Software Testing Report
Data Analysis on Australia NSW traffic penalty

Group Number: 43

Student Name: Bozhao Li

Student ID: s2187237

Student Name: Jennifer-wei Lin

Student ID: s5291152

Student Name: Sakibul Islam

Student ID: s5228873

Course Name: Software Technologies

Course code: 7810ICT

Date: 11/09/2022

Assignment: Part B

Table of Contents

1.0 Unit Tests 3

2.0 Coverage Report 12

3.0 Requirements Acceptance Testing 25

1.0 Unit Tests

No	Test Case	Expected Results	Actual Results
1.0	**Initiate functions**		
1.1	Test a date outside dd/mm/yy format, 20111/11/111	The function will take a date variable and attempt to convert it into dd/mm/yy format, spitting out an error if failed	The program strikes an error.
1.2	Test empty config file	Print “Load configuration file fails”	Print “Load configuration file fails” Program does not close
1.3	Test different config file Name=Test Description=Test System Version=0.2 Developer = test2	About tab reflects different config file	About tab reflects different config file
1.4	Load csv file from file path in float format with 2 decimal points and OFFENCE_MONTH column converted to dd/mm/yy	Load csv file from file path in float format with 2 decimal points and OFFENCE_MONTH column converted to dd/mm/yy	Load csv file from file path in float format with 2 decimal points and OFFENCE_MONTH column converted to dd/mm/yy

No	Test Case	Expected Results	Actual Results
2.0	**UI functions**		
2.1	Program loads all widgets, labels and format correctly	Program loads all widgets, labels and format correctly	Program loads all widgets,labels and format correctly
3.0	**Data cleaning and filtering functions**		
3.1	Drop irrelevant data columns for general search	'OFFENCE_FINYEAR', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_TYPE', 'LOCATION_CODE', 'LOCATION_DETAILS', 'SPEED_BAND', 'TOTAL_VALUE' columns dropped	'OFFENCE_FINYEAR', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_TYPE', 'LOCATION_CODE', 'LOCATION_DETAILS', 'SPEED_BAND', 'TOTAL_VALUE' columns dropped
3.2	Drop irrelevant data columns for Radar/Camera tab	'OFFENCE_FINYEAR', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_IND', 'CAMERA_TYPE', 'LOCATION_CODE', 'LOCATION_DETAILS', 'SCHOOL_ZONE_IND','SPEED_BAND', 'SPEED_IND', 'SEATBELT_IND', 'MOBILE_PHONE_IND', 'PARKING_IND', 'CINS_IND','FOOD_IND', 'BICYCLE_TOY_ETC_IND', 'TOTAL_VALUE' columns dropped	'OFFENCE_FINYEAR', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_IND', 'CAMERA_TYPE', 'LOCATION_CODE', 'LOCATION_DETAILS', 'SCHOOL_ZONE_IND','SPEED_BAND', 'SPEED_IND', 'SEATBELT_IND', 'MOBILE_PHONE_IND', 'PARKING_IND', 'CINS_IND','FOOD_IND', 'BICYCLE_TOY_ETC_IND', 'TOTAL_VALUE' columns dropped
3.3	Drop irrelevant data columns for Offence Code tab (Offence Description)	'OFFENCE_FINYEAR', 'OFFENCE_MONTH', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_IND', 'CAMERA_TYPE', 'LOCATION_CODE', 'LOCATION_DETAILS', 'SCHOOL_ZONE_IND','SPEED_BAND', 'SPEED_IND',	'OFFENCE_FINYEAR', 'OFFENCE_MONTH', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_IND', 'CAMERA_TYPE', 'LOCATION_CODE',

