Solent University
Coursework Assessment Brief

Assessment Details

Module Title:	Programming for Problem Solving
Module Code:	COM728
Module Leader:	Jarutas Andritsch
Level:	7
Assessment Title:	The Data Project
Assessment Number:	AE1 and AE2
Assessment Type:	AE1: Software Artefact and documentation AE2: Software Demonstration
Restrictions on Time/Word Count:	AE1: **No more than 2000 words** (excluding Table of contents, Table of Figures, Index of Tables) AE2: **10 minutes**
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
If a group	-
Assessment Weighting:	AE1: 60% AE2: 40%
Issue Date:	23rd January 2023
Hand In Date:	AE1: 24th April 2023 by 4:00 pm.
Demonstration Date:	AE2: 27th April 2023 by 4:00 pm.
Planned Feedback Date:	Within 4 working weeks
Mode of Submission:	Online via SOL
Number of copies to be submitted:	AE1: Software Artefact and Document - 1 copy of a **zip** file containing the source codes :main program as Jupyter notebook (.ipynb), own-defined module as .py) and dataset for your software artefact - 1 copy of a software document in PDF format. **This should not be included in the zip file but instead submitted as a separate file.** AE2: - A screen recording of your demonstration. This should be an MP4 video file of no more than 10 minutes in duration and less than 250 MB in size. You should use the assessment submission link for AE2 on SOL to submit the recording.
Anonymous Marking	This assessment is **exempt from anonymous marking.**

Assessment Task

AE1 Software Artefact

You are required to develop a software application that addresses the problem scenario using Python and the tools specified in this assessment brief.

You must document software implementation which provide a concise and critical discussion of your solution. You should discuss how your solution has been implemented with suitable justifications.

Introduction

Airbnb, as in "Air Bed and Breakfast," is a service that lets property owners rent out their spaces to travellers looking for a place to stay. Travelers can rent a space for multiple people to share, a shared space with private rooms, or the entire property for themselves. With Airbnb, you can stay in unique and local accommodations in cities all around the world.

In this assessment, you will process, manage, and analyse data related to Airbnb listed properties in England which have been collected from September 2021-2022. You will work with this real data set which is provided to you in the form of a CSV file. The data file, Airbnb_UK_2022.csv, contains 34 columns.

Each row in the file represents a single record for a listing. The data set contains complete data for all columns for each record in the file. This means that there are **no missing values**.

The column host_verifications and amenities contain more than one value which are in the format of a list: the values will be contained within quotes and separated by commas e.g., ['email', 'phone'] or ["Oven", "Hair dryer", "Hangers", "Dishes and silverware", "Essentials", "Shampoo", "Cooking basics", "Bed linens", "Wifi", "Heating", "Coffee maker", "Backyard", "Long term stays allowed", "Host greets you", "Microwave", "Refrigerator", "Iron", "Kitchen", "Stove", "Hot water", "Washer", "Smoke alarm"]

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

Column	Description	Type
host_id	Airbnb's unique identifier for the host/user	Integer
name	Name of the listing	String
description	Detailed description of the listing	String
host_name	Name of the host. Usually just the first name(s)	String
host_since	The date the host/user was created	Date
host_location	The host's self-reported location	String
host_response_time	The response time from host	String
host_response_rate	The response rate of host	String
host_acceptance_rate	That rate at which a host accepts booking requests	String
host_is_superhost	The host is very good host to stay [True/False]	Boolean
host_total_listings_count	The total number of listings the host has (per Airbnb calculations)	Integer
host_verifications	The approach to verify the booking to host	String
host_identity_verified	Host has been verified [True/False]	Boolean
property_type	Self-reported property type.	String
room_type	Room type	String
accommodates	The maximum capacity of the listing	Integer
bathrooms_text	The number of bathrooms in the listing.	String
bedrooms	The number of bedrooms	Integer
beds	The number of bed(s)	Integer
amenities	The features or facilities provided by host	String
price	daily price	String
minimum_nights	minimum number of night stay for the listing	Integer
maximum_nights	maximum number of night stay for the listing	Integer
instant_bookable	Whether the guest can automatically book the listing without the host requiring accepting their booking request. An indicator of a commercial listing. [True/False]	Boolean
number_of_reviews	The number of reviews the listing has	Integer
first_review	The date of the first/oldest review	Date
last_review	The date of the last/newest review	Date
review_scores_rating	The review score of rating	Float
review_scores_accuracy	The review score of accuracy	Float
review_scores_cleanliness	The review score of cleanliness	Float
review_scores_checkin	The review score of checkin	Float
review_scores_communication	The review score of communication	Float
review_scores_location	The review score of location	Float
review_scores_value	The review score of value (received experience compare with the money they paid for a stay)	Float

