I. Assessment Background/Scenario

The following brief is identical to the formative. Any work you have done for using it as part of the weekly productions or submitted for your formative CAN be used or repurposed in your summative. There is NO requirement for you to cite or reference your own work.

Client Brief

This document provides the client brief which should be used for the development of a single program or a collection of related programs, for submission with the final summative report.

You have been asked to design and develop a **prototype** application that demonstrates how data form the given data set can be formatted, cleaned, and used to generate specific outputs (as listed below).

Functional requirements

The application should provide the following functionality:

- A means to load the initial data set (which consists of three CSV files) and translate it into a suitable format, either XML, or JSON or an entity relationship structure (not CSV)
- A means to back up the data in this format using either files or a database. This should preserve the current state of the data when the program is closed, and make it available when the program is reopened.
- A process for cleaning and preparing the initial data set, managing inconsistences, errors, missing values and any specific changes required by the client (see below).
- A graphical user interface(s) for interacting with the data that enables the user to:
    - Load and clean an initial data set (from the CV format)
    - Load and save a prepared data set (from its translated format)
    - Use the prepared data set to generate output and visualisations
    - Manipulate the range of values used to generate output and visualisations

It should be assumed that this program will be able to handle other sets of data generated from the same source, i.e. data with the same column row headings but containing different values and anomalies. However, the application is **not** required to be generic (work with multiple unknown data sets). Given this best practice regarding code reuse, encapsulation and a well-defined programming interface should be applied where applicable.

Data manipulation and outputs

The client initially wants the application to perform the following actions on the data:

1. Outputs should not include any data from airports that have a **'type' 'closed'**
2. The **'type'** column contains information of the type of airport. Extract this out into a new column, one for each category of airport, for:
    a. all UK(GB) airports, that are , **large_airport**, **medium_airport**, **small airport**
    b. join each category, **large_airport**, **medium_airport**, **small airport**
    to the communication frequencies **' frequency_mhz'** that the airport uses for communication ensuring that each airport in all categories is correctly matched with its communication frequencies.
3. The client initially needs information to generate the following and output the results using appropriate representation:
    a. Produce the mean, mode and median for the **'frequency_mhz'**
        i. For **large_airport**
        ii. For frequencies **more than 100 mhz**
4. Produce a suitable graph that display the communication frequencies used by '**small_airport'** You may need to consider how you group this data to make visualisation feasible
5. Determine if there is any significant correlation between the communication frequencies used by the 3 different categories of airport. 'Are some frequencies used more than others?'. You will need to select an appropriate visualisation to demonstrate this.

Non-functional requirements

- The GUI interface provides appropriate feedback to confirm or deny a user's actions
- The application manages internal and user-generated errors

Technical requirements

- The application is built using Python 3.7.*
- The application uses one or more of the advanced APIs introduced on this module such as: NumPy, pandas, Seaborn, Matplotlib. It should NOT use alternative APIs for this functionality, however Python core libraries can be used to support where applicable, such as support for a database.
- The application runs within the anaconda environment using a Jupyter notebook
- The application or its parts do not run concurrently, do NOT use Python threads

The requirements specified here are the constraints within which you need to produce your development. They are not negotiable with the client.

II. Assessment Task(s)

Given the client brief above, produce a suitable software solution that meets the specified requirements as either a single program, or a series of clearly identifiable programs.

NOTE: Failure to submit the program(s) will result in a zero grade.

From this development produce a design report that addresses the series of focus questions given in the report questions document. These ask you to discuss your design decisions and demonstrate the underpinning theoretical aspects in the context of your software development. Code samples should be extracted from your software development, where requested, to demonstrate specific algorithms and interactions. You should support your discussion with appropriate reference to relevant sources using the correct citation and reference structure as indicated in the guide to IEEE referencing system.

Report contains 3 sections as follows:

The report consists of three sections, each containing specific questions targeting what needs to be addressed. In each section is a maximum word count and/or page numbers which should be adhered to. Your report should be labelled to clearly identify which question you are addressing. Your code samples and diagrams should be presented in the appendices and referred to (via caption labels) from your main discussion. The sections contribute to the module learning outcomes as follows:

Section 1: Theory supported by code samples (50%, 1400 words plus code samples)

Evidence for learning outcome: Demonstrate critical understanding of the theory and application of advanced programming techniques; Design and implement programs for real world problems.

