Assessment Task 3: Coding Task 2

Start Assignment

Due May 24 by 11:59pm **Points** 30 **Submitting** a file upload **File Types** zip
Available Apr 26 at 6:59pm - Jun 4 at 11:59pm

Coding Task 2

Assignment name: Coding Task 2

Course code: 2676/2752

Weighting: 30%

Due date: Wednesday May 24th, 11:59pm

l. Course Learning Outcomes Assessed

This assessment supports the following learning outcomes:

- CLO 1: Demonstrate knowledge of basic concepts, syntax and control structures in programming.
- CLO 2: Devise solutions to simple computing problems under specific requirements.
- CLO 3: Encode the devised solutions into computer programs and test the programs on a computer.
- CLO 4: Demonstrate understanding of standard coding conventions and ethical

considerations in programming.

2. Overview of Assessment

For this assignment you are expected to create a study/simulation/tool/utility of your choice related to your program of study using Python. Your work needs to comprise a number of features, as described in Section 3.1. You are also required to implement the code concepts, e.g., data types, control flow, etc, detailed in Section 3.2, and analyse your code and provide adequate documentation, as described in Section 3.3.

- Your program **must** relate to your program of study.
- You must not just "throw in the required code concepts" to your program just because they need to be there; it should be clear from the code why a certain concept is there, and you must further explain these through your comments and presentation.
- You will also need to debug your code on your own and document any issues, etc.

You are given marks on your ability to fulfil all requirements of this document.

3. Assessment Tasks

There are design requirements (5 marks), code requirements (18 marks), and documentation requirements (7 marks) for a total of 30 marks.

3.1. Design Requirements

Your program needs to support at least four features that serves the objectives of your study/simulation/tool/utility. The following design requirements must be clearly stated in your presentation (further detailed in Section 3.3).

3.1.1 The objective(s) of the study/simulation/tool/utility are clearly outlined and its relevance to your program of study. **(1 mark)**

3.1.2 A clear and concise discussion of each feature of your program. Note that a feature is a distinguishing characteristic of a program (i.e., what a program can do). The mechanics of a program (i.e., how a program works) is not a feature. **(4 marks)**

3.1. Code Requirements

The following code requirements must be applied to serve the objective of the proposed program.

3.2.1 Presentation, formatting and design. The filename(s) must be chosen to match the application (default names such as assignmentX.py will not be accepted). Names of identifiers (including functions, variables, etc.) must be descriptive. Code formatting is clean and consistent. All libraries (if any) are imported at the top of the file. The design and organization of the overall program is sensible, modular, and readable and your code does not include any unused/irrelevant code. **(2 marks)**

3.2.2 *Data types.* Your program must process at least four different data types with at least two data types of the following: range, list, tuple, set, dictionary, or arrays. **(1.5 marks)**

3.2.3 *Control flow.* The program must have at least three control flow blocks, i.e., if/elif/else and/or for/while loops. **(1.5 marks)**

3.2.4 *Files.* Your program must use at least two external input/output file(s) (e.g., .txt, .csv, or .png). Note: you will need to submit these files. **(2 marks)**

3.2.5 *Functions*. Demonstrate concepts of modular code organisation using at least four functions. When the program runs, there should be a pathway to run/call all functions (i.e., the program must not have any unused function). **(2 marks)**

3.2.6 *Scientific Python & visualisation*. Your program must use at least two out of the following three packages: NumPy, SciPy and Matplotlib. **(6 marks)**

3.2.8 *Web data crawling***.** Your program must access the Internet, search for your required data (information) and save the retrieved data. You are only allowed to use re and urllib libraries for this task and will not receive marks if this condition is not met. **(3 marks)**

3.3. Documentation Requirements

The following documentation requirements must be included as part of your final submission.

