Software Testing Report

Victoria Accident Data Visualisation Project

### Max Woollons s5349356

### Reiss Tooze s5046939

### Kail Terepai s5307474

**Table of Contents**

[1.0 Unit Tests 3](#_30j0zll)

[2.0 Coverage Report 4](#_1fob9te)

[3.0 Requirements Acceptance Testing 5](#_3znysh7)

# Unit Tests

Delete the RED text and replace with your own

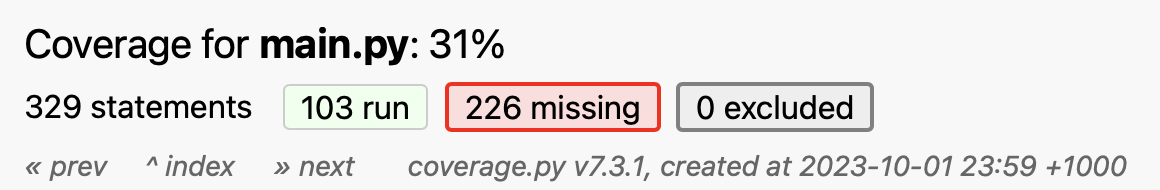
(In this table you fill out details about what unit tests you have done using the unittest module)

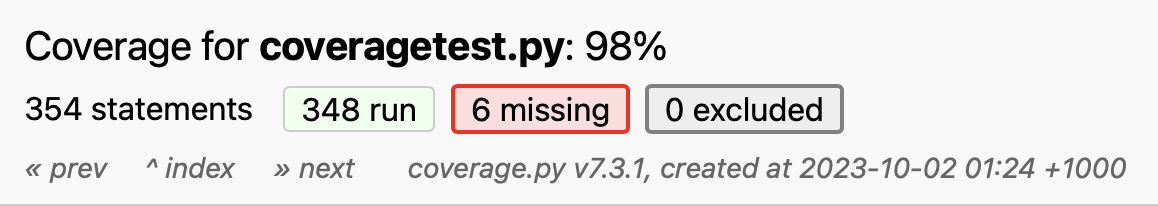
| **No** | **Test Case** | **Expected Results** | **Actual Results** |
| --- | --- | --- | --- |
| **1.0** | **Date Range is backwards** |  |  |
| **1.1** | **Date range is equal** |  |  |
| **1.2** | **Date range is linear** |  |  |
| 2.0 | Test a missing .csv file | Exception Handled | Program crashes |
| 3.0 |  |  |  |
| 1.2 |  |  |  |
| **2.0** | **Histogram Functions** |  |  |
| 2.1 | Empty input dictionary | Display error message and exit | Display error message and exit |

# Coverage Report

A description of the coverage of your unit tests, including how you evaluated coverage (function, statement, branch, condition)

Initially when running coverage, our coverage report was only showing 31%. A modified verion of main.py, called coveragetest.py was used to test the actual coverage of the code. The modified version added extra lines at the end of the main program to activate each of the functions used. Coveragetest.py was able to achieve a 98% coverage report with 6 lines of code not run. The lines of code were relating to errors which show as tkinter message boxes. The errors work as needed when the main program is run however we could not run coverage and execute multiple message boxes as doing so caused the coverage test to crash. These functions have been tested in the previous unit tests.





# Requirements Acceptance Testing

(You will need to fill out the column on the left with the requirements listed in software design documents and the columns on the right with the results of your own testing)

| **Software  Requirement No** | **Test** | **Implemented (Full /Partial/ None)** | **Test Results (Pass/ Fail)** | **Comments (for partial implementation or failed test results)** |
| --- | --- | --- | --- | --- |
| 1 | Specify a time period in the software and be shown data relating to the specified time period. | Full | Pass |  |
| 2 | Specify a time period in the software and be shown the average amount of accidents across a day at hourly intervals. | Full | Pass |  |
| 3 | Enter keywords relating to the accident and for the software to search and show data relating to the entered keyword(s). | Partial | Pass | Rather than user entered keywords, a dropdown list was used to show users options for word matching. |
| 4 | Clearly see the impact that alcohol has in Victorian accidents through visualisation. | Full | Pass |  |
| 5 | Select a year and view relevant data relating to accidents on public holidays. | None | Fail |  |
| 6 | The program shall accept a csv document for data interpretation. | Full | Pass |  |
| 7 | The program shall filter data from the CSV document including   * Date ranges * Times of day * Keywords * Alcohol involvement * Accidents involving public holidays | Full | Pass |  |
| 8 | The program shall create charts based on data.   * These charts are based on mostly filtered data and include but are not limited to bar charts, pie charts, and plots which are to be shown in a GUI. | Full | Pass |  |
| 9 | The program shall allow the exporting of generated charts and other visualisations.   * These exported files will be images. |  |  |  |