T Level Technical Qualification in Digital Production, Design and Development

Mark Scheme (Results)

June 2022

Employer Set Project

General Marking Guidance

- All learners must receive the same treatment. Examiners must mark the first learner in exactly the same way as they mark the last.
- Mark schemes should be applied positively. Learners must be rewarded for what they have shown they can do rather than penalised for omissions.
- Examiners should mark according to the mark scheme not according to their perception of where the grade boundaries may lie.
- All marks on the mark scheme should be used appropriately.
- All the marks on the mark scheme are designed to be awarded. Examiners should always award full marks if deserved. Examiners should also be prepared to award zero marks if the learner's response is not rewardable according to the mark scheme.
- Where judgement is required, a mark scheme will provide the principles by which marks will be awarded.
- When examiners are in doubt regarding the application of the mark scheme to a learner's response, a senior examiner should be consulted.
- Crossed out work should be marked **unless** the learner has replaced it with an alternative response.
- Accept incorrect/phonetic spelling (as long as the term is recognisable) unless instructed otherwise.

Levels-Based Mark Scheme Guidance

Levels-based mark schemes (LBMS) have been designed to assess students' work holistically. They consist of two parts:

1) Indicative content
   Indicative content reflects content-related points that a student might make but is not an exhaustive list. Nor is it a model answer. Students may make some or none of the points included in the indicative content as its purpose is as a guide for the relevance and expectation of the responses. Students must be credited for any appropriate response.

2) Levels-based descriptors
   Each level is made up of a number of traits which when combined together articulate the quality of response that a student needs to demonstrate. The traits progress across the levels to demonstrate the different expectations of each level. When using a l6evels-based mark scheme, the 'best fit' approach should be used.

Applying the levels-based descriptors

Examiners should take a 'best fit' approach to determining the mark.

- Examiners should first make a holistic judgement on which level most closely matches the student's response. Students will be placed in the level that best describes their answer. Answers can display characteristics from more than one level, and where this happens markers must use any additional guidance (e.g. weighting of traits) and their professional judgement to decide which level is most appropriate.

- The mark awarded within the level will be decided based on the quality of the answer and will be modified according to how securely all traits are displayed at that level:

  - Marks will be awarded at the top of that level if the student has evidenced each of the descriptor traits securely.
  - Where the response does not securely meet all traits, the marks should be awarded based on how closely the descriptor has been met.

Task 1- Planning a project

Indicative content and marker guidance
NOTES: COSTS – DON'T TAKE VALUES AS GIVEN – LOOK AT FORECASTS AND PROJECT COSTS RATIONALE – SHOULD DISCUSS THEIR CHOSEN PLANNING METHODS – CAN BE IN DEPTH BUT SHOULD REFELCT THEIR IMPLEMENTATION BE WARY OF WHICH SOFTWARE APPICATION YOU USE TO CHECK PROJECT FILES Gannt Chart: • Expect to see tasks broken down into smaller chunks where sensible for example: o May show use of an Agile approach (or similar) o Large modules (e.g. backend database, data analysis etc.) broken down into multiple sections of development and unit testing with logical resources being applied to tasks – look out for learners applying the same developer to test the modules – can be ok if justified through testing experience o When splitting tasks learners should show an understanding of how total development hours should be split between the team working on it. o A task related to of the choice of software licensing should be seen here – single fee with technical support or annual fee without. • There should be sensible use of concurrent and serial tasks for example: o Logical task sequences e.g. server setup/installed before module deployed. o Showing that as one unit is being tested, development could be taking place with other team members o Integration testing would be expected after specific modules have been unit tested o At the higher end, expect to see consideration around the testing time for each module and how this could be split up to allow testing along with other modules. o Possible to see an agile approach on a granular level e.g. per module as well as the project as a whole o Should show some awareness of the requirements of tasks having predecessor requirements • There is no single correct way organize the plan but task orders should be sensible for example 'create a test plan' should occur BEFORE testing commences, staff training would occur much later in the process when the system is nearer completion etc. • The order and implementation of the project may vary significantly depending on the SDLC approach that they are taking (check against learner rationale). for example : o a RAD/agile that looks at a Minimum value product (MVP) they may 'Deploy' some of the modules very early on, after a short portion of development time, and then test and develop further, deploying more modules as they go. o Or they may decide that their approach is to have most modules mostly developed and then deployed and tested later. The rationale will show the reasoning for the chosen project development approach demonstrated in the Gantt chart. Points the learners may consider, although some of these will vary depending on the choices learners make in terms of organisation. These include but are not limited to: • Staff (skills, industry expertise, experience) o The senior software engineer has worked on large scale projects and has experience of deployment, as well as specific experience of developing data analysis modules. o The junior software engineer (TB) has less experience but has some clear experience around staff development and UI development (though this is in a different industry), similarly, there is experience of working on a backend database here, but this is limited, which could mean carrying out work supervised. o The junior software engineer (NA) has limited applicable experience but is a cheaper option – if made use of, could see work being checked or overseen at intervals in the project by a more senior engineer. o The hardware and networking technician has the clear expertise in the field of overseeing large scale infrastructure upgrades so expect to see her in these tasks – if assigned elsewhere expect to see clear justifications in rationale – could be seen around the testing elements o The database engineer has lots of testing experience – as well as working on the database module, expect to see this team member testing across the range of modules. There is an option to use as part of staff training but would need ot justify over the cheaper option (TB). o Testing should be allocated to members of staff in a sensible manner – in some cases, in learner work, testers are being allocated as they are the developer – in some cases this can make sense – if clearly justified in the rationale • Resources o Learners should rationalize the choice of allocation of tasks to different members of staff (see above) o A justification of which licensing cost they opt for o Though costings vary – should show some logical methods used to calculate.

