Solent University
Coursework Assessment Brief

AE2 Assessment Details

Module Title:	Problem Solving Through Programming
Module Code:	COM411
Module Leader:	Daniel Olabanji
Level:	4
Assessment Title:	Software Artefact with Report
Assessment Number:	AE2
Assessment Type:	Software Artefact with Report
Is this a 'must pass' element of assessment (Y/N):	Yes
Restrictions on Time/Word Count:	4 code files (+ additional classes for A grade)
Consequence of not meeting time/word count limit:	It is essential that assignments keep within the time/word count limit stated above. Any work beyond the maximum time/word length permitted will be disregarded and not accounted for in the final grade. *
Individual/Group:	Individual
Assessment Weighting:	50%
Issue Date:	13th October 2025
Hand In Date:	9th January 2026
Planned Feedback Date:	4 weeks after deadline
Mode of Submission:	On-line Only FINAL submissions will be accepted. DRAFT submissions will not be considered an attempt and will not be marked.
Anonymous Marking	This assessment: Will be marked anonymously or

Assessment Task

In this assessment, you will develop an application to analyse data from a publicly available dataset. The dataset will be provided to you as a CSV file, which accompanies this assessment brief.

Accompanying CSV file

The CSV file is called Disneyland_reviews.csv and contains thousands of reviews about Disneyland parks around the world. Disneyland_reviews.csv contains the following information, with each row in the file representing a single review:

Column	Description	Type
Review_ID	Just a unique identifier, contains no useful information	Int

Rating	The score given by the reviewer out of 5.	Int
Year_Month	The year and month the review was made.	String
Reviewer_Location	Where the reviewer is from.	String
Branch	Which Disneyland Park is being reviewed	String

It is recommended that you familiarise yourself with the content of the data file before attempting the remainder of this assessment.

Accompanying Code Files

In addition to the CSV file, you are also provided with a zip file containing 4 python modules:

main.py
tui.py
process.py
visual.py

You are expected to confine your code to these four modules. Creating additional modules or unexpected code architectures will be seen as a lack of understanding.

The A grade task, however, will require you to use OOP and therefore develop classes. These should be placed in their own module as best practice requires. Therefore, any OOP classes you create are the only exception to this rule. All other code must be within the four file provided modules.

The Software Artefact

This section outlines what you are expected to code within the provided modules. The coding tasks are separated into sections that will get progressively more involved. As you can expect, if you are looking to achieve a high grade, you will need to complete all the sections, whereas a D grade can be achieved even if some later sections have not been attempted.

Please note, completing all the tasks will not achieve a high grade on its own, you will also need to code to a good standard:

- Submissions in which most of the code is written in main.py, in a primarily linear fashion will not score highly.
- In contrast, submissions that logically break down their code into reusable functions/classes, place functions in the most logical of the provided modules, make very good use of parameters, return values and OOP and conform to PEP8 standards will score highly.

You are expected to use Git and GitHub as part of this assessment. Your GitHub repository must be private!

Please see the assessment criteria (further down in this document) for more information about how this assessment will be graded.

Below are the individual tasks, you should attempt each section one at a time, starting with 'Section A' and working through in the correct order. You are strongly discouraged from attempting the sections in a different order (i.e., doing 'section C' before 'section B'). This will almost certainly lead to difficulties with the later tasks.

Please note. The program should run from **main.py**! The marker should not have to run any of the other module separately to see functionality. This means, if you use the other modules, then they will need to be imported in to main.py and implemented correctly.

----- Section A -----

We shall begin in main.py. You may create functions and separate them into the other modules – however, how you do this is strictly down to you.

1. First, when running the program, it should display the following (take note of the annotation):

```
-------------------------
Disneyland Review Analyser
-------------------------
```
The number of dashes should equal the number of characters in the title

2. Read in the data from the provided CSV file, said data should be stored in a list. The program should confirm to the user when it has finished reading in the dataset. It should also tell the user how many rows are in the dataset.

3. Output the following to the screen, this will act as a menu. The user should then be able to input their selection, which should be stored in a suitable variable.

```
Please enter the letter which corresponds with your desired menu choice:
    [A] View Data
    [B] Visualise Data
    [X] Exit
```

4. The program should confirm what the user has entered. If the user entered an invalid menu choice, then they should be informed of their mistake.

   Here is an example of what it would look like if the user entered 'A':

```
Please enter the letter which corresponds with your desired menu choice:
    [A] View Data
    [B] Visualise Data
    [X] Exit
A
You have chosen option A - View Data
```

5. The program should run continuously. Once the user has entered their choice, it should confirm this choice and then display the menu again, asking the user for their selection. Be careful how you implement this, the program <u>does not</u> need to display the title and load in the dataset again. The program should only end if the user indicates they wish to exit the program through the appropriate menu choice.

