Advanced Higher

Integrative Test and Evaluation (Project)

Integrative testing is the second level of testing used in any development and takes place after component testing has been carried out. Component testing takes place during the development of the solution, when individual functions or modules are created.

Integrative testing is needed for projects that require integration of separate components. Testing is carried out to verify interaction between components and to detect interface defects. These tests determine whether independently developed units of software work correctly when connected to each other.

Due to the nature of integrative testing, some of the test cases needed require temporary code. This code can generate results not explicitly mentioned in the requirements specification. Once these tests are proved successful, any temporary code is removed.

In the examples that follow, test cases that require temporary code are marked with an asterisk*****.

Example 1: a project that combines SDD and DDD

A program is being developed to act as a personal diary app. The program will:

- allow the user to add new diary entries with a date, title, and description
- search diary entries by date
- list diary entries with the most recent entry first
- delete any diary entries that have expired
- allow a maximum of three images to be associated with each diary entry
- store details of all diary entries in a secure database server

This development meets the requirements of the Advanced Higher project, because:

- it is based on the SDD content of the course:
  - details of diary entries are stored and processed in an array of records
  - a sort algorithm is used to display diary entries with the most recent entry first
- it integrates with the DDD content of the course:
  - details of diary entries are stored in a database table
  - a connection with the database server is used to execute SQL queries
  - SQL queries are used to add and delete diary entries
  - an SQL query is used to search for diary entries using date entered by the user
  - an SQL query is used to select all diary entries, to enable processing to take place

On-going testing is used throughout the development to test each component as it is created. Integrative testing is needed, as the development integrates SDD and DDD content. The following three examples describe integrative tests for this development.

Test case ID	Test case objective	Test case description	Expected result
1	Check communication from the program to the secure database server*	Use program code to connect to the secure database server	Message is displayed confirming a successful connection with the database server
2	Check that all diary entries are selected from the database and stored in the array of records*	Use program code to execute the SQL query, store the query results in the array of records, and then display the contents	Details of all diary entries in the database are stored in the array of records and are displayed successfully
3	Check the details of a new diary entry have been added to the database	Enter details of a new diary entry, use program code to execute the SQL query to store the details in a new record of the database table	A new record has been added to the database table to store the new diary entry (check contents of the database table)

Example 2: a project that combines SDD and WDD

An object-oriented program is being developed to act as a recipe manager. The program will:

- use a recipe class to define the data types and methods associated with a recipe
- store recipe details in an array of objects for processing
- allow the user to add new recipes
- save recipe details to a sequential file
- allow the user to search for recipes by ingredient or category (starter, main course or dessert)
- display recipe details in alphabetical order of recipe title

This development meets the requirements of the Advanced Higher project, because:

- it is based on the SDD content of the course:
  - a recipe class is defined
  - an array of objects is used to store and process recipe details
  - the linear search algorithm is used to search the recipe details
  - a sort algorithm is used to arrange the search results, in alphabetical order of recipe title
- it integrates with the WDD content of the course. A web page is used to:
  - present the user with output
  - allow the user to enter details of a new recipe and indicate search criteria
  - display the search resultsOn-going testing is used throughout the development to test each component as it is created. Integrative testing is needed, as the development integrates SDD and WDD content. The following three examples describe integrative tests for this development.

Test case ID	Test case objective	Test case description	Expected result
1	Check communication between the program code and the web page*	Use a HTML textbox to enter the recipe title, then use Java code to display the title entered	Message is displayed on the web page showing the recipe title entered
2	Check that all recipe details are stored in the array of objects*	Use Java code to import recipe details, store in the array of objects, and then display the array contents	Details of all recipes in the array of objects are displayed correctly on the web page
3	Check the recipes are displayed in alphabetical order of title	Use Java code to sort contents of the array of objects in alphabetical order of recipe title, and display the results	Recipe details are displayed on the web page in alphabetical order of recipe title