No	Test Case	Expected Results	Actual Results
		'POINT_TO_POINT_IND', 'RED_LIGHT_CAMERA_IND','SPEED_CAMERA_IND', 'SEATBELT_IND', 'MOBILE_PHONE_IND', 'PARKING_IND', 'CINS_IND','FOOD_IND', 'BICYCLE_TOY_ETC_IND', 'TOTAL_VALUE' columns dropped	'LOCATION_DETAILS', 'SCHOOL_ZONE_IND','SPEED_BAND', 'SPEED_IND', 'POINT_TO_POINT_IND', 'RED_LIGHT_CAMERA_IND','SPEED_CAMERA_IND', 'SEATBELT_IND', 'MOBILE_PHONE_IND', 'PARKING_IND', 'CINS_IND','FOOD_IND', 'BICYCLE_TOY_ETC_IND', 'TOTAL_VALUE' columns dropped
3.4	Filter and drop irrelevant data columns for Mobile Phone tab	'MOBILE_PHONE_IND'collumn filtered by Y and 'OFFENCE_FINYEAR', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_IND', 'CAMERA_TYPE', 'LOCATION_CODE', 'LOCATION_DETAILS', 'SCHOOL_ZONE_IND','SPEED_BAND', 'SPEED_IND', 'POINT_TO_POINT_IND', 'RED_LIGHT_CAMERA_IND','SPEED_CAMERA_IND', 'SEATBELT_IND', 'MOBILE_PHONE_IND', 'PARKING_IND', 'CINS_IND','FOOD_IND', 'BICYCLE_TOY_ETC_IND', 'TOTAL_VALUE' dropped	'MOBILE_PHONE_IND'collumn filtered by Y and 'OFFENCE_FINYEAR', 'LEGISLATION', 'SECTION_CLAUSE', 'FACE_VALUE','CAMERA_IND', 'CAMERA_TYPE', 'LOCATION_CODE', 'LOCATION_DETAILS', 'SCHOOL_ZONE_IND','SPEED_BAND', 'SPEED_IND', 'POINT_TO_POINT_IND', 'RED_LIGHT_CAMERA_IND','SPEED_CAMERA_IND', 'SEATBELT_IND', 'MOBILE_PHONE_IND', 'PARKING_IND', 'CINS_IND','FOOD_IND', 'BICYCLE_TOY_ETC_IND', 'TOTAL_VALUE' dropped
3.5	Clean data	na columns are dropped, duplicated rows are dropped, offence_month asc, offence_code are sorted by asc	na columns are dropped, duplicated rows are dropped, offence_month asc, offence_code are sorted by asc

No	Test Case	Expected Results	Actual Results
3.6	Fetch data by date	dataframe Offence Month column is searched by inputted startDate and endDate	dataframe Offence Month column is searched by inputted startDate and endDate
3.7	Offence code tab drill down by all total == None order <bool> False: descending order	Displays sum of all offense code for user defined period and sorts by desc	Displays sum of all offense code for user defined period
3.8	Offence code tab drill down by all total == None order <bool> True: ascending order	Displays sum of all offense code for user defined period and sorts by asc	Displays sum of all offense code for user defined period and sorts by asc
3.9	Test empty `listIndicators` `filter_data_by_indicators(dataframe,['0','0','0','0','0','0','0','0','0','0','0','0'])`	`return dataframe[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]`	`return dataframe[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]`

3.10	Seat belt indicator selected.
	
	`filter_data_by_indicators(dataframe,['1','0','0','0','0','0','0','0','0','0','0','0'])`
		`if listIndicators[0] == "1":`
		
		`df1 = dataframe[(dataframe['SEATBELT_IND'] == "Y")]`
			`return dataframe[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]`

No	Test Case	Expected Results	Actual Results
		`print(df1[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']])`	

3.11	Test all indicators selected
	
	`filter_data_by_indicators(dataframe,['1','1','1','1','1','1','1','1','1','1','1','1'])`
		`df1 = pandas.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')`
		
		`print(df1[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']])`
			`df1 = pandas.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')`
			
			`print(df1[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']])`

3.12	Radar/Camera tab drill down by school keyword. `fetch_data_by_offence_description(dataframe,"school")`	Dataframe filtered by keyword	Dataframe filtered by keyword
3.13	Radar/Camera tab drill down by fool keyword. `fetch_data_by_offence_description(dataframe,"fool")`	No keyword found	No keyword found