Requirements

The requirements for the system are as follows:

a) The system will allow the user to retrieve data from a CSV file using the csv module and fundamental python (control structure and file processing) to perform the following:
   - Load the data from a CSV file into memory using the **CSV module**. The path to the file will be specified by the user then use these loaded data to perform following tasks
   - Retrieve a name of listing, host_name, description, host_location, and the date the host was created for an individual host by host_id
   - Retrieve host_name, property_type, price, minimum_nights, and maximum_nights of all Airbnb listing for a specified location
   - Retrieve room_type, accommodates, bathrooms, bedroom, and beds of all Airbnb listing for a specified property type
   - **Retrieve specific columns of your choice related to an individual host by location (at least 3 columns and should be different to previous requirements)**

b) The system will allow the user to analyse/query data using the **pandas module** to perform the following:
   - Load the data from a CSV file into memory using the pandas module. You can use the path to the file that receive from task a). Then use these loaded data to perform following tasks
   - Identifying the most popular amenities or features that Airbnb guests are looking for
   - Analyse the average price of stay in each location
   - Analyse the average review scores rating for each location
   - **Analyse to get insightful information based on your own selection (should not be the same as the previous requirements)**

c) The system will allow the user to visualise the data as follows:
   - Display the proportion of number of bedrooms of Airbnb listing using pie chart
   - Display the number of listings for each room type using bar chart
   - Display the relationship between accommodates and price using scatter plot
   - Display Airbnb prices from 2019 - 2022 with line chart using subplots (one year per plot)
   - **Display a visualisation of your choice to present customer usage behaviour on Airbnb (should not be the same as previous requirements)**

Software documentation outline

- Overview: the aim and objectives of the project and brief discussion of the dataset
- Self-reported requirement completion
- Project Implementation: Project Structure and module/functions (technically explain how the module/function works)

Expectations

The assessment must be completed individually. You must not share, in part or whole, your assessment with another party other than the module tutor and for the purpose of submission to the university. You must ensure that the University's academic misconduct guidelines are followed in their entirety.

It is expected that you will develop **a modular** software application that meets the stated requirements. You have been provided with a CSV file that contains data. Your application will need to appropriately load the data contained in this file, process the loaded data, query the loaded data, and visualise suitable information from these loaded data. You should appropriately test your implemented functionality.

You are required to evidence your work throughout your assessment. You should create a suitable private Git repository with regularly commit to your repository as you develop your solution. As part of the project and demonstration for this assessment, you will be asked to document and show your project's repository.
You should ensure that your submission complies with academic misconduct guidelines, is your own work and any external sources have been appropriately referenced.

Note: If you have any special requirement or disability, please discuss this with your tutor.

Environment

You are required to use the following tools:

- **Jupyter Notebook** as your development environment
- **Python3.9 or above** as the standard python library
- Additionally, the following libraries/modules may be imported and utilised:
    - csv – to process csv file
    - os – to retrieve or check file paths
    - matplotlib – to produce visualisations
    - random – to generate random numbers
    - pandas
- **Git Tools** and **GitHub** for version control or OneDrive
- **No other python libraries or modules should be used other than the specified**

AE2 Demonstration

You are required to upload a screen recording showing a demonstration of your final working solution. This should be an MP4 video file of no more than 10 minutes in duration and less than 250 MB in size. The recorded demonstration should show you opening your project in Jupyter Notebook to show all the structure of your project, executing it and the result of selecting option. You should also show evidence of the development of the software artefact. You need to include a voice over briefly explaining in technical aspect of the implementation. There might be a question session later if it is needed by request from tutor.

