Unit:
Introduction to Programming

Assignment title:
Glenesk Activities

SAMPLE

Marking Scheme

Markers are advised that, unless a task specifies that an answer be provided in a particular form, then an answer that is correct (factually or in practical terms) **must** be given the available marks. If there is doubt as to the correctness of an answer, the relevant NCC Education materials should be the first authority.

This marking scheme has been prepared as a **guide only** to markers and there will frequently be many alternative responses which will provide a valid answer.

Each candidate's script must be fully annotated with the marker's comments (where applicable) and the marks allocated for each part of the tasks.

Throughout the marking, please credit any valid alternative point.

Where markers award half marks in any part of a task, they should ensure that the total mark recorded for the task is rounded up to a whole mark.

Marker's comments:

Moderator's comments:

Mark:	**Moderated mark:**	**Final mark:**

Penalties applied for academic malpractice:

Design

	Guide	**Maximum Marks**
1	Generate options lists: • 2 marks: Provide suitable lists of items for all Glenesk activities components with options (date of activities day, lunch options, core activities, optional activities, delivery types) • 2 marks: Provide a list of prices for all optional activities and for delivery charges. These should be the prices given in the scenario and must be an appropriate data type (double or float)	**4**
2	Design the visual aspect of the forms: • Choice of controls o 3 marks: Suitable components, neatly laid out. Use of combo boxes to minimise screen real estate and suitable method for selection of item options to prevent invalid input. o 2 marks: Suitable components, neatly laid out. Components ensure all choices available but may not be the most efficient choices in terms of screen real estate or preventing invalid input. o 1 mark: Some required components missing. Poor GUI layout. o 0 marks: Not attempted or no suitable controls chosen for input choices. • Suitable choice of colours and fonts: o 3 marks: All colours and fonts suitable and consistently applied. o 2 marks: Either suitable, consistently applied colour scheme or fonts, but not both. o 1 mark: Suitable fonts or colours but lacking consistency. o 0 marks: Colours of controls and fonts used make text hard to read. • Naming conventions: o 2 marks: Naming conventions followed for all controls. o 1 mark: Naming conventions followed for some controls. o 0 marks: Naming conventions not followed.	**8**
3	State/define all the data validation criteria: • 2 marks: Use of combo boxes, list boxes or other selection controls which remove the need for user input wherever possible. • 2 marks: Range validation on all required inputs • 2 marks: Data type validation for all typed user input (e.g. for the address and credit card number details) OR use of control properties to prevent input of wrong data type. • 2 marks: Additional validation (over and above data type) where possible e.g. checking credit card number is 16 characters long.	**8**

	Guide	**Maximum Marks**
4	Complete all Object Definition Sheets: • Default Properties with values o 4 marks: All default properties listed and with correct values. o 3 marks: Most default properties listed, and with correct values. o 2 marks: Some properties missing, or with incorrect values. o 1 mark: Few default properties listed, or most have incorrect values. o 0 marks: Not attempted or no properties listed correctly. • Event Procedures o 4 marks: All event procedures listed correctly. o 3 marks: Most event procedures listed correctly. o 2 marks: Some event procedures listed correctly. o 1 mark: Few event procedures listed correctly. o 0 marks: Not attempted or no event procedures listed correctly.	**8**
5	Design the procedures required: • Description of algorithms by use of any of the following: o Prose o Flow chart(s) o Pseudocode o Snippets of program code • Full functionality of the program should be clearly described. This will include the following processes: o 2 marks: Check that the number of core activities chosen is two or three. o 2 marks: Ensure the number of optional activities chosen is between zero and two. o 2 marks: Ensure the total number of activities chosen (core and optional) is between two and four. o 2 marks: Prompt user for different core activities choices once sold out (keep track of availability levels) o 2 marks: Prompt user for different optional activities choices once sold out (keep track of availability levels) o 2 marks: Find the total price for the activity day (price of the day plus cost of optional activities plus tickets delivery charge)	**12**
	Total available marks for design	**40**

Implementation

	Guide	Maximum Marks
1	Write the program code and procedures outlined in the design: • Sensible variable names o 2 marks: All variable names are sensible o 1 mark: Some attempt at using sensible variable names o 0 marks: No attempt to choose suitable variable names • Suitable variable types o 2 marks: All variable types are suitable o 1 mark: Some variable types are suitable o 0 marks: No attempt to choose suitable variable types • Correct assignment syntax o 2 marks: All variables suitably assigned o 1 mark: Some variables suitably assigned o 0 marks: No variables suitably assigned • Correct selection syntax o 3 marks: Correct selection syntax for all cases including compound conditions o 2 marks: Correct selection syntax for all cases except compound conditions o 1 mark: Multiple errors but at least one correct selection statement o 0 marks: Not attempted or no correct implementations • Correct iteration syntax o 2 marks: Correct iteration syntax for all cases where used o 1 mark: At least one correct instance of iteration o 0 marks: Not attempted or no correct implementations • Error handling. Award marks for each of the following, if implemented correctly: o 1 mark: Use of controls requiring selection rather than free-form input where possible to minimise errors o 1 mark: Range validation o 1 mark: Data type validation o 1 mark: Additional validation (e.g. length of input) where suitable • Suitable display mechanism(s) o 2 marks: Suitable display mechanisms for all cases where used (e.g. message box where asked for, otherwise label or other suitable mechanism) o 1 mark: Some information displayed using suitable mechanisms o 0 marks: Not attempted or no suitable implementations	