- Time scales and costs
  - Learners should identify if they made the deadline or not. The number of hours for the project and available time is very close – with consideration of predecessor tasks. The timeline is reasonable, expect some learners to come in under by suitably dividing tasks amongst appropriate staff.
  - More contingency time may be needed on jobs performed by the Junior staff members/senior members assigned to oversee work
  - Learners should discuss the cost of their solution with reference to how these costs were arrived at, and if the projected costs and increase in revenue make the project feasible – showing awareness that the project benefits will be seen over a number of years
  - Learners could look to justify using less experienced members of staff in order to keep costings down, though mitigation (e.g. overview by the senior, more time to allow for development) and rationale should be considered
- Risks
  - Some staff have key skills but from different industries
  - It is unrealistic to assume that there will be no staff absence in what will be a large-scale project - wise to build contingency time into the project as a whole and/or the tasks assigned to each member of the team – this would be for higher level learners
  - Time scale consideration around getting the project complete on time

- In some cases, learners will write a lot for the rationale, but this may be almost a description of what they did, or a transfer of staff profiles – try to focus on aspects of the learner work that shows depth of consideration of their plans

Assessment focus	Band 0	Band 1	Band 2	Band 3
	0	**1-2**	**3-4**	**5-6**
Gantt chart	No rewardable material	Project tasks are somewhat organised in logical and efficient manner making some use of an appropriate SDLC model to provide some accurate prediction of the project's timescales Resources have sometimes been assigned to project tasks effectively but there are some major and/or significant errors or omissions	Project tasks are organised in a mostly logical and efficient manner making use of an appropriate SDLC model to provide mostly accurate predictions of the project's timescales Resources have mostly been assigned to the project tasks effectively, but there are some minor errors or omissions.	Project tasks are organised in a thoroughly logical and efficient manner using an appropriate SDLC model to provide thoroughly accurate predictions of the project's timescales Resources have consistently been assigned to the project tasks effectively.
	0	**1**	**2-3**	**4**
Resource and cost plan		Some correct resources and accurate costs have been added to the plan resulting in an estimate of limited accuracy.	Mostly correct resources and accurate costs have been added to the plan resulting in an estimate that is largely accurate.	Fully correct resources and accurate costs have been added to the plan resulting in an accurate estimate.
	0	**1-3**	**4-6**	**7-9**
Rationale	No rewardable material	Rationale for project planning decisions demonstrate some effective consideration of: • cost, risks and benefits • order and timing of tasks • selection and allocation of resources • dependencies and prerequisites.	Rationale for project planning decisions demonstrate mostly effective consideration of: • cost, risks and benefits • order and timing of tasks • selection and allocation of resources • dependencies and prerequisites.	Rationale for project planning decisions demonstrate a thorough and perceptive consideration of: • cost, risk and benefits • order and timing of tasks • selection and allocation of resources • dependencies and prerequisites.

