COM00142M

Department of Computer Science

ADVANCED PROGRAMMING

# SUMMATIVE ASSESSMENT BRIEF

| | |
|---|---|
| **Author** | Dr Dawn Wood |
| **Assessment Type** | Summative |
| **Release** | Week 2 |
| **Submission** | Monday 11th January 2021 by 13.00pm |
| **Feedback** | Within 20 working days of submission |
| **Weighting** | 100% |

## I. Module Learning Outcomes

1. Demonstrate a critical understanding of the theory and application of advanced programming techniques
2. Design and implement programs for real-world problems
3. Communicate design decisions for the selection, storage and manipulation of data
4. Critically evaluate the legal and ethical impact of software developments in real-world contexts.

This assessment will contribute to all the learning outcomes for this module.

## II. Assessment Background/Scenario

### 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. It may be used for the development of the formative productions. If it is, these should NOT be shared with your peers for feedback. The summative work MUST be all your own work.

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.
- 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 enable the user to:
    - Load and clean the initial data set
    - Load and save the prepared data set
    - 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 vendors that have a ‘PROGRAM STATUS’ of INACTIVE.

2. The 'PE DESCRIPTION' column contains information on the number and type of seating available at the vendor. Extract this out into a new column, retain all other information within that column. E.g.:
    a. 'FOOD MKT RETAIL (1-1,999 SF) LOW RISK',
    b. 'RESTAURANT (61-150) SEATS LOW RISK'.
    c. Extract the greyed area out and retain the rest in the examples
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 **inspection score per year**:
        i. For each type of vendor's seating
        ii. For each 'zip code'
4. Produce a suitable graph that displays the number of establishments that have committed each type of violation. You may need to consider how you group this data to make visualisation feasible
5. Determine if there is any significant correlation between the number of violations committed per vendor and their zip code, 'Is there a tendency for facilities in specific locations to have more violations?'. 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.* and above
- The application uses advanced APIs such as: NumPy, panda, Seaborn, Matplotlib
- 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

# III. 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 structured design report (using the given template) that addresses a series of focus questions. 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 this should be adhered to. Your report should clear identify which question you are addressing. Ensure you keep your code samples separate from your explanations. 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.

# IV. Deliverables

You should make two file submissions as follows:

- 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. 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:

- Present your answers on A4 pages, with a minimum 12 point font (14 point for headings), minimum 120% line spacing (what Word calls "Multiple 1.08"), and margins of at least 2cm on either side.
- All images must be of a size/quality that the details are viewable within the document. Images will not be scaled to identify blurred or small text.
- Your code should be focused on the selected production and not the whole program, and be easily identifiable as a solution to the production problem. Well commented and structured code is more likely to be easy to follow and receive feedback.

# V. Marking Criteria

Students should be able to use this as a guide to the elements they need to cover, without it giving them an answer to the task/question. Where academic quality is required there should be a separate and clear criterion for this (e.g. 10%). The criteria should be clearly linked to the module level learning outcomes (Via Number). This can be at the task or element level.

| 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, which are discussed in terms of the advantages and disadvantages to the client brief and the developed program. | **20** |
| **1,2** | | The selected constructs are appropriate given the required interactions. There is a clear | **10** |

| | | | |
|---|---|---|---|
| | c. Implementing user interaction | rationale for their selection given best practice in GUI constructs and layout. | |
| | | 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 'speed of development', that is supported by a critical evaluation of each language's approach and 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, drawn from the discussion forums, 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.

# VI. Grading

The pass mark for postgraduate modules is 50%. For more information about grades and assessment criteria, please review the ‘Assessment and award’ section of the York Online Handbook which can be downloaded from the **Orientation Module**.

# VII. Assessment submission

You will submit your assessments in the ‘Assignments’ area of the module in Canvas.

Please check your canvas module for the specific submission date for this assignment.

For general assessment guidelines consult your Canvas Module, Orientation Module and for Academic Regulations the University of York’s website.

Any queries regarding the details of your assessment should be directed to your module tutor.

Any queries regarding assessment procedures should be directed to studentsuccess@online.york.ac.uk.