Example 3: a project that combines DDD and SDD

A movie review database is being developed. The database will:

- store details of movies, actors, reviews, and reviewers in five linked tables of a relational database
- allow users to search for details of individual movies
- allow users to add details of new movies to the database
- allow users to add a review and rating for any movie
- use forms to create an interface for all SQL functionality
- use subqueries to display details of the highest rated movie(s) and details of any movie that has at least five reviews
- use a query to display details of any movie that was not made in the UK
-

This development meets the requirements of the Advanced Higher project, because:

- it is based on the DDD content of the course:
  - details of movies, actors, reviews, and reviewers are stored in five linked tables of a relational database
  - subqueries are used to extract details from the database
  - queries make use of logical operators to search for required details
  - queries and subqueries make use of at least three database tables

- it integrates with the SDD content of the course:
  - forms created using toolbox controls provided within the integrated development environment provide an interface for all SQL functionality
  - the program forms a connection with the secure database server to execute SQL queries
  - the program is used to format and display query results

On-going testing is used throughout the development to test each component as it is created. Integrative testing is needed, as the development integrates DDD and SDD content. The following three examples describe integrative tests for this development.

Test case ID	Test case objective	Test case description	Expected result
1	Check communication between the program form used to enter details of a new review and the secure database server*****	Use program code to connect to the secure database server	Message is displayed confirming a successful connection with the database server
2	Check that the query selected by the user has executed correctly*****	Use program code to generate the query required, execute the query and display a query confirmation message	Message is displayed to confirm a successful execution of the query
3	Check the details of the highest rated movie(s) have been displayed correctly	Use program code to execute the SQL query required and then display the results	Details of the highest rated movie(s) are formatted and displayed

Example 4: a project that combines DDD and WDD

A music albums database is being developed. The database will:

- store details of albums, artists, and tracks in five linked tables of a relational database
- allow the user to search for details of individual albums, artists, and tracks
- allow details of new albums to be added to the database and stored in the relevant tables
- use web pages to create an interface for all SQL functionality
- use subqueries to display details of the most popular albums, artists, and tracks
- use a subquery to display details of the tracks in any album, that has at least ten tracks

This development meets the requirements of the Advanced Higher project, because:

- it is based on the DDD content of the course:
  - details of albums, artists, and tracks are stored in five linked tables of a relational database
  - subqueries are used to extract required album, artist and track details
  - queries and subqueries use at least three database tables
- it integrates with the WDD content of the course:
  - web pages provide an interface to display results of SQL queries
  - online forms are used to enter query search criteria
  - online forms are used to gather details of new albums
  - PHP code is used to form a connection with the secure database server and to execute SQL queries
  - PHP is used to format and display the results returned by SQL queries

On-going testing is used throughout the development to test each component as it is created. Integrative testing is needed, as the development integrates DDD and WDD content. The following three examples describe integrative tests for this development.

Test case ID	Test case objective	Test case description	Expected result
1	Check communication between the online form used to enter details and the secure database server*	Enter the title of a new album, then use PHP code to connect to the database server	Message is displayed confirming a successful connection with the database server
2	Check that the query selected by user has been formed correctly and has executed successfully*	Search for details of all albums by the band Genesis, use PHP code to generate the SQL query required. Use an 'echo' statement to display the	The 'echo' statement is used to display the correct SQL query and a message is displayed on the web page

Test case ID	Test case objective	Test case description	Expected result
		syntax of the query formed, then execute the query and display the query confirmation message	confirming successful execution of the query
3	Check the details of the most popular artist are displayed correctly	Use PHP code to generate the SQL query required, to execute the query and display the results	Details of the most popular artist are formatted and displayed on the web page

Example 5: a project that combines WDD and SDD

A website is being developed to allow the user to play a game of 'Connect Counters'.