6. If the user selects 'A' as their menu choice, the program should display the following
sub-menu and receive the user's choice:

```
Please enter one of the following options:
    [A] View Reviews by Park
    [B] Number of Reviews by Park and Reviewer Location
    [C] Average Score per year by Park
    [D] Average Score per Park by Reviewer Location
```

If the user selects 'B' as their menu choice, the program should display the following sub-menu and receive the user's choice:

```
Please enter one of the following options:
        [A] Most reviewed Parks
        [B] Park Ranking by Nationality
        [C] Most Popular Month by Park
```

----- Section B -----

7. We will now focus on the sub-menu that is displayed should the user choose 'A' at
the main menu.

   The first sub-menu option will allow the user to see all the reviews for a specific park. If the user selects this option, then the program should respond by asking which park the user wishes to see the reviews for. The program should then display all reviews for said park.

8. The second sub-menu option will simply display the number of reviews a specific park has received from a given location. Both the park and the reviewer's location should be retrieved from the user.

9. If the user chooses the third sub-menu option, then the program will ask the user for a park and a year, it will then display the average rating for the given park in the given year.

The final option for this sub-menu will be developed in a later section.

----- Section C -----

We will now focus on the sub-menu that is displayed should the user choose 'B' at the main

menu.

10. The first sub-menu option should display a pie chart showing the number of reviews each park has received.

11. The second sub-menu option will ask the user to enter a park. It will then display a bar chart that shows the <u>top 10</u> locations that gave the highest average rating for that park.

12. The final sub-menu option will ask the user to enter a park. It will then display a bar chart that shows the average rating that park received for each month of the year. You do not need to worry about the year for this task, so May 2018 and May 2019 will both be simply counted as May.
The bar chart should be ordered by month.

   The final output should be a bar chart with the months of the year (in order) along the X axis and the average rating on the Y axis.

----- Section D -----

13. Moving back to the submenu that is displayed if the user enters 'A' in the main menu:

   The final sub-menu option for the 'view data' menu is to display the average score per park by reviewer location.

   This, for every park, should output the average rating for every single location it has received a review from.

   The user should not have to enter any information for this task.

14. To achieve the highest grades, you will need to demonstrate Object Orientation. You will need to develop an 'export data' feature using OOP. The exporter should output aggregate information for each park including Number of reviews.
Number of positive reviews. All review should be taking into account in this calculation Average review score.

   Number of countries that have reviewed each park.

   The system should provide the option for the output to be in any of the following 3 formats: TXT, CSV or JSON.

   You should add an additional option (C) to the main menu, allowing the user to access this feature and select the format they wish to output the data to.

   You should include a rough class diagram for this in your supporting documentation (see below).

The Supporting Documentation

In addition to the software artefact, you will also need to submit either a word document or a PDF that contains the following (and no more!):

- A table that declares which tasks you feel you have completed, which were attempted but incomplete and which were not attempted.

- Evidence of your use of Git/GitHub. This should be in the form of two screenshots. The first should be a screenshot of your most recent commits, as displayed in the commit history page on GitHub. The second screenshot should show the oldest commits as listed in your commit history on GitHub. Your username should be clearly visible in the screenshots. You should include as many commits in the screenshots as you can fit.

- Evidence that you have tried to conform to PEP8 standards. This can be evidenced by a screenshot of PyCharm either confirming there are no errors or a screenshot of the errors that exists with a short statement confirming that you were unable to fix them.

- If you attempted the final task – a rough class diagram showing how you implemented OOP into the program.

Technologies

You must use Python and only **Python** for this assessment.

Only one third party library may be used, that being **MatPlotLib**.

Deliverables (What you should submit)

You will need to submit 2 files in total:

1. A **.zip** file containing the latest working version of your software project

Note:
1. Please use your student number as the file names Both files MUST be submitted as instructed.

2. Submissions that are missing the zip file or are corrupted will be automatically allocated an 'Fail' grade.

3. Submissions that are missing the supporting documentation, or the supporting documentation does not follow the points outlined in the above section, will be severely impacted!

Please read the FAQ section under the 'assessment' tab on the SOL page.

