Te Hoe Hōkai Pakihi
Department of Business and Digital Technologies

Bachelor of Information and Communication Technologies

Graduate Diploma in Information and Communication Technologies

Assessment Two:
Practical Assessment 2 MARKING SHEET

Advanced Programming
BCDE321

Semester Two 2022

Due date: Friday 26 August 2022

Time: 5:00 pm

Instructions:
See page 2.
This is an Individual Assignment.

TOTAL MARKS: **60**

Student Name/ ID .....

Other group members .....

If a learner needs to apply for an extension, they can do so by completing the extension request form (app505m-extension-of-time-application.pdf). Extension requests must be submitted to the lecturer prior to the assessment due date.

If an assessment is handed in late without an approved extension, a penalty of 10% per day will apply, up to a maximum of 50%. If an assessment is received more than five days after the due date without an approved extension, it will not be marked. Should a learner wish to appeal any decisions, they may do so in writing to the Head of Department within ten days of receiving the decision.

This assessment is worth 25% of the total marks for this course. To pass this course, learners must gain an average of at least 50% across all assessments, and gain at least 50% in Assessment 3.

This paper has four (4) pages including the cover sheet.

Requirements for Submission

Every learner MUST submit the followings as a single .zip file into the drop box on the course Moodle site by the deadline indicated; otherwise, ZERO mark may be given.

1 A design-level class diagram of your proposed program.

2 A help file details the commands provided by your line-oriented command interpreter and the course tutor must approve these before you start the coding for this assessment.

3 Your program must be towards solving the tasks mentioned in the section of Problem Domain.

4 Your code MUST comply with the Python coding style, i.e., PEP8.

5 A document to list (for each component claimed for marks in your program)

(a) the ownership (i.e., done by you or someone else?)

(b) self-reflection on robustness[1]

(c) self-reflection on the completeness and implementation.

6 You must carry out version control in an online repository during your development process. The URL link of your online repository needs to be provided in your self-marking sheet.

7 A filled self-marking sheet.

URL Link of Your Online Repository

[1] **Robustness**. The degree to which a system continues to function in the presence of invalid inputs or stressful environmental conditions.

MARKING GUIDE **TOTAL: 60 MARKS**

	Component	Used by your peers (2 marks)	Robustness (2 marks)	Complete and well implemented, i.e., "What is clever about this?" (2 marks)	Marks
1	Support command-line arguments				
2	Has a line-oriented command interpreter based on cmd or similar package				
3	Display command line help of available commands				
4	Change commands and options				
5	Extract data				
6	Validate data				
7	Provides object-persistence / object serialization using either pickle or shelve				
8	Can load data from a file				
9	Can deal with file directory				
10	Can raise exceptions and provide exception handling				
11	Amount of checking for pre- or post- conditions of methods				
12	Provide doctests				
13	Provide unittests				
14	Pretty print				
15	Can save and read data from a database				
Total					

	Marks		
	0	**1**	**2**
Used by peers	Not used by any peer	Half of the team members use	All team members use
Marking Rubric for Robustness			
Provide doctests	No testing case passes	Half of the testing cases pass	All testing cases pass
Provide unittests	No testing case passes	Half of the testing cases pass	All testing cases pass
Other components	Not be able to run during demonstration	Encounter some unhandled exceptions during demonstration	Encounter ZERO unhandled exception during demonstration
Complete and well implemented			
Checking pre-/ post-conditions of methods	No checking	<= 15 different checking	>=30 different checking
Provide doctests	No doctest	<= 15 different doctests	>= 30 different doctests
Provide unittests	No unittest	<= 15 different unittests	>= 30 different unittests
Other components	Not complete	Complete, but not very Pythonic	Complete and very Pythonic