Demonstration Outline

- Introduce yourself: your name
- Brief structure of your project
- Demonstrate the project work and results:
  - Demonstrate specific columns of your choice related to an individual host by location, explaining in technical aspect of the implementation.
  - Demonstrate analysis result based on your own selection, explaining in technical aspect of the implementation.
  - Demonstrate the visualisation of your choice to present the customers' behaviour of using Airbnb, explaining in technical aspect of the implementation.

  You should justify the reason of your selection to retrieve/analyse/visualise that specify columns/information

Assessment criteria

AE1 Software Artefact (60%)

Learning Outcomes	UPPER FIRST A1 – A2 Exceed expectations in many aspects		FIRST A3 – A4 Substantially exceeds expectations		UPPER SECOND B1 – B3 (High) Meet learning outcomes and exceeds expectations in several aspects			LOWER SECOND C1 - C3 (Good) Meet learning outcomes and sometimes exceeds expectations			THIRD D1 - D3 (Competent) Meet learning outcomes			FAIL F1 – F3 (Incomplete/Poor) Fails to meet learning outcomes		
SOLENT GRADE	A1	A2	A3	A4	B1	B2	B3	C1	C2	C3	D1	D2	D3	F1	F2	F3
Design computer programs in a logical and structured way using appropriate techniques and principles	- A text-user interface has been implemented as a separate module and import to the main program and other modules when the display results - All interfaces and user-interaction are well design, organised structured using function - Well display results, clear prompts and helpful error message presents when appropriate		- A text-user interface has been implemented as a separate module and import to the main program and other modules when the display results - All interfaces and user-interaction are well design with some structured using function - Well display results and helpful error message presents when appropriate		- A text-user interface has been implemented as a separate module and import to the main program and other modules when the display results - Most of the interfaces and user-interaction are good design using and good display results with clear prompts - Adequate helpful error message presents when appropriate			- A text-user interface has implemented with as a separate module and import to the main program and other modules when the display results - Some of the interfaces and user-interaction are good design using and good display results with clear prompts - Display results in readable layout or format			- A simple user interaction message is used, all code is in the same one file - Plain display results with simple layout or format - Some attempt coding use static data			- No user interface, no user interaction message has been implemented. All attempt coding use static data or initialise data		- No evidence of attempting required threshold

Develop computer programs aligned to appropriate programming standards and code conventions	- The software artefacts demonstrates that all the requirements (100%) of the problem scenario are implemented, and they all work correctly, free of syntax errors, follow all of the requirements **and** - The program makes extensive use of functions (user-defined and built-in function) and modularise the code. The function and module work correctly as it designed and correct concept of function and module and	- The software artefacts demonstrates that at least 90% of the requirements of the problem scenario are implemented, and they work correctly, free of syntax errors, follow all of the requirements **and** - The program good utilises of functions (user-defined and built-in function) and modularise the code. The function and module work correctly as it designed and correct concept of function and module and	- The software artefacts demonstrates that at least 75% of the requirements of the problem scenario are implemented, and they work correctly, free of syntax errors, follow all of the requirements **and** - The program mostly uses functions (user-defined and/or built-in function) to implement the code. The function and module work correctly as it designed and correct concept of function and module and	- The software artefacts demonstrates that at least 50% of the requirements of the problem scenario are implemented, and they work correctly, free of syntax errors, follow all of the requirements **and** - The program uses some user-defined functions to implement the code. The function and module work correctly as it designed and correct concept of function and module and	- The software artefacts demonstrates that at least 25% of the requirements of the problem scenario are implemented, and they work correctly, free of syntax errors, follow some of the requirements **and** - The program uses a few user-defined functions or use lots of functions, but it doesn't work as it designs for/correct concept e.g., define function with some parameters but the code inside function doesn't use the parameter at all or try to use function but just have function name	- The software artefacts demonstrates that only 10% of the problem scenario are implemented and they work partly correct with syntax errors, do not follow the requirements **and** - The program uses no user-defined functions. All codes have been implemented as sequence block of code	- The software artefacts do not achieve any of the requirements of the problem scenario or the software has been implemented but they are not working, not following requirement, using wrong dataset, no correct results - No attention to rules and conventions to maximise code readability - None of the following skills are utilised correctly: loops, decisions, variable, user input, output