Section 2: Design decisions supported by code samples (40%, 1200 words plus code samples)

Evidence for learning outcome: Communicate design decisions for the selection, storage and manipulation of data; Design and implement programs for real world problems.

Section 3: Reflection on the ethics, moral and legal aspects (10%, 400 words)

Evidence for learning outcome: Critically evaluate the legal and ethical impact of software developments within real world contexts.

Report references

Provide a correctly structured list of references to all the resources used for this development and report. Your responses should be appropriately supported by references to the literature and relevant resources using the Computer Science Department's referencing standard.

III. Deliverables

Your submission should contain **two files** only:

- A completed report answering the given questions as a single file in either .docx or .pdf format. This should **NOT** be included in the zipped file and should not exceed any specified word/page counts.
- A single zipped file containing your program or programs. If a database been used you should produce a file dump of the table structure to include here. This should NOT contain the original data set.

NOTE: Failure to submit the program(s) will result in a zero grade.

The report must adhere to the word count limits stated for EACH question and the page limit for the appendices.

Code samples:

All code samples submitted in your report should be extracted from your program verbatim. Those that are not will not contribute towards your grade. Code samples do **NOT** contribute to your word count, if they are presented in the appendices as requested. You should make sure they very clearly target what is requested, and submitting general or broad ranges of code may be subject to penalties. For example, if you were asked to demonstrate where a closure has been used within your code, you would not be demonstrating this by submitting a complete program that contains a

closure. You may add code style comments to your code samples to help target what you are showing but should not change the structure of the code from your software submission.

Using a database:

Where you have opted to use an SQL or relational database (other than Mongo) include after your list of references the following:

- Name of database and link to download (install package)
- Version number of the database used
- The name of the code file (Python class/Jupyter Notebook) that creates and populates the database
- The point in your code where local host and the port are set (make this clear)

You should make sure that your submitted code contains all the code required to setup and populate your database via a local host connection.

General submission guidelines:

You should refer to the assignment guidelines in Canvas for the general format of your report. You should not exceed any provided word counts (per question) or page counts for appendices. These are clearly stated on the report questions document.

IV. Marking Criteria

The following defines the marking criteria for each section/question and identifies which learning outcomes it contributes to.

The York Computer Science Department			
MLO	Section	**Criteria**	**Marks**
2	Upload code and design documentation	**Pass/fail**	**0**
Section 1. Theory supported by code samples			
1,2	a. Adaptation to a concurrent model	Appropriate concurrent mechanisms have been selected and there is a logical approach to the passing of data given the program part selected.	**10**

		The design is discussed in terms of the advantages and disadvantages to the client brief and the developed program.	**10**
1,2	c. Implementing user interaction	The selected constructs are appropriate given the required interactions. There is a clear rationale for their selection given best practice in GUI constructs and layout.	**10**
		Appropriate code samples and designs are given that demonstrate the overall effectiveness of the user interaction given the client's requirements.	**10**
1,2	d. Evaluating high level languages	There is a rational argument, either for or against, Java or Python being more effective in terms of 'manipulation of data containers', that is supported by a critical evaluation of each language's approach to using these constructs, which is supported by appropriate referencing	**10**
Section 2. Design decisions supported by code			
2,3	a. Data format	An effective format has been selected and a rational argument is presented for how it supports the nature of the data and the type of analysis required to produce the client's information needs.	**10**
2,3	b. Code constructs	Appropriate code constructs and internal data structures have been selected and applied to achieve the client's requirements. Considerations have been made for any anomalies within the data set.	**10**
2,3	c. Data visualisation	Appropriate and effective libraries/functions have been selected and their application for presenting the required output is logical. Considerations have been made for any anomalies within the data set.	**10**

2,3	d. Data analysis	Appropriate and effective libraries/functions have been selected and their application for preparing the data is logical. Considerations have been made for any anomalies within the data set.	**10**
Section 3. Reflection on ethics, morals and legal aspects			
4	a. Reflection	There are clear and appropriate examples, which are effectively used to support either a 'for' or 'against' position on the statement.	**10**
		TOTAL	**100**

NOTE: Failure to submit the program(s) will result in a zero grade.