Assessment (AE2) criteria

	Marking criteria	Weighting	MLOs	0 (0-19) Fail	2 (20-29) Fail	3 (30-39) Fail	4 (40-49) Pass	5 (50-59) Pass	6 (60-69) Merit	7 (70-79) Distinction	8 (80-89) Distinction	9 (90-99) Distinction	10 (100) Distinction
1	**Knowledge and understanding** Evidence of knowledge of the Python programming concept and principles.	20%	MLO1	No attempt. Program did not run with large errors. Python programming language was not used.	Poor attempt at evidencing problem-solving using Python programming, No Git/GitHub repository details.	Limited understanding of problem-solving using Python programming. Inadequate command (Example: If, else, and if...else statements and No Git/GitHub repository details.	Acceptable /Basic Python usage with errors in most of the tasks. Basic understanding of the flow of Python programming. Basic usage of Git/GitHub repository and commands.	Fair attempts at the tasks. Fair application of the principles associated with its analysis and interpretation. Fair usage of Git/GitHub.	Competent attempt at the tasks provided in the assessment. Competent level of understanding of object-oriented programming (OOP) and using a repository such as Git/GitHub.	Proficiently attempt the tasks using Python and a dataset. Very good report writing and proficient level of Python and OOP and using a repository such as Git/GitHub. The code followed PEP 8.	Efficiently attempt the tasks using Python and the provided dataset. Excellent level of usage of Python, OOP, and Git/GitHub. The code followed PEP 8	Outstanding attempt at all tasks using Python and the provided dataset. Superb level of proficiency in Python and OOP, and an understanding of using a repository such as Git/GitHub. The code followed PEP 8	Extraordinary attempt at all tasks using Python and the dataset provided. Industry-standard proficiency in coding in Python and using repository such as Git/GitHub. The code is perfectly written in PEP 8 with functions and classes.

| 2 | Cognitive skills
Demonstrate the ability to present, evaluate and interpret the data. | 15% | MLO - MLO4 | No attempt. No evidence of evaluation or interpretation | Poor ability to analyse, present, evaluate, and interpret the dataset. Misleading, off-topic or irrelevant to the module. | Limited ability to analyse, present, evaluate, and interpret the dataset's output—inconsistencies in the documentation and report. | Basic attempt to analyse, present, evaluate, and interpret the dataset's output. The report lacks professional depth. | Reasonable attempt to present, evaluate and interpret the dataset. The report presents relevant views and thoughts that enhance its professional quality. | Competent attempt at demonstrating the ability to present, evaluate and interpret the dataset. The report is logically sound, with appropriate and relevant views, and the code is user-friendly and professional, featuring clear and informative comments. | A solid understanding to evaluate and flawlessly interpret the dataset. The report is logical, very good, with appropriate and relevant views. The code has undergone self-improvement, becoming both user-friendly and professional, with comments provided. | Highly impressive to evaluate and interpret the dataset and provide practical usage of its analysis and interpretation. The report is professionally written, and the code is modular and well-documented, featuring improved user-friendly features and an enhanced user experience. | Outstanding evaluation and interpretation of the dataset, providing original analysis and interpretation. The report is professionally written, and the coder is modular, demonstrating great individual creativity and a superb user-friendly interface for easy application usage. | Extraordinary and Flawless evaluation and interpretation of the dataset. The report is a thorough, industry-standard analysis that demonstrates professionalism. |

| 3 | Practical and Professional Skills | 50% | MLO 1, MLO 3 | No attempt. Zip file was corrupt or empty or unable to open.

OR

No supporting documentation was submitted.

OR
Evidence of plagiarism, collusion, or any other form of academic misconduct | Code does not run properly

OR

The supporting documentation was submitted perfectly {separately to generate a Turnitin report} | Submitted code consist of libraries that are unapproved as outlined in the assessment brief

OR

Unexpected architectures that were very different to what was taught in the module. Git/GitHub was either not used or was no evidence of its usage. | Section A' tasks have been completed and produced correct results.

AND

Supporting documentation have been submitted and was in line with the requirement.

AND

Code was linear and unorganised into modules. | Section A And B tasks have been completed and produced correct results.

AND

Functions were proactively used and logical. And some of the functions were organised into the provided modules.

AND
Code was mostly written to a good standard, and the system was generally user-friendly. | Section A, B and C tasks have been completed and produced correct results.

AND

Functions are widely used and proactively organised into logical modules. Parameters and return values logically.

AND
All requirements up to this point were met. Charts were fully labelled and nicely presented. | Section A, B, C and D tasks have been completed and produced correct results.

AND

The code was written to an excellent standard that logically breaks down code into reusable functions/classes.

AND

Chart are Labelled in a well structure ways and proficiently presented | Excellent presentation of the code and statements were excellently written.

AND

Evidence of problem solving and professionalism used in the development of the parameter and tasks.

AND

Chart are Labelled in a well structure ways and excellently presented | An Outstanding attempt of the attacks and close to industry standard flawless programming.

AND

Follow all the instruction for the task above and show some creativity and evidence-based professionalism and problem solving.