3.3.1. Create and submit a file called ReadMe in which:

- In the first line(s) you include an example of the command(s) needed to run your program to demonstrate all of the implemented features. **(0.5 mark)**
- In the next lines you document which files and lines in your code addresses the code requirements from 3.2.2 to 3.2.5. If your program consists of more examples for each code requirement, e.g., more than four functions, you only need to document the required number of examples. **(2 marks)**

3.3.2 Submit a file including the link to a 2-5mins video or a screen recording that contains the following (please note that the first three items correspond to the design requirements outlined in 3.1):

- Introduction (including objective and specifications of the study/simulation /tool/utility)
- Relation of the study/simulation/tool/utility to your program of study
- Features
- Discussion on the methodology/design and the overall structure of your program (feel free to use flowcharts and slides if you wish). **(1.5 marks)**
- Demonstration of how to run the program and sample output(s). **(2 marks)**
- Reflection on the resources you used while designing and implementing your program. This includes how you ensured to (i) acknowledge any potential program or resource that you used, (ii) ask for permission if you used licensed code, (iii) document your code as accurately as possible, and (iv) handle errors and exceptions in your code. **(1 mark)**

4. Submission Instructions

Submit .py file(s), any data/input files, the link to the video as well as the ReadMe file via Canvas → Assignments → Assessment Task 3: Coding Task 2 as a single .zip file called *a3_{firstname}_{lastname}_{student_id}.zip*.

It is the responsibility of the student to correctly submit their files. Please verify that your submission is correctly submitted by downloading what you have submitted to see if the files include the correct content. Please make sure your video is accessible via the provided link (e.g., visible from an RMIT account in case you use Google Docs).

Assessment declaration:

When you submit work electronically, you agree to the assessment declaration outlined **here (https://www.rmit.edu.au/students/student-essentials/assessment-and-exams/assessment/assessment-declaration)** .

5. Assessment Criteria

Your program will be assessed based on the clarity of the presentation, the soundness of your design and correctness of your code. Please note that the following leads to a deduction of marks:

- Unnecessarily complicated code designs.
- Not including adequate comments to your code blocks and function definitions.
- Importing libraries other than Python in-built libraries, NumPy, SciPy and Matplotlib.
- Submitting .ipynb files.
- Including the video file in the submitted folder.

The following rubric outlines the criteria against which your submission will be assessed.

Coding Task 2

Criteria	Ratings			Pts
3.1.1 Objective(s) and relevance	**1 pts** **Complete** The objective(s) and relevance of the study/simulation /tool/utility are clearly outlined.	**0.5 pts** **Incomplete** The discussion on the objective(s) or relevance of the program is not clear or missing.	**0 pts** **No marks** No discussion about the objectives is provided.	1 pts
3.1.2 Features	**4 to >2.0 pts** **Complete** There are at least four features that serve the objective(s) of the proposed program. The discussion about all of them is clear.	**2 to >0.0 pts** **Incomplete** The program does not introduce at least four features OR the discussion about the features is not sound/clear.	**0 pts** **No marks** No discussion about the features is provided.	4 pts
3.2.1 Presentation, formatting and design	**2 to >1.0 pts** **Complete** All of the following criteria are met: (i) File name suits program and is not an arbitrary or default name. (ii) Identifier names are descriptive and the presentation is consistent. (iii) Only relevant code and comments are included. (iv) The design and organization of the overall program is sensible, modular, and readable.	**1 to >0.0 pts** **Incomplete** More than two of the criteria in the ‘complete’ level are not met.	**0 pts** **No marks** None of the criteria in the ‘complete’ level are met.	2 pts
3.2.2 Data types	**1.5 to >0.75 pts** **Complete** Program processes at least four different data types with at least two data type of the following: list, tuple, set, dictionary, or arrays. The program demonstrates understanding of the	**0.75 to >0.0 pts** **Incomplete** The program does not use four different data types as specified OR the use of some of the data types is incorrect or not	**0 pts** **No marks** The program does not have any sound and/or correct use of	1.5 pts