Task 2- Identifying and fixing defects in existing code

Indicative content and marker guidance

Line numbe r	Description of error	Possible fix
	houses = [['LONDON','Terraced', 3 735000], Missing , after 3	
3	ourregions = ['LONDON', 'LEEDS', 'CARDIFF', 'BRISTOL'] Array not used	MAY CHOOSE TO REMOVE, MAY CHOOSE TO INCOROPORATE
	Before select = input('Please select a purchase')) if select > 0 and select < len(houses): sel_check = True Extra)	Removal of extra bracket – NOTE HOWEVER THIS IS NEEDED FOR THE FIX FOR AN INT INPUT

14	def unique_regions():
	unique_list = []
	existing_regions = [item[0] for item in houses]
	for x in existing_regions:
	if x in unique_list:
	unique_list.append(x)
	print(unique_list)
	
	Should be not in
		def unique_regions():
		unique_list = []
		existing_regions = [item[0] for item in houses]
		for x in existing_regions:
		if x not in unique_list:
		unique_list.append(x)
		print(unique_list)

	def show_sales():
	if len(sold) > 0:
	print("Forename Surname Property cost Total")
	for i in sales:
	print(i)
	else:
	print('no sales')
	
	Should be checking length of sales not sold
		def show_sales():
		if len(sales) > 0:
		print("Forename Surname Property cost Total")
		for i in sales:
		print(i)
		else:
		print('no sales')

47	Should check as int input	try: select = int(input('Please select a purchase'

```
sel_check = False
while not sel_check:
    try:
        select = input('Please select a purchase'))
        if select > 0 and select < len(houses):
            sel_check = True
```

```
  if sub_total > 100000:
      total_fees += 3000+(sub_total-100000) * 0.2
  else:
      total_fees += sub_total *0.3
```

```
if sub_total > 100000:
    total_fees += 3000 + (sub_total - 100000) * 0.02
else:
    total_fees += sub_total * 0.03

print(houses[select - 1])
```

```
print('Customer Receipt\n\n  FORENAME:{}  SURNAME: {}  PROPERTY COST:  {}  WITH STAMP DUTY:   {}
print('\nTRANSACTION COMPLETE - PROPERTY REMOVED FROM SALES DATABASE\n')
print (houses[select])
del houses[select]
```

```
if menuselection == 1:
    return_stock
if menuselection == 2:
    region_search
if menuselection ==3:
    house_sale
if menuselection==4:
    show_sales
```

```
if menuselection == 1:
    return_stock()
if menuselection == 2:
    region_search()
if menuselection == 3:
    house_sale()
if menuselection == 4:
    add_house()
if menuselection == 5:
    show_sales()
if menuselection == 6:
```

The test plan/log should also contain inclusion of tests that show that areas of the program that appear to be coded correctly have been tested to ensure outputs are correct and the program is robust. These may include (but not limited to):

- Positive and negative values for number of products in stock
- Non numerical values for number each item
- Incorrect inputs e.g. products not in the list, numerical values entered instead of name of product
- Manual calculations of totals to check program is outputting correct values for totals
- Adding new stock to the product array
- Testing the different options available to the user through the menu

A limited understanding of program requirements would be typified by only identifying and fixing the functional errors that would be highlighted by the IDE, but would not identify and fix logical errors. The number of errors identified is not a hurdle between Band 2 and 3, the discriminator is the quality and appropriateness of the tests selected, and the level of understanding shown of the process

