Software Testing Report

Accident Analysis Software

Nicolas Donaldson - S5256284

Juniper Lethbridge - S2884940

Toshimitsu Ota - S5251464

Table of Contents

[1.0 Unit Tests 3](#_Toc116051984)

[Test Results: 6](#_Toc116051985)

[2.0 Coverage Report 7](#_Toc116051986)

[3.0 Requirements Acceptance Testing 8](#_Toc116051987)

# Unit Tests

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Test Case** | **Expected Results** | **Actual Results** |
| **1.0** | **Search.hourly\_average** |  |  |
| 1.1 | Test expected return | Returns hourly average for a selected period | Returns hourly average for a selected period |
| 1.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **2.0** | **Search.getDateRange** |  |  |
| 2.1 | Test expected return | Queries accident database to get the min and max date within the search query | Queries accident database to get the min and max date within the search query |
| 2.2 | Test if instance of list | Return value is an instance of tuple | Return value is an instance of tuple |
| **3.0** | **Search.getTotalDays** |  |  |
| 3.1 | Test expected return | Queries database and returns a list of the difference between To\_date and From\_Date | Queries database and returns a list of the difference between To\_date and From\_Date |
| 3.2 | Test if instance of tuple | Return value is an instance of tuple | Return value is an instance of tuple |
| **4.0** | **Search.listAccidentType** |  |  |
| 4.1 | Test expected return | Queries accident database and returns a list of all unique accident types | queries accident database and returns a list of all unique accident types |
| 4.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **5.0** | **Search.accidentTypeList** |  |  |
| 5.1 | Test expected return | Calculates the number of accidents with accident keyword using accident\_type\_list | Calculates the number of accidents with accident keyword using accident\_type\_list |
| 5.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| 5.3 | Test expected return (mode=’alcohol’) | Calculates the number of accidents with accident keyword using accident\_type\_list | Calculates the number of accidents with accident keyword using accident\_type\_list |
| 5.4 | Test if instance of list (mode=’alcohol’) | Return value is an instance of list | Return value is an instance of list |
| **6.0** | **Search.calcAllAccidentType** |  |  |
| 6.1 | Test expected return | Calculates number of accidents of all accident types | Calculates number of accidents of all accident types |
| 6.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| 6.3 | Test expected return (mode=’alcohol’) | Calculates number of accidents of all accident types | Calculates number of accidents of all accident types |
| 6.4 | Test if instance of list(mode=’alcohol’) | Return value is an instance of list | Return value is an instance of list |
| **7.0** | **Search.Calculate\_by\_month** |  |  |
| 7.1 | Test expected return | Calculates the number of accidents in each month. | Calculates the number of accidents in each month. |
| 7.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| 7.3 | Test expected return (mode=’alcohol’) | Calculates the number of accidents in each month. | Calculates the number of accidents in each month. |
| 7.4 | Test if instance of list (mode=’alcohol’) | Return value is an instance of list | Return value is an instance of list |
| **8.0** | **Search.Calculate\_by\_day** |  |  |
| 8.1 | Test expected return | Calculates the number of accidents in each day. | Calculates the number of accidents in each day. |
| 8.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **9.0** | **Search.listLgas** |  |  |
| 9.1 | Test expected return | Queries accident database and returns a list of all unique LGA names | Queries accident database and returns a list of all unique LGA names |
| 9.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **10.0** | **Search.calcAllLgas** |  |  |
| 10.1 | Test expected return | Calculates the number of accidents in each LGA | Calculates the number of accidents in each LGA |
| 10.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **11.0** | **Search.calculateLGA** |  |  |
| 11.1 | Test expected return | Calculates the number of accidents within a given LGA | Calculates the number of accidents within a given LGA |
| 11.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **12.0** | **Search.listRegions** |  |  |
| 12.1 | Test expected return | Queries accident database and returns a list of all unique region names | Queries accident database and returns a list of all unique region names |
| 12.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **13.0** | **Search.matchRegions** |  |  |
| 13.1 | Test expected return | Find a match with self.Region out of list of unique region names | Find a match with self.Region out of list of unique region names |
| 13.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **14.0** | **Search.calcAllRegions** |  |  |
| 14.1 | Test expected return | Calculates the number of accidents in each region | Calculates the number of accidents in each region |
| 14.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |
| **15.0** | **Search.calculate\_region** |  |  |
| 15.1 | Test expected return | Calculates the number of accidents within a given region | Calculates the number of accidents within a given region |
| 15.2 | Test if instance of list | Return value is an instance of list | Return value is an instance of list |