Criteria	Ratings			Pts
	difference between types where relevant and the program uses different data types in a sound way.	sound.	data types.	
3.2.3 Control flow	**1.5 to >0.75 pts** **Complete** The program must have at least three control flow blocks. Uses if/elif/else appropriately for non-repeating conditional execution. Conditions do not include tautologies, e.g., "x=y or x≠y", and pathways are not redundant. Every code block in every if/elif/else structure is reachable. The program uses loops appropriately for repetition. Loop condition must describe all situations under which the loop will repeat, and condition must fail eventually.	**0.75 to >0.0 pts** **Incomplete** Some of the control blocks are not sound/correct OR are irrelevant to the objective of the program.	**0 pts** **No marks** The program does not have any sound, correct and/or relevant control block.	1.5 pts
3.2.4 Files	**2 to >1.0 pts** **Complete** Program correctly handles (reads or writes) at least two external file(s). The use of files is a part of meeting the objective(s) of the program. All input/output files are included in the final submission.	**1 to >0.0 pts** **Incomplete** Only one external file is read/written OR the file handling is not sound/relevant to the objective(s) of the program. The input/output files are missing.	**0 pts** **No marks** The program does not have any sound, correct and/or relevant external file handling.	2 pts
3.2.5 Functions	**2 to >1.0 pts** **Complete**	**1 to >0.0 pts** **Incomplete**	**0 pts** **No marks**	2 pts

Criteria	Ratings			Pts
	The program has at least four functions in addition to a potential "main" function. Functions used in ways to reduce code repetition. Every created function is used in the program. Methods may contain parameters and/or return values. The functions are all implemented as a part of meeting the objective(s) of the program.	Less than four functions are implemented. The function definitions are not sound, correct or relevant to the objective(s) of the program. The functions are not commented clearly and/or are not used in the program.	The program does not have any sound, correct and/or relevant function definition.	
3.2.7 Scientific Python & visualisation	**6 to >3.0 pts** **Complete** The program uses at least two out of the three specified packages. Modules are imported and used correctly and appropriately. References to the modules' documentation are provided. The modules are used as part of meeting the objective(s) of the program.	**3 to >0.0 pts** **Incomplete** The program uses only one of the specified packages OR the use of imported modules are not sound, correct or relevant to the objective(s) of the program.	**0 pts** **No marks** The program does not import any of the specified packages or make any sound, correct and/or relevant use of them.	6 pts
3.2.8 Web data access	**3 to >1.5 pts** **Complete** The program uses re and urllib libraries to access, search and retrieve information from the web. The web crawling process is sound, correct and relevant to the objective(s) of the program.	**1.5 to >0.0 pts** **Incomplete** The program only searches or retrieves information from the web. The search process does not use regular expression and/or the retrieval is not sound, fully correct or relevant to the objective(s) of the program.	**0 pts** **No marks** The program does not search or retrieve any data on the web.	3 pts
3.3.1 ReadMe file	**2.5 to >2.0 pts** **Complete** The first line(s) of your ReadMe file includes an example of the command(s) needed to	**2 to >0.0 pts** **Incomplete** Some of the listed requirements in the	**0 pts** **No marks** ReadMe file is not	2.5 pts

Criteria	Ratings			Pts
	run your program to demonstrate all the implemented features of the program. The rest of the lines list the files/line numbers where the code requirements in 3.2.2 to 3.2.5 are implemented.	'complete' level are missing or are incorrect.	submitted.	
3.3.2 Video Presentation	**4.5 to >2.5 pts** **Complete** Video contains the specified requirements in 3.3.2. The presentation is clear, sound, and correct and is not shorter/longer than specified.	**2.5 to >0.0 pts** **Incomplete** Some of the required discussions are missing, unclear or incorrect. The video is longer than specified.	**0 pts** **No marks** The link to the video is not submitted, is broken or the content of the video is not addressing the requirements.	4.5 pts
				Total Points: 30