This is a 2-player game played on a 5 x 5 grid. Users take it in turns to either position a coloured counter in the grid to form a continuous sequence of counters (horizontally, vertically or diagonally), or block their opponent's sequence. Once the grid is full, the player who has the longest sequence of counters gets a point (in the event of a draw, both players receive points).

The software will:

- allow each player's name to be entered at the start of the game, together with the number of rounds being played (the maximum number of rounds is three)
- allow players to take it in turn to indicate the position of their coloured counter in the grid
- control the game play and award points
- display the name of the winner(s) and points awarded at the end of each round
- display the name of the overall winner, once all rounds of the game have been played

This development meets the requirements of the Advanced Higher project, because:

- it is based on the WDD content of the course:
  - an online form is used to gather and submit details at the start of the game
  - players use an online form of submit buttons to indicate the grid position they want to use on the 'Game Play' page
  - PHP is used to assign variables and process the form data
  - session variables are used to store details entered by the players for the duration of the game
  - external CSS is used to format the layout of all web pages in the website
  - a media query is used to create multiple layouts
- it integrates with the SDD content of the course:
  - a 2-D array is used to represent the position of the players' counters

On-going testing is used throughout the development to test each component as it is created. Integrative testing is needed, as the development integrates WDD and SDD content. The following three examples describe integrative tests for this development.

Test case ID	Test case objective	Test case description	Expected result
1	Check that the number of rounds entered by the user is passed to the game code successfully*	Enter the number of rounds = 2, assign to a PHP session variable and use this to control a fixed loop displaying the round number being played	Messages 'Round 1 being played' 'Round 2 being played' are displayed successfully on the 'Game Play' page of the website
2	Check that the grid position selected by player 1 is updated correctly in the 2-D array*	Once the game starts, the first grid position selected by player 1 is (2,4), a value of 1 should be assigned to position (1,3) of the 2-D array and the full contents of the array displayed	Contents of 2-D array are displayed correctly on the 'Game Play' page of the website 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3	Check that players' names entered at the start are displayed correctly at the end of each round	Enter players' names and number of rounds = 2 at the start of the game, play the game for two rounds, and player details displayed at the end of each round	At the end of rounds 1 and 2, a message is displayed showing the correct player names and both scores on the 'Game Play' page of the website

Example 6: a project that combines WDD and DDD

A photo gallery website is being developed. The website will:

- allow all users to view thumbnails of all publicly available images
- allow new users to create an account for the website
- allow registered users to login and view thumbnails of images that they have stored and have marked as 'private'
- allow all users to click a thumbnail image and display a full-sized image
- allow registered users to add details of new images to the database, indicating whether access to the images is 'public' or 'private'
- use web pages to create an interface for all SQL functionality

This development meets the requirements of the Advanced Higher project, because:

- it is based on the WDD content of the course:
  - users login to the website using an online form
  - PHP is used to assign variables and process the form data
  - session variables are used to store a user's login details while they are logged in to the website
  - external CSS is used to format the layout of all web pages in the website
  - media query is used to create multiple layouts
- it integrates with the DDD content of the course:
  - details of uploaded images are stored in a database table
  - a separate database table is used to store users' login details
  - a connection with the database server is used to execute SQL queries
  - an SQL query is used to check a user's login credentials
  - SQL queries are used to select the images to be displayed

On-going testing is used throughout the development to test each component as it is created. Integrative testing is needed, as the development integrates WDD and DDD content. The following three examples describe integrative tests for this development.