## Test Results:

Text

Description automatically generated

# Coverage Report

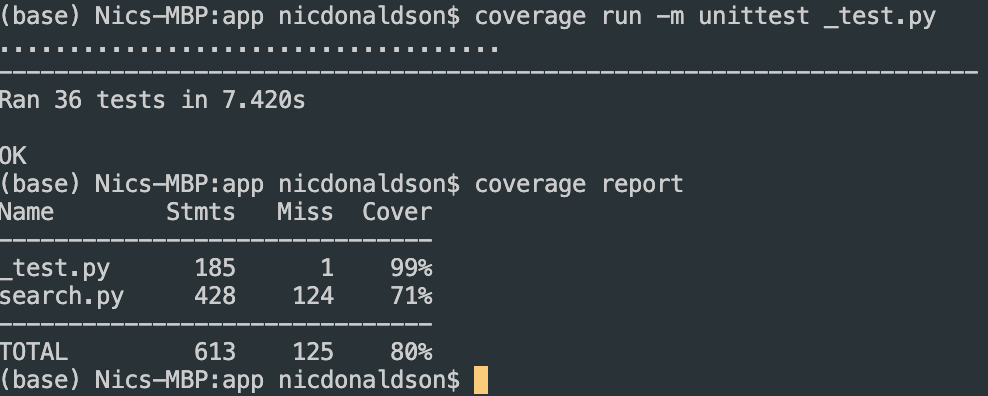


Image: Coverage Report

Search.py contained many functions. To achieve high coverage, this meant writing many test cases that individually tested each function. In total, 15 functions were tested. The intention was to assert the expected outcome of each function, along with testing the instance of each function. For functions that utilised the mode ‘alcohol’, additional tests were conducted on the expected outcome and instance, to verify no defects existed, when switching between modes for each function. To test these functions, a Search object had to be created. Attributes of the Search class included To\_Date, From\_Date, Accident\_Type\_List, Lga, and Region. All 36 test cases passed successfully. 99% of the testing document was covered, whilst 80% of search class file was covered.

# 3.0 Requirements Acceptance Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Software  Requirement No | Test | Implemented (Full /Partial/ None) | Test Results (Pass/ Fail) | Comments (for partial implementation or failed test results) |
| 1.1 | For a user-selected period, the program shall display the information of all accidents that happened in the period. | FULL | PASS |  |
| 1.2 | For a user-selected period, the program shall produce a chart to show the number of accidents in each hour of the day. | FULL | PASS |  |
| 1.3 | For a user-selected period, the program shall retrieve all accidents caused by an accident type that contains a keyword (user entered). | FULL | PASS |  |
| 1.4 | The program shall allow the user to analyse the impact of alcohol in accidents by generating charts with both alcohol related data and none-alcohol related data. | FULL | PASS |  |
| 1.5 | The program shall show geographical analysis of accidents by generating top 10 LGA chart, accidents by region chart, and accidents over map chart. | FULL | PASS |  |
| 2.0 | The program shall have a GUI implementation. | FULL | PASS |  |
| 2.1 | The program shall allow for user input through GUI. | FULL | PASS |  |
| 2.2 | The program shall accept a csv dataset file from user input. | FULL | PASS |  |
| 2.3 | The program will limit returned data by user input. | FULL | PASS |  |
| 2.4 | The program will limit returned data by time range. | FULL | PASS |  |
| 2.5 | The program shall graphically display data through charts & tables. | FULL | PASS |  |
| 2.6 | The program shall output a summary of results. | FULL | FAIL | Feature Removed |
| 3.0 | The program shall have a database via sqlite3 library. | FULL | PASS |  |
| 3.1 | The program shall allow for users to upload a .csv dataset file to the database | FULL | PASS |  |
| 3.2 | The program shall convert .csv files into a database | FULL | PASS |  |
| 3.3 | The program shall perform SQLite queries | FULL | PASS |  |
| 4.0 | The program shall have maths & statistical modules. | FULL | PASS |  |
| 5.0 | The program shall have a datetime module. | FULL | PASS |  |
| 6.0 | The program shall have a PyPlot module. | FULL | PASS |  |