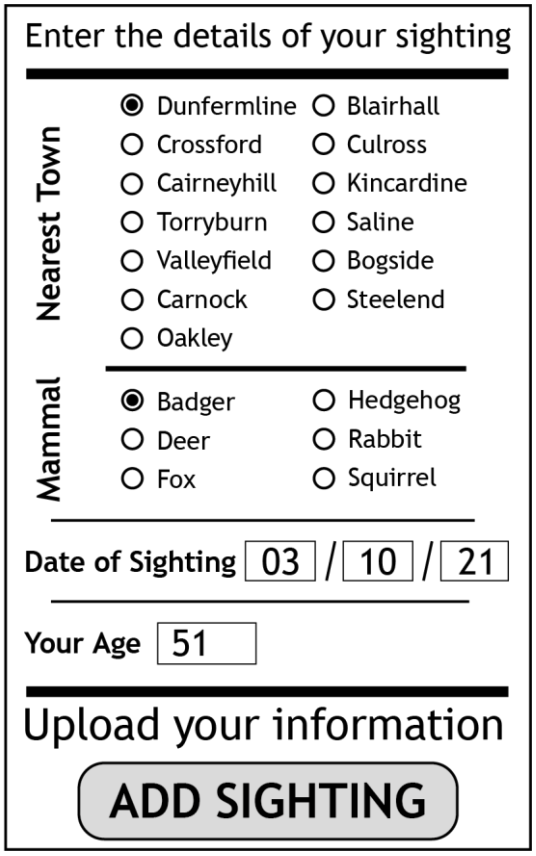
Task 1: software design and development (part A)

Problem description

During the month of September, West Fife Walkers asked their walkers to enter sightings of different Scottish mammals into an existing app. When the user selects the 'add sighting' button, the app checks the data is valid and either:

- adds the valid sighting as a new line in a single text file stored on a server
  or
- responds with an error stating the inputs are missing or invalid

The app's user interface is shown below.



Purpose

A program is now required to analyse the data stored in the text file. The program should:

- display the age of the oldest person to add a sighting
- display the dates of sightings of a chosen mammal in a particular town
- count and display the number of sightings for each date in the text file.

1a State two assumptions that can be made about the data in the text file.

(2 marks)

Candidate name_____ Candidate number_____

1b Your teacher or lecturer will provide you with a text file called 'mammals.txt'.

- The text file has data for 200 sightings in September
- Each line of the text file stores the nearest town to the sighting, the name of the mammal seen, the date of sighting and the age of the walker, as shown below:

```
Saline,Rabbit,01/09/21,79
Culross,Squirrel,01/09/21,42
Dunfermline,Hedgehog,01/09/21,55
Kincardine,Rabbit,01/09/21,41
Dunfermline,Squirrel,01/09/21,89
Crossford,Squirrel,01/09/21,24
Culross,Squirrel,03/09/21,94
Saline,Squirrel,03/09/21,28
…
```

- The sightings in the text file are sorted in order of date from 01/09/21 to 30/09/21.

A top-level design for the main steps of the sightings program is shown below. Data read from the text file is stored in an array of records in the program.

Program top-level design (pseudocode)

1	Read from text file into sightings array of records	OUT: sightings(town,mammal,date,age)
2	Find and display the age of the oldest walker in the sightings data	IN: sightings(town,mammal,date,age)
3	Find and display the dates of sightings of a chosen mammal in a particular town	IN: sightings(town,mammal,date,age)
4	Count and display the number of sightings for each date in the text file	IN: sightings(town,mammal,date,age)

Refinements for step 3 of the design are shown below.

```
3.1   Ask user to enter town
3.2   Call a function to return a string input that starts with an upper-case character
3.3   Ask user to enter mammal
3.4   Call a function to return a string input that starts with an upper-case character
3.5     Display "The dates of sightings were:"
3.6     Start loop for each sighting in array of records
3.7           If sighting matches entered town and mammal then
3.8                 Display date
3.9           End if
3.10  End loop
```

Steps 3.1 and 3.3 indicate that the user enters a string. These two inputs will be compared with the file data in step 3.7.

The town and mammal values stored in the text file all start with an upper-case character. If the user enters a string starting with a lower-case character a logic error will occur as the comparison at step 3.7 will not find a match.

Using a design technique of your choice, design the function that could be used in both steps 3.2 and 3.4 to convert the user's inputs to strings that start with an upper-case character.

(4 marks)

Candidate name_____ Candidate number_____

- 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.