AND
Well-structured modules where all modules are responsive, and charts and label are superb. | A flawless, industry standard professional approach programming with a documentation that is meticulous and accurate

AND

Deep analysis and professional-level execution and implementation where all the aforementioned criterions are meet. |

| 4 | Transferable Skills Demonstrate the qualities and transferable skills necessary for employment requiring the exercise of some personal responsibility | 15% | All MLOs | No attempt. The supporting documentation was off- topic and did not follow the points requested in the 'supporting documentation' section of the assessment brief. | A poor attempt at demonstrating professionalism, version control, application support document writing qualities. | Limited attempt at demonstrating qualities and inadequate evidence of professionalism. Not fully following the instructions as requested in the assessment. | Acceptable attempt at demonstrating qualities and transferrable skills necessary for employment through satisfactory documentation. | The student demonstrated an understanding of the transferable skills and what they learned in this module that will be needed in the future module and workplace | The student demonstrate that the program runs fairly during the test by simply running main.py. | All requirements up to this point are met. The student's report demonstrated that the code is mainly written to a good standard in Python. | An excellent attempt at demonstrating qualities and transferrable skills necessary for employment and meeting all requirements. | An outstanding attempt at demonstrating qualities and transferrable skills necessary for employment and future modules. | An extraordinary attempt at demonstrating qualities and transferrable skills necessary for employment and future modules. |

Use of AI in this Assessment

Generative AI is permitted at Solent University under specific conditions and must continue to follow the university's rules around Academic Misconduct and the AI and Academic Integrity policy. In this assessment, you are allowed to use AI for the following tasks:

Generative AI may only be used to:

- **Assist with improving your report writing**
- **But it must not be used to solve any part of the assessment.**
- **It must not be used to provide solutions for any aspect of the assessment.**

If it is found that Generative AI has been used to solve this assessment, the report will be subject to failure or referred for academic misconduct. Please be aware.

AI and Academic Integrity Policy

Learning Outcomes

This assessment will enable you to demonstrate in full or in part your fulfilment of the following learning outcomes identified in the Module Descriptor:

1. Describe a problem-solving process and its value in the computing profession as well as the wider context.
2. Design, implement, test, and debug software solutions to meet requirements
3. Demonstrate awareness of contemporary techniques for the design, development, testing, correcting, deploying and documenting of software solutions from specifications and/or problem descriptions, using relevant standards and tools.
4. Apply analytical and critical thinking skills to systematically analyse and apply structured problem-solving techniques to computer problems.
5. Interpret and follow approaches to version and source control
6. Recognise legal, social, ethical & professional issues related to software development.

Important Information

Solent University Academic Regulations 2025-26

Late Submissions

You are reminded that:

i.   If this assessment is submitted late i.e. within 7 calendar days of the submission deadline, the mark will be capped at 40% if a pass mark is achieved;
ii. If this assessment is submitted <u>later</u> than 7 calendar days after the submission deadline, the work will be regarded as a non-submission and will be awarded a zero;

iii.   If this assessment is being submitted as a referred piece of work, then it <u>must</u> be submitted by the deadline date; <u>any</u> Refer assessment submitted late will be regarded as a non-submission and will be awarded a zero.

<u>Assessment regulations</u>

Extenuating Circumstances

The University's Extenuating Circumstances (EC) procedure is in place if there are genuine short term exceptional circumstances that may prevent you submitting an assessment. You are able to self-certify for up to two assessment dates in any semester without supporting evidence for an extension of up to seven calendar days for coursework or to defer an exam to the resit period.

Alternatively, if you are not 'fit to study' (or you have used up your two self-certification opportunities), you can request:

- an extension to the submission deadline of 7 calendar days, or
- a request to submit the assessment at the next opportunity, i.e. the resit period (as a Defer without capping of the grade).

In both instances you must submit an EC application with relevant evidence.   If accepted under the university regulations there will be no academic penalty for late submission or non-submission dependent on what is requested.   You are reminded that EC covers only short-term issues (20 working days) and that if you experience longer term matters that impact on your learning then you must contact the Student Hub for advice.

Please find a link to the EC policy below:

[Extenuating Circumstances](#)

Academic Misconduct

Any submission must be your own work and, where facts or ideas have been used from other sources, these sources must be appropriately referenced. The University's Academic Regulations includes the definitions of all practices that will be deemed to constitute academic misconduct. You should check this link before submitting your work.

Procedures relating to student academic misconduct are given below:

[Academic Misconduct](#)

Ethics Policy

The work being carried out must be in compliance with the university Ethics Policy. Where there is an ethical issue, as specified within the Ethics Policy, then you will need an ethics release or ethics approval prior to the start of the project.

The Ethics Policy is contained within Section 2S of the Academic Handbook:

[Ethics Policy](#)

Grade marking

The University uses an numeric grade scale for the marking of assessments. More detailed information on grade marking and the grade scale can be found on the portal and in the Student Handbook.

Grade Marking Scale

Guidance for online submission through Solent Online Learning (SOL)

Online Submission