Assessment focus	Band 0	Band 1	Band 2	Band 3
	0	1-2	3--5	6-8
Use of testing to identify defects	No rewardable material	Tests selected show a basic understanding of the identified program requirements. Test log includes some appropriate test data. Testing has identified some error in the code provided.	Tests selected show a good understanding of the identified program requirements. Test log includes a good a range of normal, erroneous and extreme data. Testing has identified most errors in the code provided.	Tests selected show a thorough and detailed understanding of the identified program requirements. Test log includes a comprehensive range of normal, erroneous and extreme data. Testing has comprehensively identified the errors in the code provided.
		1	2-3	4
Understanding of the testing process		Test log shows a basic understanding of how errors/problems were identified and how they were rectified.	Test log shows a good understanding of how errors/problems were identified and how they were rectified.	Test log shows a thorough and detailed understanding of how errors/problems were identified and how they were rectified.
		1-3	4-6	7-9
The solution		Code has some functionality, but significant errors still persist. Changes made apply some precise logic and programming structures which would result in some correct outcomes.	Code is mostly functional, but some minor errors still persist. Changes made apply mostly precise logic and programming structures which would result in mostly correct outcomes.	Code is fully functional. Changes to code apply fully precise logic and programming structures throughout which would result in consistently correct outcomes

Task 3- Designing a solution

Indicative content and marker guidance
General guidance: • Algorithm designs should demostrate decomposition of the problem into simpler and more understood primitives. A decomposition diagram is not required, but if provided mark for decomposition, otherwise for consideration of decomposition in design documents e.g. flowchart and pseudocode • The design should provide high level coverage of the process as well as identify reusable components • Detailed algorithms (pseudocode) do not need to be provided for ALL repeated processes. For example if the process for calculating sales for regions is very similar, the learner would not need to provide algorithms for each region, rather they should show how reusable code and may provide some additional annotation to explain the process as necessary. • Good decomposition will show all the necessary processes and sub-processes that make up the main problem these might include: ▪ Importing data from the CSV file ▪ Taking input to select region ▪ Iterating through data for specified data ▪ Taking a date range specified for data ▪ A calculation related to increase in value ▪ Generation of data frames and graphs ▪ Aspects related to validating user input Some general characteristics of a good algorithm that may be demonstrated are: • the steps are clearly defined • each step is uniquely defined and should depend on the input and the result of the preceding steps • receives input, selection of regions and dates, as well as regions – inputs for user interface should also be considered • produces appropriate type of output e.g. screen display, return value or return list, which results are required, what happens if no results can be computed maybe error • Sensible naming conventions for variables and processes • Use of key words, symbols, hierarchies, and structures as appropriate to the chosen method to represent the algorithm (i.e., pseudocode or flowchart) Scenario specific characteristics may include: • Suitable logic for selection of regions and dates to show value increases • Suitable calculations to show increase in values over time • Links to CSV to get data. • Sensible use of CSV or run-rime data structure (e.g data frame) to hold different parts of the data for processing e.g. ▪ List to hold menu options, or predefined date ranges ▪ New data frame to hold data for the selected date range to aid calculations • Understanding of given data such as: ▪ Use of header row in CSV to required data (e.g. region, house type) ▪ Need to convert dates to a usable date format • Simplification for user e.g. choose a number from menu rather than type the region in full **Example detailed algorithms for repeated processes:** Note – these are intended to be indicative of the types of algorithms that may be presented. These **do not** show all processes. Accept any responses that provide logically correct outcomes/solutions

```
Get and validate Region choice
WHILE not_valid_input_flag = TRUE
        SEND ''Please enter the number of the region you wish to view TO DISPLAY

        SET region_list TO [<list of regions>]

        RECEIVE region_choice (integer) FROM KEYBOARD

        IF region_choice NOT integer THEN:
                SEND 'That is not  a valid choice' TO DISPLAY
                SET not_valid_input_flag TO TRUE
        ELSE IF region_choice <1 OR >6 THEN:
                SEND 'That is not  a valid choice' TO DISPLAY
                SET not_valid_input_flag TO TRUE
        ELSE:
                SET region_choice TO region_list [int(region_choice)-1]          RETURN
region_name
```

```
Get and validate date range

    SET not_valid_flag TO TRUE

    while flag == TRUE
       SEND ''Please enter a start date for your time range Jan-20) TO DISPLAY

        RECEIVE input (STRING) FROM KEYBOARD

       try:
         SET input(STRING) TO input(DATE1)
       except:
          SEND "Sorry, you did not enter a valid date" TO DISPLAY
          flag = True
       else:
          SET start_date TO input(DATE1)
          RETURN start_date
```