Task 1: software design and development (part B)

A top-level design for the main steps and partial refinements of the sightings program is shown below. Data read from the text file is stored in an array of records in the program.

Program top-level design (pseudocode)

1	Read from text file into sightings array of records	OUT: sightings(town,mammal,date,age)
2	Find and display the age of the oldest walker in the sightings data	IN: sightings(town,mammal,date,age)
3	Find and display the dates of sightings of a chosen mammal in a particular town	IN: sightings(town,mammal,date,age)
4	Count and display the number of sightings for each date in the text file	IN: sightings(town,mammal,date,age)

Refinements

```
3.1   Ask user to enter town
3.2   Call a function to return a string input that starts with an upper-case character
3.3   Ask user to enter mammal
3.4   Call a function to return a string input that starts with an upper-case character
3.5   Display "The dates of sightings were:"
3.6   Start loop for each sighting in array of records
3.7       If sighting matches entered town and mammal then
3.8           Display date
3.9       End if
3.10  End loop
```

Refinement of function for steps 3.2 and 3.4

```
Set firstChar to ASCII value of first character in string
If the firstChar is between 97 and 122 then
    Set firstChar to firstChar −32
    Set string to concatenation of the new first character and the remaining string
End if
Return the string
```

```
4.1   Set dayToCount to first date in sightings array
4.2   Set count to 1
4.3   Start loop from second record to end of sightings array
4.4       If date in current record is the same as dayToCount then
4.5           Add 1 to count
4.6       Else
4.7           Display dayToCount and count
4.8           Set dayToCount to date in current record
4.9           Set count to 1
4.10      End if
4.11  End loop
4.12  Display dayToCount and count
```


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

- use a procedure to:
  - read data from the file to an array of records
  - find and display the age of oldest walker
  - find and display dates of sightings
  - count and display sightings for each date in the file
- use a function to validate upper-case characters
- be maintainable and modular
- follow the design and the refinements provided

(15 marks)

Print evidence of:

- your completed program code
- your output showing the number of sightings for each date in the text file.

Include your name and candidate number on all evidence.

1d Step 4 of the main algorithm counts the number of sightings for each date in the file. There are six sightings on 1 September 2021.

Describe how a watchpoint could be used to test that these sightings are counted correctly.

(2 marks)

1e With reference to your own program code, evaluate:

(2 marks)

♦ the efficiency of the function that changes the first character of the user's input to upper-case
♦ the maintainability of your program, referring to modularity

Candidate name_____ Candidate number_____

Task 2: database design and development (part A)

West Fife Walkers organise walks around the woodlands of West Fife. It wants to keep a record of their walkers, the different routes available and who has taken part on each walk. Each route has been designed by a route planner.

West Fife Walkers wants to create a database to store:

- details of all walkers including their name, address, and contact number
- details of route planners including their name, contact number and year of registration
- details of each route including a general description, its level of difficulty, suitable footwear, the name of the woodland and the distance of the route
- details of walks, including the route walked, which walkers took part and the date the walk took place

Functional requirements for the database:

- display suitable footwear for a chosen route
- display a list of walkers who prefer to walk a route with a chosen level of difficulty
- display a list of walkers who have walked the longest route
- find the travelling distance from a walker's home to the starting point of a chosen route
- count the number of route planners registered in a given year
- display the total number of walkers who have walked a route designed by each route planner

2a The following tables have sample data that shows:

- walkers and the walks they have completed
- each time a route has been walked
- the planner who designed each route

Walker	Walk
Walker1	Walk1
Walker2	Walk2
Walker1	Walk3
Walker2	Walk4
Walker3	Walk5
Walker4	Walk6

Walk	Route
Walk1	Route1
Walk2	Route1
Walk3	Route2
Walk4	Route3
Walk5	Route4
Walk6	Route4

