SE 3XA3: Test Report
Ultimate Calculator

Group 15 L01
Jarod Rankin, rankij5
Mathew Petronilho, petronim
Logan Brown, brownl33
Syed Bokhari, bokhars

April 8, 2022

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
April 5 2022	1.0	FR and NFR Evaluation
April 7 2022	1.1	Rest of document complete

This document outlines the results of all the testing done for the Ultimate Calculator, as stated in the test plan.

1 Functional Requirements Evaluation

1.1 Calculation Testing

Test: FR-C-T1

Description: Test to ensure all operation section windows open properly, they each have a way to calculate an output, and can calculate the correct output synchronously along with the other windows

Type: Functional, Dynamic, Manual

Initial State: Operation section windows are open

Input: Press of the calculate button on each window

Output: The operation answer in the respective operation section window

Expected: The correct output of the answer should be visible in the output text box when the calculate button is pressed. Calculations can be run synchronously

Result: Pass

Test: FR-C-T2

Description: Test to determine if appropriate error message is returned after mathematical operations with undefined outputs

Type: Functional, Dynamic, Manual

Initial State: Operation section window is open

Input: Numbers that will cause undefined outputs

Output: Error message

Expected: Application should remain working and error message should be displayed

Result: Pass

Test: FR-C-T3

Description: Test to determine if calculator outputs are mathematically correct

Type: Unit, Dynamic, Automated

Initial State: Application is running

Input: Valid arbitrary inputs

Output: Correct calculation

Expected: All operations implemented by the calculator should return mathematically correct results

Result: Pass

1.2 User Interface Testing

Test: FR-UI-T1

Description: Test to determine if application starts at main menu and that all operation types are visible from there

Type: Functional, Dynamic, Manual

Initial State: An empty command line terminal

Input: Initialization of the Ultimate Calculator application through the command line

Output: A main menu window for the application

Expected: The main menu along with the operation types of conversions, algebra, stocks, health, GPA, geometry, and binary should all be visible

Result: Pass

Test: FR-UI-T2

Description: Test to determine if the minimum amount of operation types are present

Type: Functional, Dynamic, Manual

Initial State: Main menu for the application has been initialized

Input: Selection of all operation types

Output: The operation type windows

Expected: The amount of operation types present in the system is more than MIN_UNIQUE_OP

Result: In total there are 7 unique operation types, which is more than MIN_UNIQUE_OP
Pass

Test: FR-UI-T3

Description: Test to determine if the minimum amount of operation types are present and that they all open with empty inputs

Type: Functional, Dynamic, Manual

Initial State: Operation type windows are open

Input: Selection of all operations belonging to a specific operation type

Output: The operation sections

Expected: The amount of operation sections present for each operation type is more than MIN_OP_SECTION. Inputs are empty upon opening of each operation section

Result: Each operation type has the required amount of operation sections and they all open with empty inputs
Pass

Test: FR-UI-T4

Description: Test to determine if each operation window has the total number of inputs be equal to or greater than MIN_INPUT, and ensure that each input field allows the user to properly input values

Type: Functional, Dynamic, Manual

Initial State: Operation type windows are open

Input: Selection of all operation section windows

Output: The operation sections

Expected: Each input field in each operation window should be visually populated with a number

Result: Pass

Test: FR-UI-T5

Description: Test to determine if error message is displayed for invalid inputs

Type: Functional, Dynamic, Manual

Initial State: Operation section window is open

Input: Non-valid input type

Output: Warning

Expected: Application should not crash upon entering invalid inputs, and should display a message to tell the user an error occurred

Result: Every operation section outputs appropriate error message with invalid input
Pass

Test: FR-UI-T6

Description: Test to determine if error message is displayed after attempting to calculate with empty inputs

Type: Functional, Dynamic, Manual

Initial State: Operation section window is open

Input: Empty inputs

Output: Warning

Expected: Application should not crash upon entering no inputs, and should display a message to tell the user an error occurred

Result: Pass

Test: FR-UI-T7

Description: Test to determine if an output is always displayed with valid arbitrary inputs

