IN-COURSE ASSESSMENT (ICA) SPECIFICATION

Module Title: **Software for Digital Innovation**	Module Leader: **Dr Yar Muhammad**
	Module Code: **CIS4044-N**
Assignment Titles: **Part 2: Data Processing and Visualisation [60%]**	Deadline Date: **Wednesday 12th January 2022**
	Deadline Time: **4:00pm**
	Submission Method: Online (Blackboard) ☑ Middlesbrough Tower ☐

Online Submission Notes:

1. Please follow carefully the instructions given on the Assignment Specification
2. When Extenuating Circumstances (e.g. extension) has been granted, a fully completed and signed Extenuating Circumstances form must be submitted to the School Reception or emailed to scedt-assessments@tees.ac.uk.

FULL DETAILS OF THE ASSIGNMENT ARE ATTACHED INCLUDING MARKING & GRADING CRITERIA

Software for Digital Innovation (CIS1028-N)
Assignment Specification

Contents

Introduction

This document describes the second element of assessment for the module. It is an individual in-course assignment (ICA).

This ICA assesses the following learning outcomes (from the Module Guide):

1. Apply problem solving skills in a formal structured environment to meet a specification.

3. Critically examine relevant software tools and evaluate their appropriateness and limitations in a particular scenario.
4. Apply effectively existing software libraries and utilise appropriate software tools to real world applications.

5. Critically evaluate the security and other risk implications related to the utilisation of software libraries and tools to real world applications in the context of Data Science (DS), Artificial Intelligence (AI), or Financial Technology (FinTech).

Assessment in Brief (60% of Module)

You have been tasked by the local authority to produce a system to help speed up Freedom of Information requests. You have two key programming objectives:

1. Process and visualise Covid-19 confirmed cases from a government provided source file, see *§Instructions for COVID-19 Confirmed Cases* for more details.
2. Process and visualise Stop and Search data from data supplied publicly over the internet by the Police, see *§Instructions for Stop and Search* for more details.

You should provide the end user with a single starting point to run your separate Python modules. This could be a console menu system, or a Graphical User Interface (GUI).

You must provide evidence of non-functional testing through a combination of black-box and unit testing. You will submit your final program(s) to Blackboard, along with a README outlining any installation or running instructions.

The final part of the assessment asks you to write a report that examines the relevant software tools used to complete the assessment, and for you to critically evaluate the security and other risk implications related to the utilisation of software libraries and tools to real world applications, see *§Instructions for Report* for more details.

See the marking criteria for a detailed breakdown of the marks available. Read the individual instruction sections carefully for additional guidance.

Instructions for COVID-19 Confirmed Cases

Before attempting this ICA ensure you have completed the portfolio work. This will ensure you have all the basic *Python* programming skills needed to develop a viable solution.

Download **`specimenDate_ageDemographic-unstacked.csv`** from Blackboard. All COVID-19 data processing and visualisation must be done against this file. The file itself has been sourced from https://coronavirus.data.gov.uk/details/about-data#cases-by-age and is used as part of the UK Governments own reporting.

The file contains daily number of cases data aggregated by age into 0-59, 60 plus and individual five-year bands. 7 day rolling averages and rates are also included where the data is presented by specimen date (the date the sample was collected from the patient).

Begin processing this data by identifying the unique areas within the file. *As a minimum* you should create new columns that show the percentage change of infection rates between each day for each area (without relying on the rolling averages supplied in the source file). There are opportunities for further processing for you to do, feel free to discuss this with your tutor.

Visualise this data using Matplotlib (see https://matplotlib.org/). There are many more visualisation charts available than what was covered in the taught material. Explore the documentation and pick ones that are suitable.

You are expected to produce **multiple visualisations** of the data (you can use https://coronavirus.data.gov.uk/details/cases for inspiration). Some *possible* visualisations you could produce:

- Total number of cases reported each day over a given date range.
- Total number of cases reported each week over a given date range.
- Total number of cases reported each month over from the start of reporting to end.
- Areas with the highest number of cases on a given day.
- Areas with the largest positive change of cases for a given 7 day period.

- Comparison of two or more areas concerning their daily change over a given date range.
- Comparison of two or more areas concerning their cumulative cases total over a given date range.
- And so on.

Ensure you provide details of each chart in a README file in the root directory of your source code.

Apply good programming practice throughout. Ensure common functions are in their own Python module and can be accessed by both the COVID-19 Confirmed Cases and Stop and Search modules.

Provide evidence of non-functional testing through a combination of black-box and unit testing for any code developed for this programming objective.