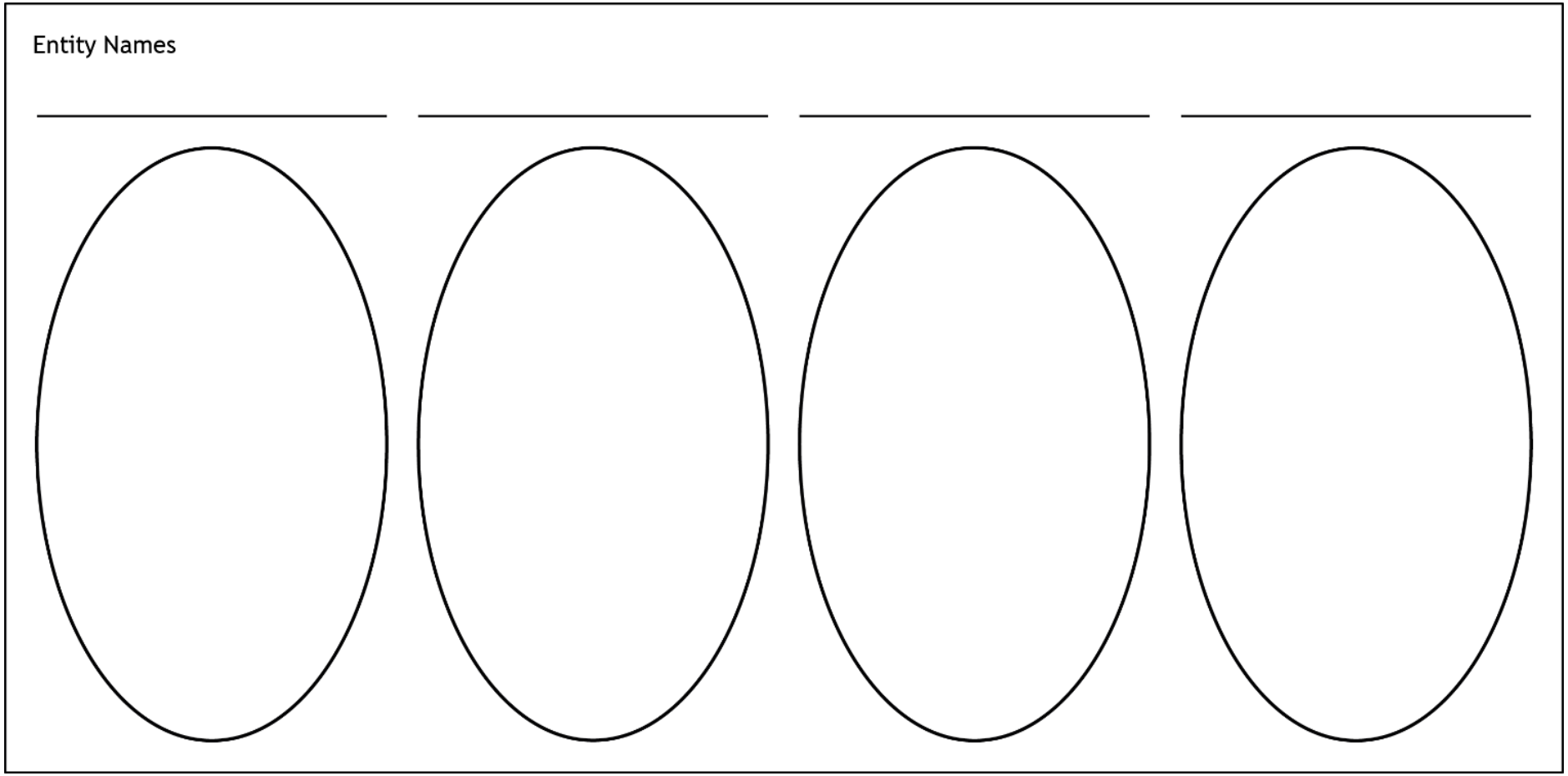
Route	Planner
Route1	Planner3
Route2	Planner2
Route3	Planner1
Route4	Planner3

Using the information from the sample data, complete the blank entity-occurrence diagram on the following page by:

- naming the entities
- completing the sample instances provided for each entity
- showing the association between those instances

(3 marks)