					and one line of coding inside and or the codes have been implemented as sequence block of code cell in Jupyter notebook		
	- Clear and structured comments are used throughout the code where appropriate	- Clear comments are used throughout the code where appropriate	- Comments are mostly present where appropriate and are mostly clear	- Comments are sparsely used in the code	- Comments are rare in the code	- Little or no comments are in the code or randomly inserted	
	- All appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- All appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- All appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- Most appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- Some appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- No appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	
	- All the following skills are utilised thoroughly and	- All the following skills are utilised efficiently:	- Most of the following skills are utilised efficiently:	- An adequate range of the following skills are utilised	- Some of the following skills are utilised correctly:	- Few of the following skills are utilised	

	efficiently: loops, decisions, variable, user input, output - All Required data that use to work with program are receive or specify by user	loops, decisions, variable, user input, output - All Required data that use to work with program are receive or specify by user	loops, decisions, variable, user input, output - Required data that use to work with program are majorly receive or specify by user	correctly: loops, decisions, variable, user input, output - Some of required data that works in a program are receive or specify by user rather than initial value in a code	loops, decisions, variable, user input, output - Most of required data that works in a program are initial value in the coding	correctly: loops, decisions, variable, user input, output - All required data that works in a program are initial value in a code, no input value from a user	
Utilise suitable tools to design, implement, test and evaluate solutions	- Extensive error and exception handling has been used in the code	- Well error and exception handling has been used in the code	- There is some evidence of error and exception handling	- There is some evidence of error handling but no exception handling	- There is a few evidence of error handling	- There is no evidence of error or exception handling	- No evidence of attempting required threshold

AE2 Demonstration (40%)

Learning Outcomes	UPPER FIRST A1 – A2 Exceed expectations in many aspects		FIRST A3 – A4 Substantially exceeds expectations		UPPER SECOND B1 – B3 (High) Meet learning outcomes and exceeds expectations in several aspects			LOWER SECOND C1 - C3 (Good) Meet learning outcomes and sometimes exceeds expectations			THIRD D1 - D3 (Competent) Meet learning outcomes			FAIL F1 – F3 (Incomplete/Poor) Fails to meet learning outcomes		
SOLENT GRADE	A1	A2	A3	A4	B1	B2	B3	C1	C2	C3	D1	D2	D3	F1	F2	F3
Design computer programs in a logical and structured way using appropriate techniques and principles	- All interfaces and user-interaction are well design, organised structured using function - Well display results, clear prompts and helpful error message presents when appropriate		- All interfaces and user-interaction are well design with some structured using function - Well display results and helpful error message presents when appropriate		- Most of the interfaces and user-interaction are good design using and good display results with clear prompts - Adequate helpful error message presents when appropriate			- Some of the interfaces and user-interaction are good design using and good display results with clear prompts - Display results in readable layout or format			- A simple user interaction message is used - Plain display results with simple layout or format - Some attempt coding use static data			- No user interface, no user interaction message has been implemented. All attempt coding use static data or initialise data		- No evidence of attemptin g required threshold
Develop computer programs aligned to appropriate programmin g standards and code conventions	- The program makes extensive use of functions (user-defined and built-in function) and modularise the code. The function and module work correctly as it designed and		- The program good utilises of functions (user-defined and built-in function) and modularise the code. The function and module work correctly as it designed and correct concept		- The program mostly uses functions (user-defined and/or built-in function) to implement the code. The function and module work correctly as it designed and correct concept of function and module and			- The program uses some user-defined functions to implement the code. The function and module work correctly as it designed and correct concept of function and module and			- The program uses a few user-defined functions or use lots of functions, but it doesn't work as it designs for/correct concept e.g., define function with some parameters but the code inside			- The program uses no user-defined functions. All codes have been implemented as sequence block of code		- The software artefacts do not achieve any of the requirem ents of the problem scenario or the software

	correct concept of function and module and - Clear and structured comments are used throughout the code where appropriate - All appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	of function and module and - Clear comments are used throughout the code where appropriate - All appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- Comments are mostly present where appropriate and are mostly clear - All appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- Comments are sparsely used in the code - Most appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	function doesn't use the parameter at all or try to use function but just have function name and one line of coding inside and or the codes have been implemented as sequence block of code cell in Jupyter notebook - Comments are rare in the code - Some appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions	- Little or no comments are in the code or randomly inserted - No appropriate rules and conventions followed to maximise code readability, including indentation, naming of variables /functions - Few of the following skills are utilised	has been implemented but they are not working, not following requirement, using wrong dataset, no correct results - No attention to rules and conventions to maximise code readability - None of the following skills are utilised correctly: loops, decisions, variable, user input, output