Type: Functional, Dynamic, Manual

Initial State: Operation section window is open

Input: Valid arbitrary inputs

Output: Display of output

Expected: Application should display the result of the calculation to the user interface

Result: Pass

Test: FR-UI-T8

Description: Test to verify clear button works on applicable operation sections

Type: Functional, Dynamic, Manual

Initial State: Operation section window is open

Input: Clear button

Output: Empty text boxes

Expected: Application should clear any text boxes of their current text

Result: Pass

Test: FR-UI-T9

Description: Test to determine if each operation window has a close button and the window closes once the button is clicked

Type: Functional, Dynamic, Manual

Initial State: Operation windows are open

Input: Selection of the close button on the operation window

Output: Operation type window closes

Expected: The operation type windows will close once the close button is clicked

Result: Pass

Test: FR-UI-T10

Description: Test to determine if each window has a close button and if the main calculator window prompts the user with a question confirming their choice to close the program. All windows should close when the main menu window is closed

Type: Functional, Dynamic, Manual

Initial State: Operation type and main menu windows are open

Input: Selection of the close button on the main menu window

Output: Operation type window closes

Expected: The operation type windows and the main calculator windows will close once the close button is clicked from the main menu window and the dialog is confirmed

Result: Pass

2 Nonfunctional Requirements Evaluation

2.1 Look and Feel Testing

Test: NFR-LF-T1

Description: Tests that the main menu GUI is similar in appearance to a standard calculator

Type: Static, Manual

Tester(s): Survey participants

Pass: Average survey score of at least SURVEY_SCORE% for question 1

Result: Pass

Test: NFR-LF-T2

Description: Tests that the GUIs of the application all have a coherent design

Type: Static, Manual

Tester(s): Testing team and survey participants

Pass: All font styles, font sizes, colours, and buttons used in the application are consistent and an average survey score of at least SURVEY_SCORE% for question 2

Result: Pass

2.2 Usability Testing

Test: NFR-U-T1

Description: Tests that all the navigational buttons open up the correct windows and that all operation sections can be reached easily

Type: Dynamic, Manual

Tester(s): Testing team

Pass: Navigational buttons open their corresponding windows and all operation sections can be reached within MAX_NAVIGATION_CLICKS mouse clicks from the main menu

Result: Pass

Test: NFR-U-T2

Description: Tests that the navigational buttons on the application are descriptive and contain icons that allow the user to seamlessly transition from one menu to another

Type: Dynamic, Manual

Tester(s): Testing team

Pass: Results from the usability survey determine that users found the buttons descriptive and that navigating through the application was easy

Result: Pass

2.3 Performance Testing

Test: NFR-P-T1

Description: Tests that when the application opens each window the window will open in equal to or less than the MAX_RESPONSE_TIME

Type: Functional, Dynamic, Manual

Tester(s): Testing team

Pass: Each window is opened in less than the MAX_RESPONSE_TIME

Result: Pass

Test: NFR-P-T2

Description: Tests that when the application opens each operation window the application will compute an answer for each operation type equal to or less than the MAX_RESPONSE_TIME

Type: Functional, Dynamic, Manual

Tester(s): Testing team

Pass: Each operation is completed in less than the MAX_RESPONSE_TIME

Result: Pass

Test: NFR-P-T3

Description: Tests that outputs to calculations give a maximum number of significant digits

Type: Functional, Dynamic, Manual

Tester(s): Testing Team

Pass: Arbitrary valid input gives outputs with MAX_SIG_FIGS amount of significant digits

Result: Pass

2.4 Operational and Environmental Testing

Test: NFR-OE-T1

Description: Tests that the application works without an internet connection

Type: Functional, Dynamic, Manual

Tester(s): Testing team

Pass: Application starts while disconnected from internet

Result: Pass

2.5 Maintainability and Support Requirements

Test: NFR-MS-T1

Description: Tests that the application is an open source software where anyone can submit bugs and issues

Type: Structural, Static, Manual

Tester(s): Testing team

Pass: The code base is accessible via GitHub or GitLab and the tester is able to create an open issue and track the status of the changes.