Entity-occurrence diagram

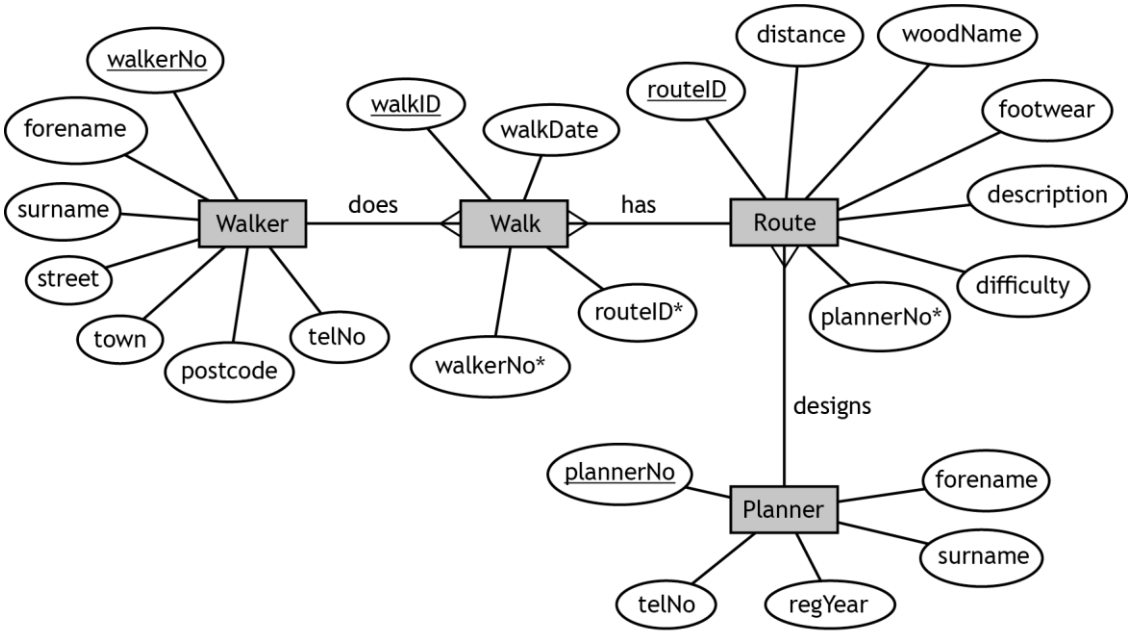


- Check your answers carefully, as you cannot return part A after you hand it in.
- When you are ready, hand part A to your teacher or lecturer and collect part B.

Candidate name_____ Candidate number_____

Task 2: database design and development (part B)

The entity-occurrence diagram is used to draw the entity-relationship diagram for the Walkers database shown below.



The design is then implemented.

Your teacher or lecturer will provide you with a completed and populated database file.

2b West Fife Walkers wants the most successful planner to design a new route. The most successful planner is the person whose route(s) have been walked more than other routes.

A query is required to display details of the four planners, along with the total number of times their route(s) have been walked. The most successful planner should be displayed at the top.

Implement the SQL statement to produce the following output.

(4 marks)

forename	surname	plannerNo	Total participants
Heidi	Benton	101	273
Selena	Booker	103	260
Daniel	Little	104	169
Takeshi	Chen	102	103

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

Include your name and candidate number on all evidence.

2c The most successful planner begins to design the new route. As this route will be longer than the current longest route, West Fife Walkers would like to produce a list of all walkers who have walked the current longest route to ask them questions.

Implement the SQL statement(s) required to produce the list. The expected output is partially shown below.

(5 marks)

walkerNo	forename	surname	telNo
162000	Nancy	Burch	07820 622714
165692	Lola	Kent	07533 447224
167407	Lelia	Mercado	07740 567706
167549	Jayne	Mcneil	07758 443003
169193	Ofelia	Nash	07796 861247
191025	Trina	Hinton	07677 367751
192174	Lorena	Boyle	07142 881757
…	…	…	…

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

Include your name and candidate number on all evidence.

2d The footwear field in the Route table contains suitable footwear for the routes:

- Trail shoes
- Walking boots
- Waterproof shoes
- Boots (high ankle)
- Boots (robust, waterproof)
- Walking shoes

A query is designed to find all the routes that are suitable for any type of shoe.

The SQL statement shown below was implemented and tested.

```
SELECT Route.routeID, woodName, description
FROM Route
WHERE footwear = "Trail shoes"
OR footwear = "Waterproof shoes"
OR footwear = "Walking shoes";
```


The actual output from this SQL statement matches the expected output and is shown below.

routeID	woodName	Description
Dea002	Dean Wood	The forestry road is an easy stroll with open views of the surrounding area and its wildlife. The other half of the walk takes you through woodland down into the valley.
Dev002	Devilla	This walk completes the circuit round the end of the squirrel walk passing seats and nice picnic areas. On the way back a connecting path is taken to join one of the forestry roads.
Bal001	Balgownie	The Balgownie side of the walk is a mixture of open forestry road and woodland paths. The eastern edge has a well-established path through a community wood.

West Fife Walkers will add new routes to the database in the future. These new routes may include additional types of shoes.

Re-write the query so that it will always produce the expected output even if additional types of shoes are added.

Test that your amended query still produces the above output.

(2 marks)

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

Include your name and candidate number on all evidence.

2e Initial analysis identified the following functional requirements:

- display suitable footwear for a chosen route
- display walkers who prefer to walk a route with a chosen level of difficulty
- display a list of walkers who have walked the longest route
- find the travelling distance from a walker's home to the starting point of a chosen route
- count number of route planners registered in a given year
- display the total number of walkers who have walked a route designed by each route planner

State the functional requirement that cannot be implemented using the West Fife Walker's database.