Assessment focus	Band 0	Band 1	Band 2	Band 3
	0	1-2	3-5	6-8
Decomposition of the problem	No rewardable material	Basic or superficial decomposition of the identified problems that superficially cover the required: • inputs • processes11 • outputs.	Mostly detailed decomposition of the identified problems that sufficiently cover the required: • inputs • processes • outputs.	Thorough and detailed decomposition of the identified problems that comprehensively cover the required: • inputs • processes • outputs.
		1-2	3-4	5-6
Application of logical thinking and conventions		Algorithms would produce some correct outcomes as a result of: • some precise logic • some appropriate structure and sequence which is likely to be inefficient.	Algorithms would produce mostly correct outcomes as a result of: • mostly precise logic • appropriate structure and sequence but which may lack efficiency.	Algorithms would produce correct outcomes as a result of: • precise logic • Appropriate and efficient structure and sequence.
		Some use of accepted conventions.	Mostly appropriate use of accepted conventions though some minor inconsistencies may still exist.	Appropriate and consistent use of accepted conventions.
		1	2	3
Communication of the design		Superficial communication of the design uses technical language which is only sometimes appropriate for the audience.	Adequate communication of the design uses technical language which is mostly appropriate for the audience.	Effective communication of the design uses technical language which is appropriate for the audience.

Task 4a- Developing the solution

Indicative content and marker guidance

The solution
• Provides a developed coded solution that utilises the given code and adds the additional functionality as stated in the requirements.
• Integration of existing code may include:
  ○ 'import' function to pull code as a whole in when needed (note given code and learner code will need to be in the same folder)
  ○ Integration into learners own code base as a function
• The solution will be well structured and modular in nature – clearly see subsections this may be separated modules or the use of Procedures, Functions or classes
• Code will be annotated to aid future maintenance of the system
• Data/information should be output in a meaningful way to the user. This may include use of a data-frame, graph and/or text based summary
• Output data should show consideration of different sale people over time but his may be interpreted in slightly different ways. Such as:
  ○ Higher level responses will make better use of patterns and trends over time and allow the user to select regions and house types, as well as dates in different combinations to give more meaningful data analysis.
  ○ lower levels responses may only extract based on a single criterion e.g. a single region, making use of a hard coded date range instead of user input
  ○ Learners make use of tabulated formats or line graphs to display this information in a clearer way.
  ○ Lower level responses may extract some meaningful information but may only output in an unsorted fashion or may not give the user sufficient choice
• There are different ways that this task can be interpreted so the characteristics of higher level responses will show a greater discrimination of the data to be extracted.
• Code should be robust, typical errors that may be accounted for in this scenario include negative values, non-numerical characters, entering a choice not provided in the menu, mis-spelling of region names/wrong capitalization
 	
Possible areas included that contribute to 'user experience':
• Outputs are meaningful and make sense to the user e.g. outputs are accompanied by meaningful text to contextualise them
• Simplification of input processes e.g. use of numbers in a menu rather than writing the full region names.
• Creation of new columns in a dataframe to calculate values
• Numerical values are rounded to a sensible number of decimal places and currency symbol added as appropriate
• Use of visualization e.g. bar graphs to show values over time
• Helpful messages and robust input handling
 	
Security consideration relating to the solutions could include:
• Avoiding global variables, passing data back from functions
• Avoiding the use of a single generated dataframe to ensure security of the data – good practice to generate a new data frame within a function
• error handling: If the system crashes, is any data returned in the error message

Example code snippets for parts of the solution:

Note – these are intended to be indicative of the types of re that may be presented. These **do not** show all processes. Accept any responses that provide logically correct outcomes/solutions

Developing user menus to incorporate functionality

```
print("\t\t****Welcome to the Dashboard****")

print('1 Return all current data')
print('2 Return data for a specific region')
print('3 Return data based on property types in a region')
print('4 find region with greatest increase')
```

Showing regional totals

```
def region_validation():
    region = input("Enter region name:")
    return region.capitalize()

def start_date():
    startdate = input("PLEASE ENTER A START DATE AS MONTH-YEAR e.g. JAN-20")
    return startdate.capitalize()

def end_date():
    enddate = input("PLEASE ENTER AN END DATE AS MONTH-YEAR e.g. JAN-20")
    return enddate.capitalize()
```

Generating data based on user inputs

```
for x, y in enumerate(df["Region Code"]):
    if y == region_codes[selected_region]:
        property_indexes.append(x + 2)

for x in property_indexes:
    if df.iloc[x - 2]["Property Type"] == selected_property:
        print(df.iloc[x - 2])
```