Result: Pass

Test: NFR-MS-T2

Description: Tests that the system allows new operations to be added

Type: Structural, Static, Manual

Tester(s): Testing team

Pass: The code base is viewed and the code is modular exhibits low coupling and high cohesion

Result: Pass

2.6 Survey Results

As mentioned in the test plan, we conducted a survey to evaluate the usability and appearance of the calculator. We surveyed 10 people, including fellow students and family members. The results can be seen in Figure 1.

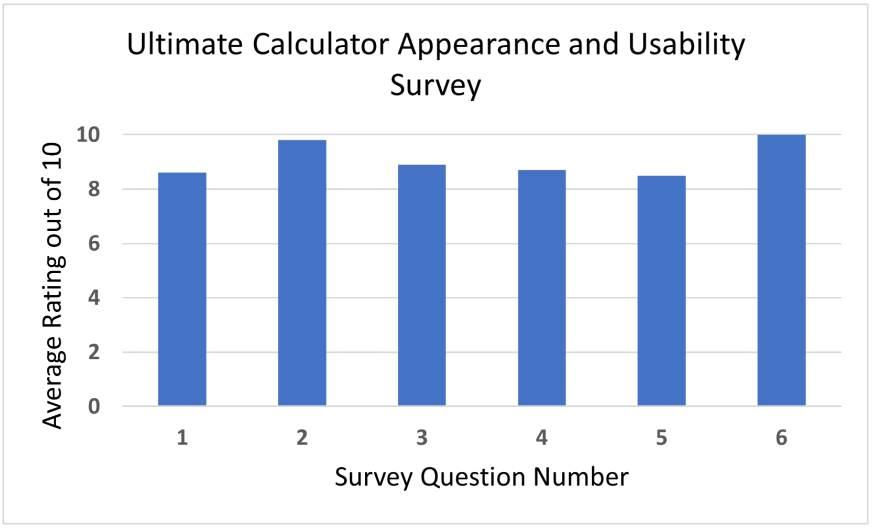

Figure 1: Survey Results

As you can see, the overall results of the survey were very positive. They helped to aid in completing our tests for our non-functional requirements.

3 Comparison to Existing Implementation

In the original source code of the Ultimate Calculator, there were no test cases that were implemented. In the updated version of Ultimate calculator, the source code was designed to be more modular than the original. The various calculator functionalities were split into their respective modules and unit testing was done to ensure accurate calculation results. The development team created documentation including a software requirements specification, test plan document and design documentation to organize the information via traceability of the requirements and the modules. The documentation also helps to catalog the test cases. The functional requirements were met through various testing methods such as unit testing and visual testing and there were a total of 110 unit test cases prior to the final demonstration of the project.

4 Unit Testing

Unit testing was done for each module that contained the calculators functionality. These modules are found in the calculators file in the projects repository. These files provide the functionality to each module in the application. Using the unittest library, each tester created a unit test file for the modules they were responsible for creating and maintaining. At a minimum basic units tests were created for each function, along with tests that made sure the error handling for each function worked correctly.

5 Changes Due to Testing

The unit tests and manual tests were helpful in finding small oversights that we were unaware of. Some errors we found include numbers being too large to be displayed in the output text boxes, users being able to enter an infinite amount of characters which could break the application, and a handful of other errors. All these errors were fixed in the application following their discovery. Testing did not really change the requirements but was quite helpful in finding errors we didn't consider.

6 Automated Testing

Automated testing was used for the calculation portion of the calculator application. A unit testing module was created for each calculation module and every method was thoroughly tested to ensure accurate calculator results. The python unittest framework was used to test various calculation method outputs to ensure they met the threshold of a correct output.

7 Trace to Requirements

Table 2: **Traceability Matrix for Calculation Requirements**

		Requirements									
		FR1	FR2	FR3	FR4	FR5	FR6	FR7	FR8	FR9	FR10
Test Cases	FR-C-T1						X	X			
	FR-C-T2										
	FR-C-T3										

Table 3: **Traceability Matrix for Calculation Requirements Continued**

		Requirements										
		FR11	FR12	FR13	FR14	FR15	FR16	FR17	FR18	FR19	FR20	FR21
Test Cases	FR-C-T1											
	FR-C-T2	X										
	FR-C-T3				X							

