

**MODULAR PROGRAMME**

**COURSEWORK ASSESSMENT SPECIFICATION**

**Module Details**

|  |  |  |
| --- | --- | --- |
| **Module Code** UFCFVQ-15-M | **Run** 20JAN/1 | **Module Title**  Programming for Data Science |
| **Module Leader**  Dave Wyatt | **Module Coordinator** | **Module Tutors** Dave Wyatt; Corey Ford; Paul Hewson; Jan Van Lent; Jacob Baker |
| **Component and Element Number**  B: CW1 | | **Weighting: (% of the Module's assessment)**  **75%** |
| **Element Description** PRACTICAL SKILLS ASSESSMENT | | **Total Assignment time** 30 hours |

**Dates**

|  |  |
| --- | --- |
| **Date Issued to Students**  10th March 2021 | **Date to be Returned to Students**  27th May 2021 |
| **Submission Place**  **GitLab submission** | **Submission Date**  29th April 2021 |
| **Submission Time** **2.00 pm** |

**Deliverables**

|  |
| --- |
| This assessed work has three separate deliverables:   * **Git Repository Link**, to be submitted via [GitLab Repository Link Submissions Point](https://forms.office.com/Pages/ResponsePage.aspx?id=CBLvBzxBXkuc3WTvMFdU8DZJChtYGhdHiTYJUz3c6elUNzdHM1RTMkZYNlExREs3VTNKOUZCRTdSNC4u) * **Two Programming Tasks**, to be completed in increments as instructed in the assignment specification and submitted via GitLab |

**Module Leader Signature**

|  |
| --- |
| DAVE WYATT |

Programming for Data Science -   
Assignment Specification

Portfolio of Programming Exercises and Development Process Report

# Information

This single assignment for this module will run during the Spring Term, and involves three separate components:

|  |  |  |
| --- | --- | --- |
| Component | Release Date | Assignment % |
| Git Repository Link | March 10th | 10 |
| Programming Task 1 | March 10th | 45 |
| Programming Task 2 | March 24th | 45 |
| Total |  | **100** |

Details for each of the above assignment components are provided below.

# Git Repository Link

* A GitLab repository should be created using the [gitlab.uwe.ac.uk](https://gitlab.uwe.ac.uk/) system according to the naming convention detailed in A.1 below
* The repository should be used to commit all coursework development work.
* The initial repository commit should be an appropriately edited README.md file (including a student ID, student name and MSc programme information)
* NOTE: any commits to this repository that occur after the coursework deadline will be treated as LATE.

## A.1. Requirements

* The repository should use the following naming convention: ***00973398***\_***wyatt***\_fvq\_2021, where the first 8-digit number (***00973398*** in this example) is the student’s ID and the second part (***wyatt*** in this example) is the student’s surname.

## A.2. Deliverables

* The repository URL is the main deliverable for this part of the coursework, e.g., [***https://gitlab.uwe.ac.uk/di2-wyatt/00973398\_wyatt\_fvq\_2021***](https://gitlab.uwe.ac.uk/di2-wyatt/00973398_wyatt_fvq_2021)

## A.3. Submission

* The Repository URL should be submitted at the following link: [GitLab Repository Link Submissions Point](https://forms.office.com/Pages/ResponsePage.aspx?id=CBLvBzxBXkuc3WTvMFdU8DZJChtYGhdHiTYJUz3c6elUNzdHM1RTMkZYNlExREs3VTNKOUZCRTdSNC4u)

## A.4. Grading Criteria

* Marks are allocated as follows:
  + 5 marks will be awarded to any student who has submitted an appropriate GitLab Repository URL using the link provided in A.3. above
  + Up to 5 marks for editing the README.md file and committing it with an appropriate commit message

# B. Programming Task 1

* This programming task focuses on using Python to calculate a set of standard statistical measures for a given dataset using built-in functions and data structures ONLY.
  + You are **NOT** permitted to use any Python library functions to meet the requirements given in B.1. below, such as those provided by Pandas, NumPy or SciPy.
* There are two data files available for use in this task
  + The first data file is called ***task1.dat***. Each line in the data file provides a single data value. This file available in the Assignment section on Blackboard
  + The second data file is called ***task1.csv***. This file is a CSV file which includes a header row with multiple named data values. The first column of data matches the data stored in the ***task1.dat*** file. This file available in the Assignment section on Blackboard
* Students are expected to follow appropriate coding standards (including code commenting, consistent variable/function naming, clear code structure, and appropriate use of data structures).
* One useful strategy that can help a programmer to identify the strengths/weaknesses in their approach is to reflect on the development process they followed. For each programming task a reflective report should be submitted which focuses on the process taken to develop a solution to the task. The report should:
  + include an explanation of how they approached the task
  + include any pseudo code or other algorithmic aid used to help complete the task
  + identify the strengths/weaknesses of the approach used
  + consider the approach used could be improved
  + suggest alternative approaches that could have been taken