No	Test Case	Expected Results	Actual Results
3.14	Mobile Phone tab drill down by all `getMobilePhoneAllOffences(dataframe)`	Dataframe filtered by `['MOBILE_PHONE_IND'] == "Y")`	Not all types of offences shown
3.15	Mobile Phone tab drill down by offence description	Mobile Phone tab drill down by offence description	Mobile Phone tab drill down by offence description
4.0	**Main program**		
4.1	Offence code tab scatter diagram	After start and end dates are searched, pressing drill down on the offence code tab creates a scatter diagram with a title, x and y axes labelled and distinct colours for each offence code	After start and end dates are searched, pressing drill down on the offence code tab creates a scatter diagram with a title, x and y axes labelled and distinct colours for each offence code
4.2	Radar/Camera tab bar chart	After start and end dates are searched, pressing drill down on the Radar/Camera tab creates a bar chart with a title, x and y axes labelled and green colour.	After start and end dates are searched, pressing drill down on the Radar/Camera tab creates a bar chart with a title, x and y axes labelled and green colour.
4.3	Mobile Phone tab line chart	After start and end dates are searched, pressing drill down on the Mobile Phone tab creates a line chart with a title, x and y axes labelled and blue colour in contrast to previous Radar/Camera tab.	After start and end dates are searched, pressing drill down on the Mobile Phone tab creates a line chart with a title, x and y axes labelled and

No	Test Case	Expected Results	Actual Results
			blue colour in contrast to previous Radar/Camera tab.
4.4	Show detail for all tabs	After start and end dates are searched and drill down tab is pressed on a tab, pressing show detail on that corresponding tab will create a pop up that allows users to hover over the graph where each data point will have detail about offence code, offence description and number of cases	After start and end dates are searched and drill down tab is pressed on a tab, pressing show detail on that corresponding tab will create a pop up that allows users to hover over the graph where each data point will have detail about offence code, offence description and number of cases
4.5	Compare two tabs	After start and end dates are searched and drill down tab is pressed on a tab, pressing the compare button below it transforms the text to compare to and navigating to any other tab will have their compare button also labelled compare to. If the other tab has also had the drill down button pressed, the second compare to button a user presses creates a pop up of two graphs for side by side comparison where they will no longer be able to hover over for data point detail about offence code, offence description and number of cases. Upon closing the popup, the compare button is returned to the label compare.	After start and end dates are searched and drill down tab is pressed on a tab, pressing the compare button below it transforms the text to compare to and navigating to any other tab will have their compare button also labelled compare to. If the other tab has also had the drill down button pressed, the second compare to button a user presses creates a pop up of two graphs for side by side comparison where they will no longer be able to hover over for data point detail about offence code,

No	Test Case	Expected Results	Actual Results
			offence description and number of cases. Upon closing the popup, the compare button is returned to the label compare.
4.6	Test load new dataset with min date of 01/01/1900, max date of 01/01/2022 and abcd in description of `'MOBILE_PHONE_IND' == Y`	Csv file is loaded from the filepath. The file is searched for max a min dates which is converted into dd/mm/yy format. Upon opening the program, the welcome text has changed to reflect the min and max time period of the new dataset. The default start and end date in the program has also been updated to the max and min of the new dataset.	Csv file is loaded from the filepath. The file is searched for max a min dates which is converted into dd/mm/yy format. Upon opening the program, the welcome text has changed to reflect the min and max time period of the new dataset. The default start and end date in the program has also been updated to the max and min of the new dataset.
4.7	Test load with invalid format of no column named OFFENCE_MONTH	Exception error message popup `"The dataset file is invalid, please choose a correct file."`	Exception error message popup `"The dataset file is invalid, please choose a correct file."`
4.8	Test load with invalid format of xlxs file	Exception error message popup `"The dataset file is invalid, please choose a correct file."`	Exception error message popup `"The dataset file is invalid, please choose a correct file."`
4.9	Reset to original dataset	Reset to specifications in config.ini [default] OriginalFilePath=origin\	Reset to specifications in config.ini [default] OriginalFilePath=origin\

No	Test Case	Expected Results	Actual Results
		OriginalFIleName=data.csv	OriginalFIleName=data.csv
4.10	Test reset to deleted original dataset	Exception error message popup `"The dataset file is invalid, please choose a correct file."`	Exception error message popup `"The dataset file is invalid, please choose a correct file."`
4.11	Quit program	System exit	System exit

2.0 Coverage Report

No	Test Case	Coverage
1.0	**Initiate functions**	Function coverage – Each function in the program has been called at least once Statement coverage – Each statement/line in the program has been executed.
1.1	Test a date outside dd/mm/yy format, 20111/11/111	
1.2	Test empty config file	Branch coverage – Except branch tested
1.3	Test different config file Name=Test Description=Test System Version=0.2 Developer = test2	Branch coverage – Except branch tested
1.4	Load csv file from file path in float format with 2 decimal points and OFFENCE_MONTH column converted to dd/mm/yy	