Instructions for Stop and Search

https://data.police.uk/ is the site for open data about crime and policing in England, Wales, and Northern Ireland. A publicly available API provides detailed crime data and information about individual police forces and neighbourhood teams. The documentation for this API can be found at https://data.police.uk/docs/

It is possible to list out stop and searches by force, full details on this method can be found at https://data.police.uk/docs/method/stops-force/

Using Python retrieve data from the API or without using API, process it, and then visualise using Matplotlib. You are offered more freedom when it comes to processing and visualising data on this task. Think about what information someone may ask regarding stop and search.

Some *hypothetical* questions you may want to answer:

- How many teenagers were stopped and searched by Cleveland Police in the month of July 2020?
- What is the breakdown of age ranges for stop and search for a given police force each month?
- Has there been a reduction in stop and search during the first UK COVID-19 lockdown?

Ensure you provide details of each chart in a README file in the root directory of your source code.

Apply good programming practice throughout. Ensure common functions are in their own Python module and can be accessed by both the COVID-19 Confirmed Cases and Stop and Search modules.

Provide evidence of non-functional testing through a combination of black-box and unit testing for any code developed for this programming objective.

Instructions for Report

You are to write a short report that:

- critically examines the relevant software tools used for this assessment (including the portfolios) and evaluate their appropriateness and limitations,

- critically evaluates the security and other risk implications related to the utilisation of software libraries and tools to real world applications in the context of Data Science (DS), Artificial Intelligence (AI), or Financial Technology (FinTech).

This is an individual report, so what you submit must be your own work and not that created in collaboration with others. The document should be approximately **1500 words**.

Software tools can refer to Integrated Development Environments (IDEs), other software packages, and Python packages/modules outside of the standard library.

The report requires independent research (particularly when investigating security and other risk implications). You should provide evidence of research through appropriate references. All references should be in the Harvard style following Cite Them Right Online.

Organise the report into suitable sections.

Read the marking criteria below and ensure that your final piece of work addresses all the criterion statements.

Deliverables

You must submit your work as a ZIP file with the filename `student_number_surname_forename.zip` (i.e. `u0029122_Min_Mart.zip`) via *Blackboard* with the following directory structure:

- ***source***: this directory should contain the entire source code of your solution. Include any necessary project configuration files and third-party libraries. Include a README file containing any special instructions for installation, running or unit testing.
- ***documentation***: this directory should contain your black-box testing document and your report.

Unless otherwise stated all documentation must be submitted in .docx or .pdf format.

Marking Criteria

Criterion	Excellent (70-100%)	Very Good (60-75%)	Good (50-65%)	Satisfactory (40-55%)	Fail (0-40%)
Python Solution structure [15]	Sensible scoping throughout, with little wasted code that fully adheres to PEP 8.	Sensible scoping throughout, with little wasted code, minor deviations from PEP 8.	Appropriate Python modules imported, variables created succinctly with few deviations from PEP 8.	Few Python modules imported, variables haphazardly created with many deviations from PEP 8.	Code is not in a runnable state. Contains syntax errors, or unresolved runtime errors.
	Code is excellently documented.	An attempt has been made to document the code using Docstring.	Appropriate code comments are spread through the solution.	Code comments are sparse.	Few useful comments in code.
	The solution throws no uncaught exceptions.	Exception handling covers exceptions from imported packages, file verification, user input, and type casting.	Exception handling is present but limited to file verification and type checking.	Exception handling is very limited.	Exception handling is not present.
	The code structure is split logically into separate files/modules to facilitate code re-use and allows for easier unit testing.	The code structure is split logically into separate files/modules to facilitate code re-use.	Some attempt has been made to move common functions into their own files/modules to facilitate code re-use, but code duplication remains.	The COVID-19 confirmed cases, and stop and search solutions contain large amounts of working duplicate code.	Some code present but taken directly from taught materials. Little in the way of further independent development.
	The solution can be driven from a GUI, without the end user having to type any Python.	The solution can be driven from a single starting point through an interactive shell menu. An attempt has been made to use a GUI.	The solution is split across several python modules that must be run individually. An attempt has been made to use a single starting point (interactive console menu).	The solution is split across several python modules that must be run individually.	Solution does not meet either programming objectives.
	A README in the root directory that shows the author, how to run the program(s), provides details if the command line arguments, includes dependency information, and how to run the unit tests.	A README in the root directory that shows the author, how to run the program(s), includes dependency information, and how to run the unit tests.	A README in the root directory that shows the author, how to run the program(s), and includes dependency information.	A README in the root directory that shows the author and how to run the program(s).	A README is not present.