## B.1. Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Priority |
| FR1.1 | Find the arithmetic mean | This requirement should assume the data to be processed is stored in a Python List | ***MUST*** |
| FR1.2 | Find the standard deviation | This requirement should assume the data to be processed is stored in a Python List | ***MUST*** |
| FR1.3 | Find the min/max values | This requirement should assume the data to be processed is stored in a Python List | ***MUST*** |
| FR1.4 | Find the 25th percentile | This requirement should assume the data to be processed is stored in a Python List | ***MUST*** |
| FR1.5 | Find the 50th percentile | This requirement should assume the data to be processed is stored in a Python List | ***MUST*** |
| FR1.6 | Find the 75th percentile | This requirement should assume the data to be processed is stored in a Python List | ***MUST*** |
| FR1.7 | Print the set of summary statistics in tabular form | The summary statistics should be displayed in the following format:  count ……..  mean ……..  std ……..  min ……..  25% ……..  50% ……..  75% ……..  Max …….. | ***MUST*** |
| FR1.8 | Read data from a file into memory | The ***task1.dat*** data file contains a single data value per row. All data values should be read into memory and placed in a List data structure | ***MUST*** |
| FR1.9 | Design and build a data structure to store CSV data in memory | The data structure must be capable of storing multiple data values per CSV row and custom named data values | ***SHOULD*** |
| FR1.10 | Create a function to read CSV data from a file into memory | The ***task1.csv*** data file contains a multiple data values per row. This function should read the first row of the CSV file and use this to name each data value in the data source before ingesting the remaining rows into the data structure built in FR1.9 | ***SHOULD*** |
| FR1.11 | Create a generic statistical summary function | This function should be able to calculate all the statistics listed in FR1.1-FR1.6 for the data read into memory in FR1.10 | ***COULD*** |
| FR1.12 | Create a function to print a custom table | This function should be able to output all the statistics generated in FR1.11. | ***COULD*** |

## B.2. Deliverables

* There is single deliverable for this task:
  + A Jupyter Notebook file (in (.ipynb format) containing a complete solution to Programming Task 1 (based on the template provided[[1]](#footnote-1)).
  + The Jupyter Notebook should also include a Development Process Report written using Markdown reflecting on the process taken to develop a solution to this task
    - The report should not exceed 500 words.

## B.3. Submission

* The Jupyter Notebook file (incl. Development Process report) should be submitted electronically via the student’s coursework GitLab repository.

## B.4. Grading Criteria

* Marks are allocated as follows:
  + up to 30 marks for the Python code solution
    - Marks will be awarded for each completed requirement **according to the level of completion** (**0%**, **25%, 50%**, **75%** or **100%**).
      * Each *MUST* requirement is worth up to 2 marks
      * Each *SHOULD* requirement is worth up to 3 marks
      * Each *COULD* requirement is worth up to 4 marks
  + up to 5 marks for adherence to good coding standards.
  + up to 10 marks for the Development Process Report
    - Marks will be awarded for appropriate use of technical language, critical reflection on development process and quality of engagement with the reflective process

# C. Programming Task 2

* This programming task focuses on using NumPy, Pandas and Matplotlib/Seaborn to combine, clean and analyse two datasets related to student performance.
* Two data files have been provided for this task. These data files provide some real data from the Open University.
  + The ***task2a.csv*** data file contains background information about 26746 students including gender, age, disability status and score.
  + The ***task2b.csv*** data file contains information about the number of click events made by 26074 students using the University’s Virtual Learning Environment (VLE) system.
  + The files are available in the Assignment section on Blackboard
* Students are expected to follow appropriate coding standards (including code commenting, consistent variable/function naming, clear code structure, and appropriate use of data structures).
* For this programming task a reflective report should be submitted which focuses on the process taken to develop a solution to the task. The report should:
  + include an explanation of how they approached the task
  + include any pseudo code or other algorithmic aid used to help complete the task
  + identify the strengths/weaknesses of the approach used
  + consider the approach used could be improved
  + suggest alternative approaches that could have been taken