No	Test Case	Coverage
2.0	**UI functions**	Function coverage – Each function in the program has been called at least once Statement coverage – Each statement/line in the program has been executed
2.1	Program loads all widgets, labels and format correctly	
3.0	**Data cleaning and filtering functions**	Function coverage – Each function in the program has been called at least once Statement coverage – Each statement/line in the program has been executed
3.1	Drop irrelevant data columns for general search	
3.2	Drop irrelevant data columns for Radar/Camera tab	
3.3	Drop irrelevant data columns for Offence Code tab (Offence Description)	
3.4	Filter and drop irrelevant data columns for Mobile Phone tab	
3.5	Clean data	
3.6	Fetch data by date	
3.7	Offence code tab drill down by all	Branch coverage – if branch tested

No	Test Case	Coverage
	total == None order <bool> False: descending order	Condition coverage - Boolean sub-expression order <bool> False
3.8	Offence code tab drill down by all total == None order <bool> True: ascending order	Branch coverage – if branch tested Condition coverage - Boolean sub-expression order <bool> True

3.9	Test empty `listIndicators`
	
	`filter_data_by_indicators(dataframe,['0','0','0','0','0','0','0','0','0','0','0','0'])`
		Branch coverage – if branch tested
		
		`df1 = pd.DataFrame(data=None, columns=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'])`
		
		`if df1.empty:`
		
		`# return the input dataframe`
		
		`return dataframe[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]`

3.10	Seat belt indicator selected.
	
	`filter_data_by_indicators(dataframe,['1','0','0','0','0','0','0','0','0','0','0','0'])`
		Branch coverage – if branch tested
		
		`df1 = pd.DataFrame(data=None, columns=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'])`
		
		`# seat belt indicator selected`
		
		`if listIndicators[0] == "1":`
		
		`df1 = dataframe[(dataframe['SEATBELT_IND'] == "Y")]`

No	Test Case	Coverage

		Branch coverage – else branch tested
		`if df1.empty:`
		`# return the input dataframe`
		`return dataframe[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]`
		`else:`
		`# return the merged dataframe`
		`return df1[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]`

3.11	Test all indicators selected listIndicator = ["1", "1", "1", "1", "1", "1", "1", "1", "1", "1", "1", "1"] print(check_filter_data_by_indicators(dataframe, listIndicator)) listIndicator = ["0", "1", "1", "1", "1", "1", "1", "1", "1", "1", "1", "1"] print(check_filter_data_by_indicators(dataframe, listIndicator)) listIndicator = ["0", "0", "1", "1", "1", "1", "1", "1", "1", "1", "1", "1"]	Branch coverage – if branch tested # seat belt indicator selected if listIndicators[0] == "1": df1 = dataframe[(dataframe['SEATBELT_IND'] == "Y")] # seat belt indicator selected if listIndicators[0] == "1": df1 = dataframe[(dataframe['SEATBELT_IND'] == "Y")] # speed indicator selected

No	Test Case	Coverage

	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "1", "1", "1", "1", "1", "1", "1", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "0", "1", "1", "1", "1", "1", "1", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "0", "0", "1", "1", "1", "1", "1", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "0", "0", "0", "1", "1", "1", "1", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "0", "0", "0", "0", "1", "1", "1", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
		if listIndicators[1] == "1":
		
		df2 = dataframe[(dataframe['SPEED_IND'] == "Y")]
		
		if df1.empty:
		
		df1 = df2
		
		else:
		
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# parking indicator selected
		
		if listIndicators[2] == "1":
		
		df2 = dataframe[(dataframe['PARKING_IND'] == "Y")]
		
		if df1.empty:
		
		df1 = df2
		
		else:
		
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')

No	Test Case	Coverage

	listIndicator = ["0", "0", "0", "0", "0", "0", "0", "0", "1", "1", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "0", "0", "0", "0", "0", "0", "1", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "1", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
	
	listIndicator = ["0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "1"]
	
	print(check_filter_data_by_indicators(dataframe, listIndicator))
		# school zone indicator selected
		
		if listIndicators[3] == "1":
		
		df2 = dataframe[(dataframe['SCHOOL_ZONE_IND'] == "Y")]
		
		if df1.empty:
		
		df1 = df2
		
		else:
		