	- All the following skills are utilised thoroughly and efficiently: loops, decisions, variable, user input, output - All Required data that use to work with program are receive or specify by user - The own selection option extensively implemented with complicated or creative or interesting providing meaningful insight	- All the following skills are utilised efficiently: loops, decisions, variable, user input, output - All Required data that use to work with program are receive or specify by user - The own selection option implemented with well-select or creative or interesting providing meaningful insight	- Most of the following skills are utilised efficiently: loops, decisions, variable, user input, output - All Required data that use to work with program are receive or specify by user - The own selection option implemented with good selection to provide understandable/meaningful/ new insight	- An adequate range of the following skills are utilised correctly: loops, decisions, variable, user input, output - Some of required data that works in a program are receive or specify by user rather than initial value in a code - The own selection option implemented with basic selection/ similar with the set requirement. Not provide new understanding or insight	- Some of the following skills are utilised correctly: loops, decisions, variable, user input, output - Most of required data that works in a program are initial value in the coding or fix by a user - The own selection option implemented with basic selection/duplicate with the set requirements, provide basic information	correctly: loops, decisions, variable, user input, output - All required data that works in a program are initial value in a code, no input value from a user - The own selection option implemented with basic selection/duplicate with the fix requirements provide random information or incorrect information	
Demonstrate the application	- Provide comprehensive	- Well-explain structure of project	- Good explain structure of project	- Explain structure of project	- Presenting the code directly as sequence or very	- Presenting the code without overview	- Unclear explanation of

of key concepts	structure of project - Present the running program which shows bugs-free/no error. It always runs successfully with well display expected input and output **and** - Confidently discusses technically aspects of a program and provide the justification to implement or select data for own selection requirement	- Present the running program which shows bugs-free/no error. It is runs successfully with expected input and output in a well layout format **and** - Well discusses technically aspects of a program and provide the justification to implement or select data for own selection requirement	- Present the running program which shows bugs-free/no error. It can be run successfully with specific input and output result with a good layout format **and** - Detail discusses technically aspect of a program and provide the justification to implement or select data for own selection requirement	- Present the running program which may contain one or two error/bugs that affect some functionality. It can be run successfully with minor modifications **and** - Clear discusses technically aspect of a program and provide few justifications to implement or select data for own selection requirement	brief overview of project structure - Present the running program which run but contains some bugs that affect basic functionality in some running program **and** - Provide some discussion of technical aspect of a program without giving the justification of implementation for own selection requirement	structure of project - Present the running program which can run but it is not functional correctly or just only one tasks that works **and** - Unable to specify or explain technical aspect of a program and justifications	structure of project - Program cannot run - Unable to identify technical aspect of a program

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

Living CV

As part of the University's Work Ready, Future Ready strategy, you will be expected to build a professional, Living CV as you successfully engage and pass each module of your degree.

The Living CV outputs evidenced on completion of this assessment are:

1. I can solve real-world problems by getting and analysing large amounts of data
2. I can confidently write Python code to obtain, manipulate, and analyse real-world dataset
3. I am experienced in using environment tool such as Jupyter notebook to design, implement, test and evaluate solutions
4. I can conduct written and verbal presentations to share insights to audiences of varying levels of technical sophistication

Please add these to your CV via the Living CV builder platform on Solent Futures Online Solent Futures Online

Important Information

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 later 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 must be submitted by the deadline date; any Refer assessment submitted late will be regarded as a non-submission and will be awarded a zero.

Please find a link to the Late Submissions below:

Assessment regulations

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. If you are not 'fit to study', you can either request an extension to the submission deadline of 7 calendar days or you can 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 Handbook 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 alpha numeric grade scale for the marking of assessments. Unless you have been specifically informed otherwise your marked assignment will be awarded a letter/number grade. 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