	Guide	**Maximum Marks**
	• Suitable data capture mechanism(s) o 2 marks: Suitable data capture mechanisms for all cases where used o 1 mark: Some use of suitable data capture mechanisms but over-reliance on free-form typing o 0 marks: Not attempted or no suitable mechanisms chosen (e.g. all input requested via Input Boxes which would be laborious and not event driven) • Clear annotation o 2 marks: Program fully and clearly commented o 1 mark: Program sparsely commented o 0 marks: Program not commented, or no comments bear any relation to the code they are supposed to be explaining • Correct logic o 1 mark: Program checks an option for each required component has been selected o 1 mark: Program checks that the number of core activities chosen is between two and three o 1 mark: Program checks that the total number of activities chosen (core and optional) is between two and four o 1 mark: Program informs the user if a core activity is fully booked for their chosen day and asks them to choose again o 1 mark: Program informs the user if an optional activity is fully booked for their chosen day and asks them to choose again o 1 mark: Program will prompt users if they fail to select an option for one of the required components o 1 mark: Program correctly calculates the cost of the booking (Activity Day cost plus optional activities plus ticket delivery charge) o 1 mark: Program correctly displays the final cost to the user o 1 mark: Program accepts contact and credit card details correctly, with appropriate data validation	**30**
Total available marks for implementation		**30**

Testing

	Guide	Maximum Marks
1	Design a testing strategy for each procedure: 1. Normal tests 2. Extreme tests 3. Exceptional tests • 15 marks: Award full marks if there is a testing strategy for each procedure, which includes normal, extreme and exceptional test cases where appropriate for each item of functionality • 12 – 14 marks: Testing strategy is thorough but one or two procedures or items of functionality have been omitted • 8 – 11 marks: Testing strategy covers most procedures/ items of functionality but does not include extreme or exceptional tests OR these are covered but some important functionality is not tested • 4 – 7 marks: Testing strategy covers less than half the functionality and only includes tests for normal data • 1 – 3 marks: Nominal attempt at a testing strategy for at least one item of functionality • 0 marks: No testing strategy for any of the procedures	**15**
2	Design a testing strategy for the complete system: • 1 mark: Suitable integration testing • 1 mark: Normal tests • 1 mark: Extreme tests • 1 mark: Exceptional tests	**4**
3	Provide evidence, including **sample** screenshots, or tests • 3 marks: Comparison of actual outcomes with expected outcomes. Deduct marks if no comparisons are made or tests do not match what the program actually does • 3 marks: Clear screenshots showing actual outcomes. Deduct marks if screenshots show different outcomes from what is listed as test results or if only a few screenshots are provided	**6**
	Total available marks for testing	**25**

Publishing programme and installation documentation

	Guide	Maximum Marks
1	• 3 marks: Produce a publishable working copy of a compiled version of the completed assignment. • Produce installation notes. These should clearly explain: o 1 mark: How to install the program o 1 mark: What the system requirements are	**5**
	Total available marks for publishing programme and installation documentation	**5**

Learning Outcomes matrix

Task	**Learning Outcomes assessed**	**Marker can differentiate between varying levels of achievement**
Design Task 1	1	Yes
Design Task 2	1	Yes
Design Task 3	2	Yes
Design Task 4	1	Yes
Design Task 5	1	Yes
Implementation	2, 3, 4	Yes
Testing Task 1	1	Yes
Testing Task 2	1	Yes
Testing Task 3	1	Yes
Publishing	2, 3, 4	Yes

Grade descriptors

Learning Outcome	**Pass**	**Merit**	**Distinction**
Create project documentation	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses data capture and validation	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses sequential programming with different data types	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses iteration and selection constructs	Demonstrate adequate level of understanding	Demonstrate robust level of understanding	Demonstrate highly comprehensive level of understanding
Implement a program that uses file I/O	Demonstrate ability to perform the task	Demonstrate ability to perform the task consistently well	Demonstrate ability to perform the task to the highest standard
Implement a program that uses arrays	Demonstrate ability to perform the task	Demonstrate ability to perform the task consistently well	Demonstrate ability to perform the task to the highest standard