Test case ID	Test case objective	Test case description	Expected result
1	Check communication between the online form used to login to the website and the secure database server*	Login to the website with any username and password, then use PHP code to connect to the database server	Message is displayed confirming successful connection with the database server
2	Check that the user is successfully logged into the website and their	Login to the website with a stored username and password, execute the SQL	Personalised message is displayed on the 'User Gallery' page of the

Test case ID	Test case objective	Test case description	Expected result
	login details have been passed to a new page*	query to check the login details, assign details to the session variables. Display a personalised confirmation message on a separate page	website, confirming successful login to the website
3	Check that only thumbnails of images marked 'public' are displayed on the 'Public Gallery' page of the website	Click the link to load the 'Public Gallery' page, execute the SQL query to identify the images marked 'public', then only display thumbnails of these images	Only thumbnails of the images marked 'public' are displayed on the 'Public Gallery' page of the website. (compare with the details stored in the Photos database table)

Fitness for purpose (SDD, DDD and WDD)

Functional requirements and fitness for purpose

During the analysis stage of the development cycle, candidates identify the functional requirements when creating a requirements specification. The functional requirements are the inputs, processes, and outputs that **must** be included in the design and implementation of any solution to a problem.

A solution is fit for purpose if (following design, implementation and testing) it meets **all** the functional requirements. In the evaluation stage of the Advanced Higher project, candidates discuss if their solution is fit for purpose.

The following examples use functional requirements identified in appendix 1. Both examples assume that a program, website, and database are designed, implemented, and tested.

Example of an evaluation of a solution that is fit for purpose (SDD)

Functional requirements (from appendix 1)

The functional requirements are defined in terms of the inputs, processes, and outputs listed below.

All inputs are imported from a sequential file and all outputs are displayed on the screen. The program is activated by double clicking on the file icon and then selecting 'Run' from the menu. Each process should be a separate procedure or function that is 'called' from the main program.

Inputs:

- itemID
- price
- number in stock

Processes:

- read in data from external file to a 2-D array
- sort the data in order of itemID, from low to high
- search a 2-D array, based on end-user input, for the required itemID

Output:

- If a match is found, the data (itemID, price, number in stock) corresponds to the end-user input.
- If no match is found, a suitable message informs the end-user.

Fitness for purpose

Following comprehensive testing, the program is fit for purpose.

The solution:

- reads data from an external stock file, splits the data and allocates it to a 2-D array
- sorts the data in numerical order using the itemID
- allows the user to display a stock item by selecting an itemID
- searches the data in the 2-D array for the itemID selected and returns the result
- displays formatted output, showing the itemID, price, and number in stock for the selected items
- displays a message 'sorry your item has not been found' if the stock item is not found in the 2-D array

Example of an evaluation of a solution that is NOT fit for purpose (DDD)

Functional requirements (from appendix 1)

The functional requirements are defined in terms of the inputs, processes, and outputs detailed below.

Inputs (customer):

- register: user email, password, password re-entered, firstName, lastName, address, postcode, email
- search details: category
- search details: itemName
- sort details: Field (price or rating) and order required (ascending or descending)

Input (administrator):

- edit item details: itemID, price
- edit customer details: customerID, address, postcode, email
- add item details: itemID, itemName, description, category, price
- delete item details: itemID
- delete customer details: customerID
- monthly orders: month

Processes:

- auto-generate the customerID when a new customer registers
- queries to insert records into the Customer and Item tables
- queries to sort the item details in order of price and rating
- queries to delete specific customer and item records from the database
- queries to edit records in the Customer and Item tables
- queries to search Item table

- queries to display details of all orders placed in a particular monthOutput:
- confirmation of successful insertions
- confirmation of successful deletions
- confirmation of successful edits
- answer tables showing details of sorted items (sorts)
- answer tables showing details of required items (searches)

Fitness for purpose

Following comprehensive testing, the database-driven website and its user interface are not fit for purpose.

Although the solution successfully implements all insertions, deletions, and edits of the back-end database table data, it does not:

- provide confirmation of these actions
- allow the customer to sort the results of a stock item search

The solution does successfully:

- store the required information for each new customer (including an auto-generated customerID) when they register
- allow a customer's address, postcode, and email to be edited and deleted by an administrator
- store the required information for each new stock item
- allow stock item details to be edited and deleted by an administrator
- display stock items (descriptions, categories and prices) following a customer search for stock items by either name or category
- display all the orders for a month selected by an administrator