(1 mark)

Candidate name_____ Candidate number_____

Task 3: web design and development (part A)

West Fife Walkers is a group of friends who like to walk in Scotland. The group has decided to offer walking tours of the woodlands close to their homes. West Fife Walkers starts a discussion on social media to identify the preferences of the local communities for tours.

Hi everyone,
Outdoor walking is a great exercise but often people don't know where to begin. Our walking group would like to offer our local communities tours of a few of our nearby woods.
If you are interested in joining us, please get in touch to tell us what information you would like about the woodland walks. WF

DS This sounds great. I walk my dog in the wood behind my house but would like to find out about other places.

KH Brilliant!! 
Can you provide maps as well so we could follow the tour trails on our own. I'd happily pay for a map.

AN I'd like to learn more about local wildlife.

CC I've seen walking sites with maps and detailed directions. If you could provide that it would be great.

RS I've seen some great walking websites. 

GT I'm not much of a walker. Short walks please.

JK Everything is online these days. Would I be able to book a tour online?

SW I'd like the walks to last more than an hour. Got to get my steps up. 

MP Like some others I like short walks. It would be good to know exactly how long and how challenging each walk is. You could maybe rate them.

Following the above feedback, West Fife Walkers decides to go ahead with organising walking tours.

A local web developer has offered to create a website that provides information about the tours and the walks.

3a The web developer wants to draw initial wireframe designs for the website.

Using the questions and answers posted in the social media post, create two end-user requirements and one functional requirement that could be given to the web developer.

(3 marks)

End-user requirement
End-user requirement
Functional requirement

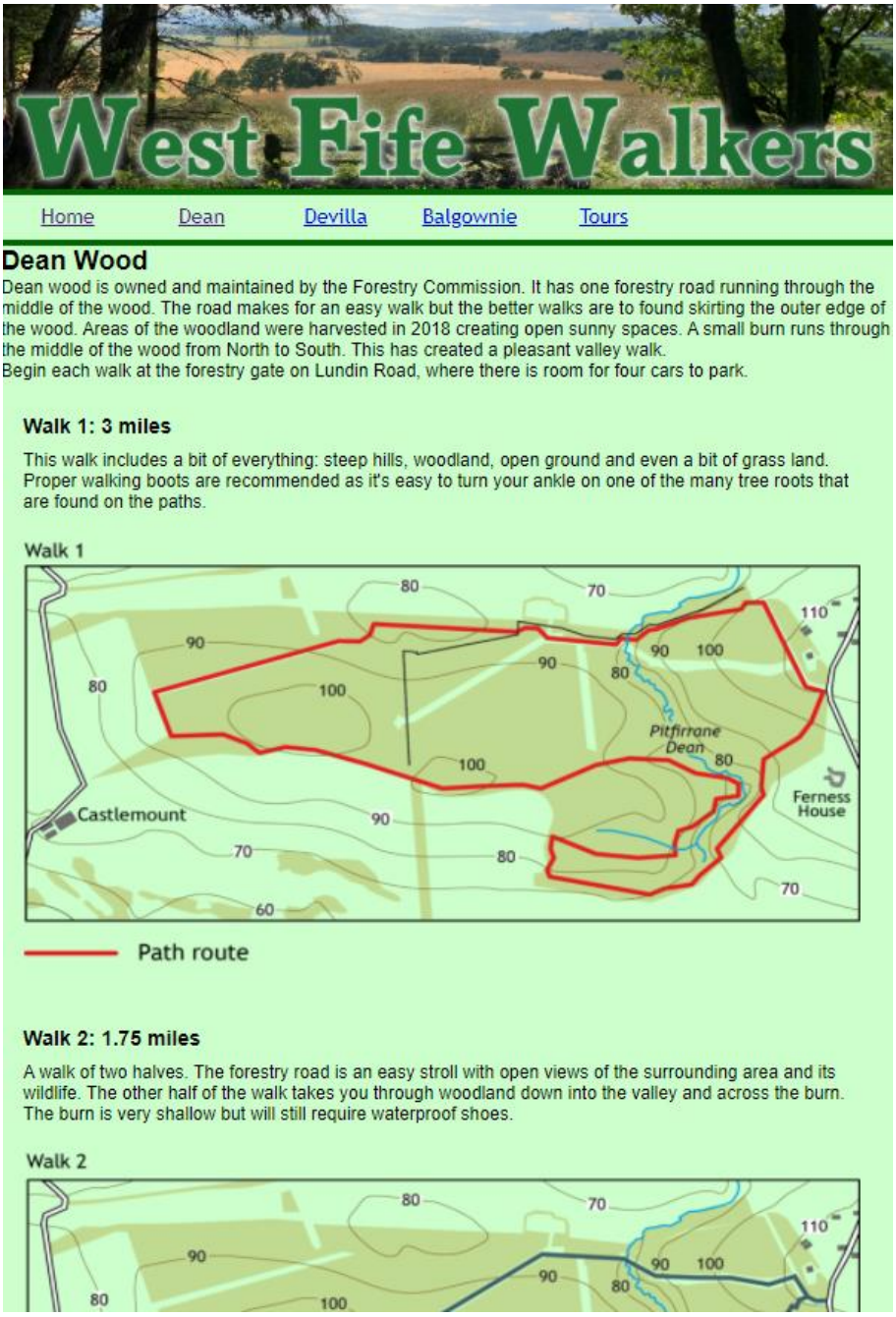
Candidate name_____ Candidate number_____