COVID-19 Confirmed Cases [10]	Data processing shows evidence of further statistical analysis through grouping and aggregation.	Solution shows evidence of further data processing beyond the suggestions in the brief to identify trends, patterns, or draw further comparison i.e. between areas, dates, age ranges etc. Results of data processing can be output to separate csv files.	Use of pandas demonstrates selection of subsets, filtering, selection of specific rows and columns. Input parameters can be manipulated at run time (i.e. area names, date ranges).	Data can be loaded into Python using pandas. Basic filtering on show. All instance variables (i.e. area names, date ranges) are hard coded, with little customisation available to the end user.	Unable to process confirmed cases data (supplied in csv format).
	An excellent set of visualisations identifying trends, patterns, comparisons between multiple regions over diverse timescales.	Solution contains visualisations that are not included in the taught materials, showing evidence of further independent research.	Simple visualisations that show some attempt at customisation beyond the taught materials. An attempt has been made to show comparative data between regions on the same plot. All charts are well labelled and appropriately titled.	Simple visualisations that do not go beyond the taught material. Missing some labels and titles. Data parameters are fixed and cannot be altered by the end user.	Unable to visualise COVID-19 confirmed cases data.
Stop and Search [15]	Data can be cached (saved) after retrieval in event of network failure. Methods provided to manage or invalidate this cache. Demonstrates experimentation and processing with other methods provided by the API.	Data can be retrieved by Python using an appropriate package. Request parameters can be manipulated at runtime. The availability method within the API can be checked to return valid force and date choices.	Data can be retrieved by Python using an appropriate package. Request parameters can be manipulated at runtime (choice of force and date).	Data can be retrieved by Python using an appropriate package. Request is hardcoded.	Unable to retrieve stop and search data from the API.
	Data processing shows evidence of further statistical analysis through groupings and aggregations.	Code shows evidence of further data processing beyond the suggestions in the brief to identify trends, patterns, or draw further comparison between areas, dates, age ranges etc. Results of data processing can be output to separate csv files.	Demonstrates filtering and selection of specific items. Input parameters can be manipulated at run time (i.e. age range, gender, outcome etc.).	Data can be processed using Python. Basic filtering on show. All instance variables (i.e. age range, gender, outcome etc.) are hard coded, with little customisation available to the end user.	Unable to process stop and search data (retrieval from the API).

	An excellent set of visualisations identifying trends, patterns, comparisons between multiple forces and multiple data points.	Submission presents plots that are not included in the taught materials, showing evidence of further independent research.	Simple visualisations that show some attempt at customisation beyond the taught materials. An attempt has been made to show comparative data between age, outcome, or data across forces on the same plots. All plots are well labelled and appropriately titled.	Simple visualisations that do not go beyond the taught material. Missing some labels and titles. Data parameters are fixed and cannot be altered.	Unable to visualise stop and search data.
Non-functional requirements testing [10]	An extensive black-box test plan that covers all aspects of the system. Extensive unit tests that cover all operations within the system.	A comprehensive black-box test plan that covers most test cases including success, failure, and range. Unit testing demonstrated sufficiently. Gaps do remain for some operations.	An acceptable black-box test plan showing individual test cases, but it is not sufficiently extensive. An attempt at unit testing has been made, but it is not entirely suitable.	A limited black-box test plan is evident showing individual test cases, but it is poorly formatted and/or contains few tests. No attempt at unit testing.	No evidence of testing provided.
Report [10]	Report critically examines the relevant software tools used for this assessment and evaluates their appropriateness and limitations. Report critically evaluates the security and other risk implications related to the utilisation of software libraries and tools to real world applications	Report describes the key software tools used for the assessment, evaluation includes facts and opinions of others with appropriate citations. Report evaluates the security and other risk implications related to the utilisation of software libraries and tools to real world applications, evaluation includes facts and opinions of others with appropriate citations.	Report describes the key software tools used for the assessment; evaluation is limited to self-experience. Report addresses security and other risk implications related to the utilisation of software libraries and tools in real world applications. Evidence is mostly limited to hypothetical situations.	Report describes some software tools used for the assessment. Limited evaluation present. Report addresses some security and other risk implications related to the utilisation of software libraries and tools real world applications. Limited evaluation present.	Report does not examine the relevant software tools used for the assessment. Report does not address the security and other risk implications related to the utilisation of software libraries and tools in real world applications.

	Report is very well presented, absent of proofing errors, referencing is in expected format without error, word count +/- 10%.	Report is well presented, very few proofing mistakes, few deviations in referencing format, word count +/- 10%.	Report is reasonably presented, minor proofing mistakes, minor deviations in referencing format, word count +/- 15%.	Report is reasonably presented, does contain some proofing mistakes, deviations in referencing format word count +/- 20%.	Report is poorly formatted, with little meaningful content, few references of worth and not in expected format, Word count +/- 30%.

Document History

Revision 0 (25-Sept-21) Initial version of the 2021/22 assessment specification.