|  |
| --- |
| Dartmouth |
| Systems Development Continuous Assessment 2 |
| [Document subtitle] |

|  |
| --- |
| [Author name]  [Date] |

# Requirements summary

|  |  |  |
| --- | --- | --- |
| **No.** | **Requirement Type** | **Description** |
| 1 | Non-functional | Must be made of multiple python modules of our own design. |
| 2 | Non-functional | Modules and their functions should be tested independently. |
| 3 | Non-functional | The program should only use libraries from the Python 3.8 standard library |
| 4 | Functional | Must use a command-line interface to prompt the user for input parameters |
| 5 | Functional | Must use a command-line interface to display results |
| 6 | Functional | Must use a command-line interface to provide user instructions |
| 7 | Functional | Commands which produce many lines of output must prompt the user for a file name to write and save the output to |
| 8 | Functional | When your program prompts for user input it must always accept the options of “Quit” and “Restart” |
| 9 | Functional | The program must allow the user to retrieve the centre coordinate (latitude or longitude) of a postcode (only for EX postcodes) using a command |
| 10 | Functional | The program must allow the user to retrieve all reported street level crimes within a radius of 1 km, 2 or 5 km of the centre coordinate using a command |
| 11 | Functional | The program must allow the user to sort the resulting data of a command by distance from the postcode centre, by date (most recent first) and by crime category |
| 12 | Functional | The program must be able to produce a tabular report of street level crimes in CSV format |
| 13 | Non-functional | Any CSV files produced by the program must be suitable for use by a spreadsheet program |
| 14 | Functional | The program must be able to save a report with a user specified file name |
| 15 | Non-functional | The program is easy to use. |
| 16 | Functional | The program provides the user with appropriate guidance as to how it is meant to be used. |

# Work breakdown

|  |
| --- |
| **Description of work item** |
| A python module which is script written in the event driven programming style. It will allow the user to give input commands and the program will carry out functions depending on the commands (and their arguments) entered by the user and give the appropriate response; all in a command line interface. There will be commands to: quit the program, get help, retrieve street level crime data (as per requirement 10), sort the resulting data (as per requirement 11), save resulting data to a CSV file named by the user (as per requirement 12). Each command will provide the user with an entry point to the functionality of the other modules. Addresses requirements 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 16 |
| A python module which provides functions to filter data. Addresses requirements 1, 2, 3, 10 |
| A python module which provides functions to sort data by different values. Addresses requirements 1, 2, 3, 11 |
| A python module which provides a programming interface which retrieves csv data. Addresses requirements 1, 2, 3, 9 |
| A python module that provides functions to save output data to a csv file, with or without an output to the python console, with a name specified by the user. Addresses requirements 1, 2, 3, 7, 12, 13, 14 |

# Test Report

# Program Instructions

This is a guide to configuring and using your crime data report generator.

If you are using the example data just as it was provided to the students in the exact same folder structure, please create a folder called ‘data’ in the same directory that your “crime\_data.py” file is in and put the “Devon\_and\_Cornwall\_crime\_data\_2019” and “Devon\_postcodes” folders in the “data” folder. If the program is failing to load the csv files because of different folder or file names then read the more in depth guide on configuration in the file “readme.txt”.