Your teacher or lecturer will provide you with the complete website for West Fife Walkers.

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

3b The web developer has asked West Fife Walkers for feedback on the website.

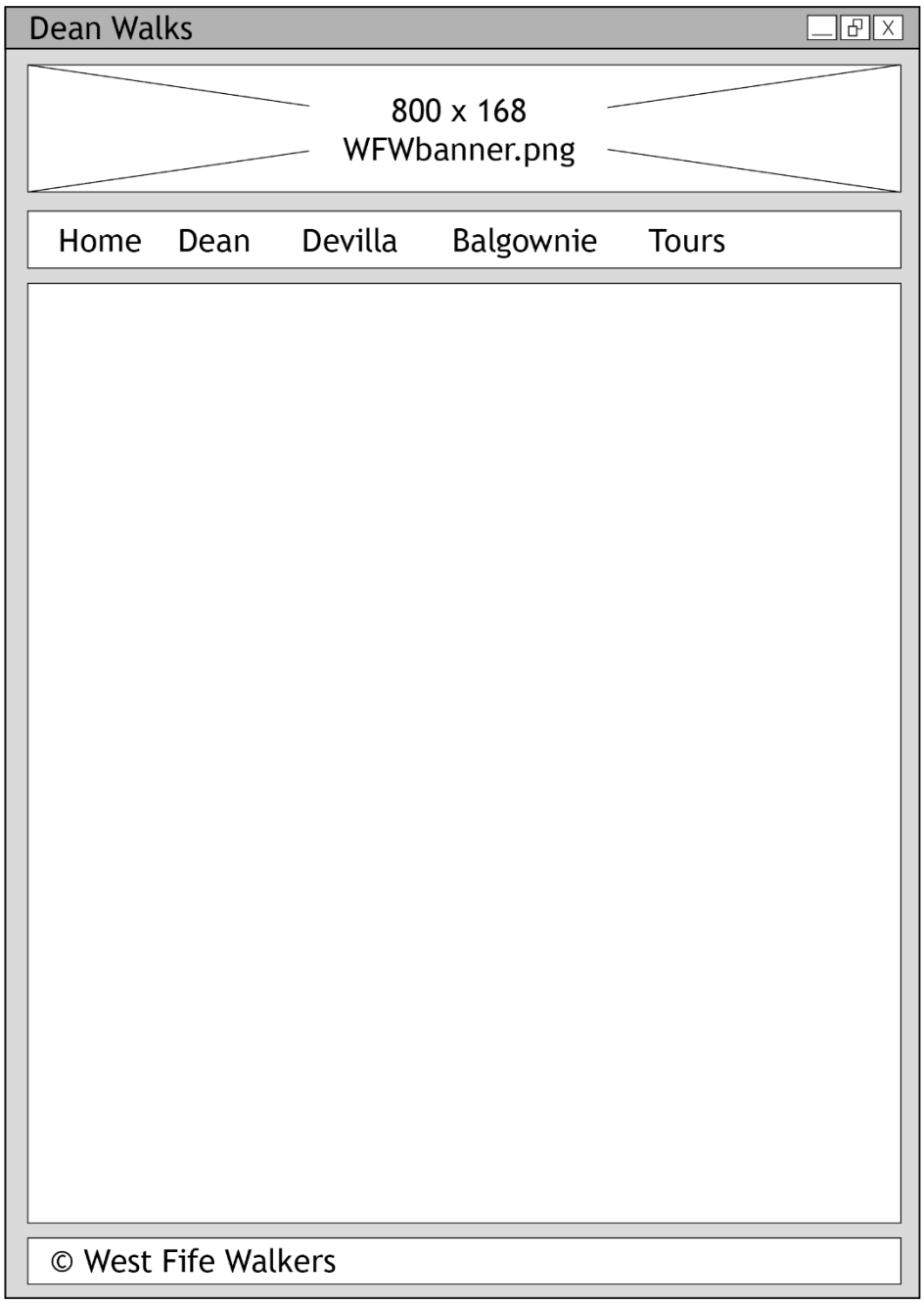
One member of the group comments that the 'dean.html' page is too long stating "to see walks 2 and 3, I had to scroll down quite far". Others agree that this is a problem but cannot think how it could be improved, as the maps must remain large enough to read.