## C.1. Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Description | Priority |
| FR2.1 | Read CSV data from a file (with a header row) into memory | The ***task2a.csv*** data file contains 26746 rows of data and includes a header row. All data should be read into a DataFrame object. | ***MUST*** |
| FR2.2 | Read CSV data from a file (without a header row) into memory | The ***task2b.csv*** data file does not include a header row. The following column names should be used: ***id\_student***, ***click\_events***. All data should be read into a DataFrame object. | ***MUST*** |
| FR2.3 | Merge the data from two Dataframes | The two DataFrame objects created in FR2.1 and FR2.2 should be merged into a new DataFrame | ***MUST*** |
| FR2.4 | Remove any rows that contain missing values | Remove all rows from the DataFrame that contain a missing value in any column. | ***MUST*** |
| FR2.5 | Filter out unnecessary rows | Remove all rows from the DataFrame where ***click\_event*** is smaller than ***10***. | ***MUST*** |
| FR2.6 | Rename the ***score*** column | Rename the ***score*** column to ***final\_mark*** | ***MUST*** |
| FR2.7 | Remove unnecessary column(s) | The following columns are no longer needed: ***region***, ***final\_result*** and ***highest\_education***. Remove them from the DataFrame. | ***MUST*** |
| FR2.8 | Write the DataFrame data to a CSV file | Once FR2.1-FR2.7 are satisfied, store the resulting DataFrame into a CSV file called ***update.csv***. NOTE: column headers should be included. | ***MUST*** |
| FR2.9 | Investigate the effects of age-group on attainment and engagement | Using an appropriate Pandas function, find the average engagement and final mark for each age grouping | ***SHOULD*** |
| FR2.10 | Present the results of the age-group investigation using an appropriate visualisation | Using an appropriate visualisation tool (such as Matplotlib or Seaborn) show the results of the investigation from FR2.9 | ***SHOULD*** |
| FR2.11 | Investigate the effects of engagement on attainment | Use an appropriate visualisation tool to investigate if there is any correlation between engagement and final mark | ***COULD*** |
| FR2.12 | Test the hypothesis that there is a significant effect on attainment | Using NumPy’s stats module, test if any effect identified in FR2.11 is statistically significant | ***COULD*** |

## C.2. Deliverables

* There is single deliverable for this task:
  + A Jupyter Notebook file (in (.ipynb format) containing a complete solution to Programming Task 2 (based on the template provided[[2]](#footnote-2)).
  + The Jupyter Notebook should also include a Development Process Report written using Markdown reflecting on the process taken to develop a solution to this task
    - The report should not exceed 500 words.

## C.3. Submission

* The Jupyter Notebook file (incl. Development Process report) should be submitted electronically via the student’s coursework GitLab repository.

## C.4. Grading Criteria

* Marks are allocated as follows:
  + up to 30 marks for the Python code solution
    - Marks will be awarded for each completed requirement **according to the level of completion** (**0%**, **25%, 50%**, **75%** or **100%**).
      * Each *MUST* requirement is worth up to 2 marks
      * Each *SHOULD* requirement is worth up to 3 marks
      * Each *COULD* requirement is worth up to 4 marks
  + up to 5 marks for adherence to good coding standards.
  + up to 10 marks for the Development Process Report
    - Marks will be awarded for appropriate use of technical language, critical reflection on development process and quality of engagement with the reflective process
* All submissions MUST represent a student’s own work.
  + Students are NOT permitted to collude or submit anyone else’s work as their own (including any partial solutions found on the web)
  + An analysis of submissions will be made across the cohort to identify any evidence of collusion and/or plagiarism during the marking phase
  + The following link provides more information about assessment offences and any associated penalties: <https://www.uwe.ac.uk/study/academic-information/assessments/assessment-offences>.

1. There is a Jupyter Notebook template available in the Assignment folder on Blackboard - ***UFCFVQ-15-M Programming Task 1 Template.ipynb*** [↑](#footnote-ref-1)
2. There is a Jupyter Notebook template available in the Assignment folder on Blackboard - ***UFCFVQ-15-M Programming Task 2 Template.ipynb*** [↑](#footnote-ref-2)