Assessment focus	Band 0	Band 1	Band 2	Band 3	Band 4
	0	1-2	3-4	5-6	[redacted]
Functionality	No rewardable material	The solution implements code with some functionality but some major errors still persist.	The solution implements mostly functional code but code may lack efficiency and some minor errors still persist.	The solution implements functional and efficient code throughout.	
		1	2	3	
Logic and programming structures		The code uses some precise logic and programming structures which would result in some correct outcomes.	The code uses mostly precise logic and programming structures which would result in sufficiently correct outcomes.	The code uses precise logic and programming structures throughout which would result in consistently correct outcomes.	
		1	2	3	
Robustness		The code handles some common user errors	The code handles most common user errors	The code thoroughly handles common, and most unexpected, user errors	
		1-2	3-4	5-6	
Security		The code mitigates against some common vulnerabilities as a result of some effective application of secure coding practices.	The code mitigates against most relevant vulnerabilities through mostly effective application of secure coding practices	The code thoroughly mitigates against relevant vulnerabilities through effective application of secure coding practices	
		1-2	3-4	5-6	7-8
Code organisation		The code is partially maintainable by a third party but would present significant difficulties through the use of: • inconsistent naming conventions • limited logical organisation • limited informative commenting	The code is partially maintainable by a third party but would present some minor difficulties through the use of: • some consistent naming conventions • some logical organisation • some informative commenting	The code is maintainable by a third party and would present only a few minor difficulties through the use of: • mostly consistent naming conventions • mostly logical organisation • mostly informative commenting	The code is easily maintainable by a third party through the use of: • consistent and appropriate naming conventions • fully logical organisation • highly informative commenting

	0	1-2	3-4	5-6	7-8
User experience	No rewardable material	Basic user experience is provided through limited effective use of: • input handling • user guidance and error messages • outputs	Adequate user experience is provided through somewhat effective use of: • input handling • user guidance and error messages • outputs	Good user experience is provided through mostly effective use of: • input handling • user guidance and error messages • outputs	Excellent user experience is provided through consistently effective use of: • input handling • user guidance and error messages • outputs

Task 4b- Reflective evaluation

Indicative content and marker guidance
Indicative content will vary according to the approach learners have taken in task 4a and the effectiveness of the solution they created. Generic features of effective evaluations are likely to include: ● the extent to which the solution meets the: ○ requirements of the set task brief ○ needs of the users. ● a justification of how the solution could be further developed/enhanced. ● specific examples from the solution to support points made ● contextualisation of any points made and explaining what they did and justifying why. Contextulisation for this scenario may include: ● Choice of data output(e.g text, table, graph type) – which is most suitable for showing value of properties over time/region with the highest increase in property value ● Rounding vs truncation for calculations and consideration of data types to specify number of decimal places for the calculation output ● How the brief was interpreted – such as how regions were selected by the user, were specific date ranges entered ● How existing code was integrated ● Choice of libraries/functions to get required data ● How most region with the highest increase in property value has been determined e.g. have user inputs for dates been considered, how has the information been grouped/sorted ● How data was extracted (e.g. use directly form the csv file, use of subsets of data, how the choice of data frame or datafile impacted on search/extraction) ● Use of variables (global vs local, passing data between functions) ● Input error handling – e.g. why they might have excluded text or negative values, menu options etc

Assessment focus	Band 0	Band 1	Band 2	Band 3
	0	1-2	3-4	5-6
Programming outcomes	No rewardable material	Judgements reached are somewhat supported showing a superficial or basic understanding of how well the solution met the: ● requirements of the set task brief ● needs of the users.	Judgements reached are mostly well supported showing a good understanding of how well the solution met the: ● requirements of the set task brief ● needs of the users.	Judgements reached are comprehensively well supported showing a thorough and detailed understanding of how well the solution met the: ● requirements of the set task brief ● needs of the users.
		1	2	3
Future Developments		A superficial or simplistic rationale is provided for what future developments should be implemented.	A good rationale which is reasonably well supported is provided for what future developments should be implemented.	A convincing and well-supported rationale is provided for what future developments should be implemented.