Complete the wireframe below to redesign the 'dean.html' page, so that the user can view the information and map for each walk, one walk at a time on the same web page.

Annotate your design and describe any actions.

(2 marks)

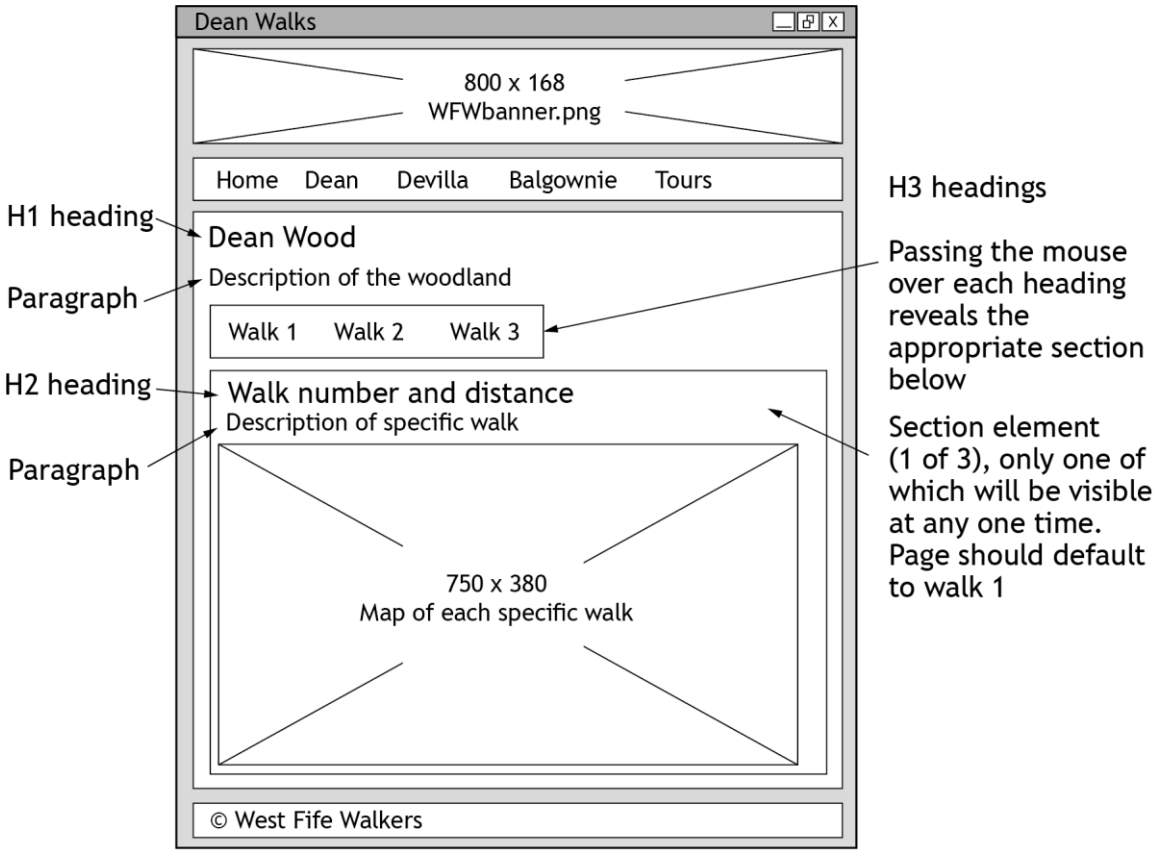


- Check your design carefully. You cannot return to part A after you hand it in.
- When you are ready, hand part A to your teacher or lecturer and collect part B.

Candidate name_____ Candidate number_____

Task 3: web design and development (part B)

3c Using the design below, edit the 'dean.html' file and 'styles.css' file.