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# mobile phone indicator selected
		
		if listIndicators[4] == "1":
		
		df2 = dataframe[(dataframe['MOBILE_PHONE_IND'] == "Y")]
		
		if df1.empty:
		
		df1 = df2
		
		else:
		
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')

No	Test Case	Coverage

		# food safety indicator selected
		if listIndicators[5] == "1":
		df2 = dataframe[(dataframe['FOOD_IND'] == "Y")]
		if df1.empty:
		df1 = df2
		else:
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# point to point camera indicator selected
		if listIndicators[6] == "1":
		df2 = dataframe[(dataframe['POINT_TO_POINT_IND'] == "Y")]
		if df1.empty:
		df1 = df2
		else:

No	Test Case	Coverage

		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# red light camera indicator selected
		if listIndicators[7] == "1":
		df2 = dataframe[(dataframe['RED_LIGHT_CAMERA_IND'] == "Y")]
		if df1.empty:
		df1 = df2
		else:
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# speed camera indicator selected
		if listIndicators[8] == "1":
		df2 = dataframe[(dataframe['SPEED_CAMERA_IND'] == "Y")]
		if df1.empty:
		df1 = df2

No	Test Case	Coverage

		else:
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# criminal infringement notice scheme indicator selected
		if listIndicators[9] == "1":
		df2 = dataframe[(dataframe['CINS_IND'] == "Y")]
		if df1.empty:
		df1 = df2
		else:
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# bicycle, wheeled toy and other non-Motor vehicle indicator selected
		if listIndicators[10] == "1":
		df2 = dataframe[(dataframe['BICYCLE_TOY_ETC_IND'] == "Y")]
		if df1.empty:

No	Test Case	Coverage

		df1 = df2
		else:
		df1 = pd.merge(df1, df2, on=['OFFENCE_MONTH', 'OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER'], how='outer')
		
		# no checkbox is selected
		if df1.empty:
		# return the input dataframe
		return dataframe[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]
		else:
		# return the merged dataframe
		return df1[['OFFENCE_CODE', 'OFFENCE_DESC', 'TOTAL_NUMBER']]

3.12	Radar/Camera tab drill down by school keyword. `fetch_data_by_offence_description(dataframe,"school")`	Branch coverage – if branch tested flag = True

No	Test Case	Coverage
3.13	Radar/Camera tab drill down by fool keyword. `fetch_data_by_offence_description(dataframe,"fool")`	Branch coverage – if branch tested flag = False
3.14	Mobile Phone tab drill down by all `getMobilePhoneAllOffences(dataframe)`	

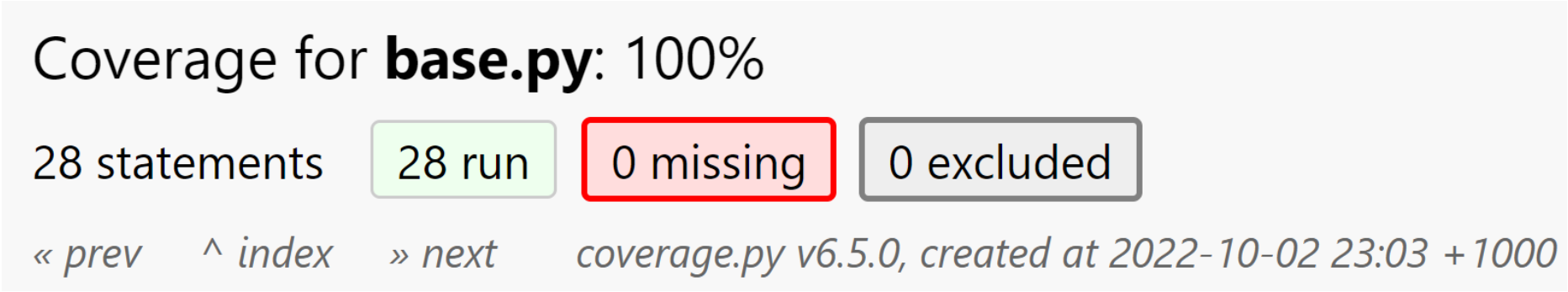
3.15	Mobile Phone tab drill down by offence description	Branch coverage – if branch tested
		if content == None:
		for mobileDescription in listMobileDescription:
		# fetch data by offence description which contains key words
		dfTemp = dataframe.loc[dataframe['OFFENCE_DESC'] == mobileDescription]
		dfTemp = dfTemp.sort_values(by='OFFENCE_MONTH', ascending=True, ignore_index=True)
		arrayDataframe.append(dfTemp)