Table 4: **Traceability Matrix for UI Requirements**

		Requirements									
		FR1	FR2	FR3	FR4	FR5	FR6	FR7	FR8	FR9	FR10
Test Cases	FR-UI-T1	X			X						
	FR-UI-T2		X								
	FR-UI-T3			X		X					
	FR-UI-T4								X	X	X
	FR-UI-T5										
	FR-UI-T6										
	FR-UI-T7										
	FR-UI-T8										
	FR-UI-T9										
	FR-UI-T10										

Table 5: **Traceability Matrix for UI Requirements Continued**

		Requirements										
		FR11	FR12	FR13	FR14	FR15	FR16	FR17	FR18	FR19	FR20	FR21
Test Cases	FR-UI-T1											
	FR-UI-T2											
	FR-UI-T3											
	FR-UI-T4											
	FR-UI-T5		X									
	FR-UI-T6			X								
	FR-UI-T7					X						
	FR-UI-T8						X	X				
	FR-UI-T9								X			
	FR-UI-T10									X	X	X

Table 6: **Traceability Matrix for Non-Functional Requirements**

		Requirements									
		NFR1	NFR2	NFR3	NFR4	NFR5	NFR6	NFR7	NFR8	NFR9	NFR10
Test Cases	NFR-LF-T1	X									
	NFR-LF-T2		X								
	NFR-U-T1			X							
	NFR-U-T2				X						
	NFR-P-T1					X					
	NFR-P-T2						X				
	NFR-P-T3							X			
	NFR-OE-T1								X		
	NFR-MS-T1									X	
	NFR-MS-T2										X

8 Trace to Modules

Req.	Modules
FR1	M1
FR2	M1
FR3	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR4	M1
FR5	M3, M5, M6, M7, M8, M10, M11, M12, M14, M15, M16, M17, M18, M20, M21, M22, M23, M25, M26, M27, M28, M30
FR6	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR7	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR8	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR9	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR10	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR11	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR12	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR13	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR14	M2, M4, M9, M13, M19, M24, M29, M31
FR15	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR16	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR17	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR18	M5, M6, M7, M10, M11, M14, M15, M16, M17, M20, M21, M22, M25, M26, M27, M28, M30
FR19	M1
FR20	M1, M3, M5, M6, M7, M8, M10, M11, M12, M14, M15, M16, M17, M18, M20, M21, M22, M23, M25, M26, M27, M28, M30
FR21	M1

Table 7: Trace Between Requirements and Modules

9 Code Coverage Metrics

In the test plan, our goal was to have a STATEMENT_COV % statement coverage in regards to any modules in which unit testing is applicable. Through the use of a tool known as Coverage.py, we found these results:

1. Statement coverage of algebra_calculator.py: 94%
2. Statement coverage of binary_calculator.py: 99%
3. Statement coverage of conversion_calculator.py: 96%
4. Statement coverage of geometry_calculator.py: 95%
5. Statement coverage of gpa_calculator.py: 100%
6. Statement coverage of health_calculator.py: 100%
7. Statement coverage of main_calculator.py: 95%
8. Statement coverage of stocks_calculator.py: 100%

As you can see from the results we gathered, we well surpassed our goals for statement coverage with unit testing.

10 Appendix

10.1 Symbolic Parameters

SURVEY_SCORE = 80
STATEMENT_COV = 80
MAX_RESPONSE_TIME = 2
MAX_NAVIGATION_CLICKS = 2
MAX_SIG_FIGS = 64
MIN_UNIQUE_OP = 5
MIN_OP_SECTION = 1
MIN_INPUT = 1

10.2 Usability Survey Questions

All questions will be answered on a 1-10 scale

1. How familiar does the main menu screen feel to a standard calculator?
2. How cohesive do the styles of each window (colours, button sizes, input methods, etc.) feel to one another?

3. Starting from the main menu, try navigating to the Temperature Converter operation. How easy was it to locate said operation?
4. How intuitive does the navigation between different sections of the calculator feel?
5. How fluid do the transitions between different operations of calculator feel?
6. How timely do the answers received from calculations feel?

10.3 Generic Calculator for Comparison

Figure 2: Generic Calculator