Test your completed page, ensuring the headings, paragraphs and images for all three walks can now be viewed on one page or with minimal scrolling.

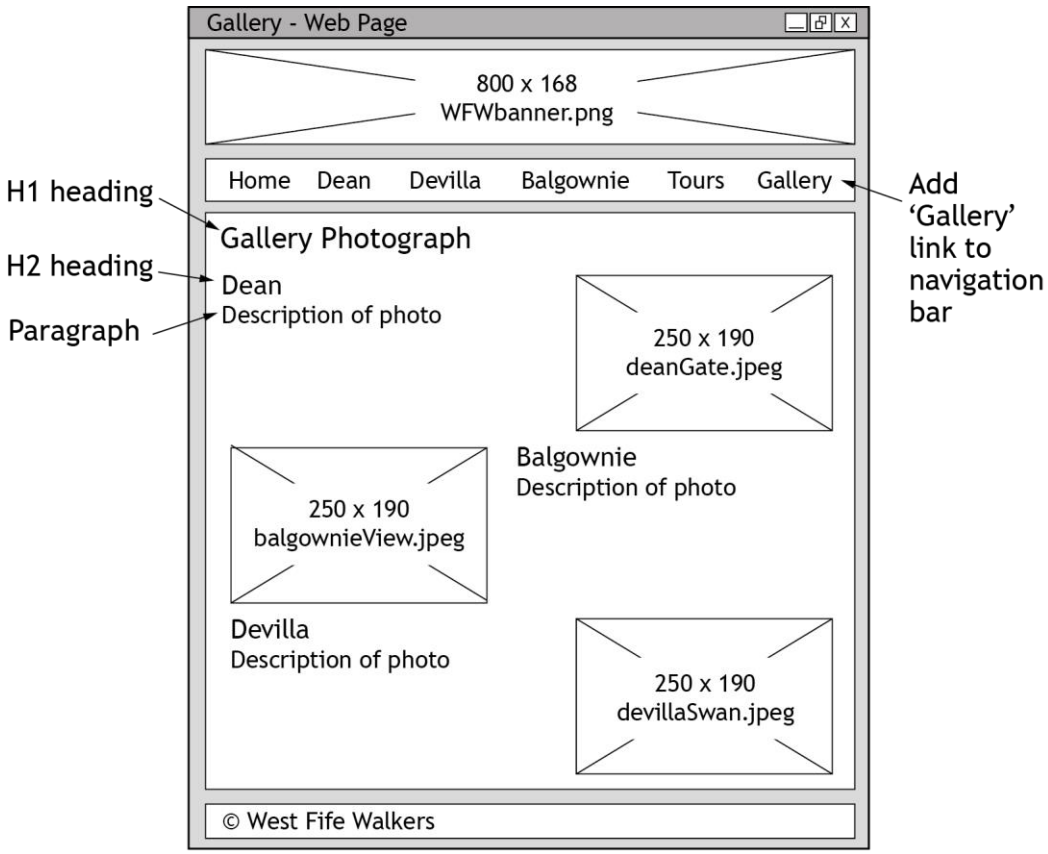
Add your name and candidate number to the `<footer>` element of each page you edit.

(4 marks)

3d West Fife Walkers would also like to add a 'Gallery' page to the website showing photographs of each wood, along with descriptions of each photograph.

Create a new page for the website and implement the wireframe design shown below.

(3 marks)



Add the following description of photographs to the Gallery page:

- Dean — "Looking west out of Dean Wood across the nearby fields."
- Balgownie — "Take a rest at this hilltop bench with lovely open views."
- Devilla — "Devilla's three bodies of water are populated with a wide variety of wildlife."

Print evidence of the:

- edited 'styles.css' file
- edited 'dean.html' file
- 'Dean' page as viewed in a browser
- new 'gallery.html' file
- 'Gallery' page as viewed in a browser

3e(i) The ‘Tours’ page contains a form that walkers can complete to book a place on a tour.

Walkers are instructed to complete the form once, after which West Fife Walkers will organise all the walks for the summer. Describe how the input for the ‘wood selection’ and ‘number of friends’ form elements could be comprehensively tested.

(2 marks)

Candidate name_____ Candidate number_____

3e(ii) State one reason why the form is not fit for purpose.

(1 mark)

Candidate name_____ Candidate number_____

Copyright acknowledgements

Administrative information

Published: January 2022 (version 1.0)

History of changes

Version	Description of change	Date

Security and confidentiality

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

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