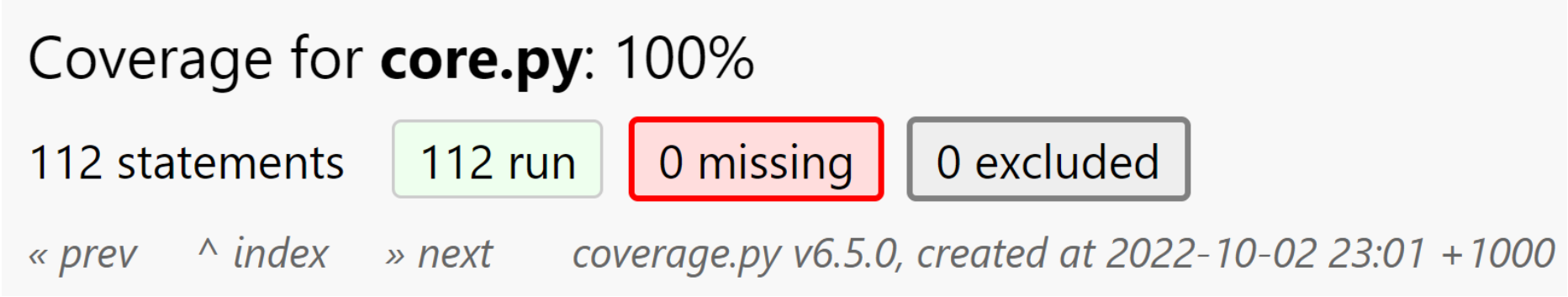
4.0	**Main program**	Function coverage – Each function in the program has been called at least once Statement coverage – Each statement/line in the program has been executed. Branch coverage – branches tested

No	Test Case	Coverage
		Condition coverage - Boolean sub-expression tested
4.1	Offence code tab scatter diagram	
4.2	Radar/Camera tab bar chart	
4.3	Mobile Phone tab line chart	
4.4	Show detail for all tabs	
4.5	Compare two tabs	
4.6	Test load new dataset with min date of 01/01/1900, max date of 01/01/2022 and abcd in description of `'MOBILE_PHONE_IND' == Y`	
4.7	Test load with invalid format of no column named OFFENCE_MONTH	
4.8	Test load with invalid format of xlxs file	
4.9	Reset to original dataset	
4.10	Test reset to deleted original dataset	
4.11	Quit program	

The coverage report result is shown below.

Module	*statements*	*missing*	*excluded*	*coverage*
base.py	28	0	0	100%
Core.py	112	0	0	100%





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	The program lets user upload a new dataset if all the fields are similar to NSW Traffic Penalty Dataset	Full	Pass	
2	Display error if the new dataset is not valid	Full	Pass	
3	The title and description of the program are displayed on the software home screen and are visible.	Full	Pass	
4	The colour scheme of the software is coherent and ambient	Full	Pass	
5	Display all the offences by selecting start and end dates	Full	Pass	
6	View a specific offence for a period by selecting offence code	Full	Pass	
7	Combine two or more offence codes to view the result	Full	Pass	
8	Display the relevant data description after drilling down offence codes	Full	Pass	

Software Requirement No	Test	Implemented (Full /Partial/ None)	Test Results (Pass/ Fail)	Comments (for partial implementation or failed test results)
9	Display graphs by selecting type of indicators/ captured by/ mobile phone usage criteria	Full	Pass	
10	Change tabs/ options to drill down different kind of offences	Full	Pass	
11	Query offences using keywords	Full	Pass	
12	The program lets user download the relevant graphs/ charts	Full	Pass	
13	The program queries data at a response rate of no more than 2 seconds	Full	Pass	
14	After using a new dataset, the program lets user revert to original NSW Traffic Penalty Dataset	Full	Pass	
15	Original dataset of the program remains unchanged after the queries	Full	Pass	
16	The program shows records found and response time in the status bar	Full	Pass	
17	The program lets user view the largest and the smallest count of offences for the selected period	Full	Pass	