**‘Hammurabi’ testing documentation**

Test computer specification:

OS Name Microsoft Windows 10 Home

Version 10.0.17134 Build 17134

System Name MSI

System Model GE62 6QF

Processor Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz, 2601 Mhz, 4 Core(s), 8 Logical Processor(s)Installed Physical Memory (RAM) 16.0 GB

Contents:

2 – User interface

3/4 – Data handling

4 – Speed Tests

5 – File management & References

(For installation and de-installation tests, refer to publication documentation)

These tests are ‘White Box tests’, performed in reference to source (1)

User Interface Tests

Testing the UI of the programme, including logical math checks and all buttons and inputs working as intended

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test number | Test | Method | Expected | Actual | Notes |
| UI 1 | ‘New game’ button test | Click of the ‘New game’ button on loading form and main form | Game is reset to year 1 variables, main form is displayed | Game is reset to year 1 variables, main form is displayed | Works as intended |
| UI 2 | ‘Make it so’ button Test | Click of the ‘Make it so’ button on the main form | Variables are stored from user input, yearPass function called | Variables are stored from user input, yearPass function is called | Works as intended |
| UI 3 | Text boxes functionality | Input non-integers into text boxes. (String, double, blank) | Messagebox should appear alerting user that input is invalid, when ‘make it so’ is pressed | Messagebox appears when ‘make it so’ is pressed, whenever non-integer data is present in a text box | Works as intended, but not consistent – txtSellLand alerts immediately, other 2 only function on button press. Adjust for consistency |
| UI 4 | Interface graphics auto-adjustment | User ‘Make it so’ button to call yearPass, update variables, and check picture boxes & graph reflect values | Picture boxes and graph should adjust to show the variables after yearPass is called, to given values | Images update to current values, graph shows stored variables | Works as intended |
| UI 5 | Picture box on\_click functions | Click on the 3 picture boxes to display the relevant message boxes | When clicked, a message box should show data. On close, will return to form | When clicked, Message box displays relevant data to user. Closes and returns to form | Works as intended |
| UI 6 | Graph adjustment via radio buttons | Click radio buttons to adjust the data graphs show | Graphs adjust to relevant data, depending on what radio button is clicked | When clicked, relevant data is displayed as the series on the graph | Works as intended |

Data handling tests

Test how the programme handles data. Includes exception handling, variable calculation and data capture.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test number | Test | Method | Expected | Actual | Notes |
| DH 1 | ‘Scores.txt.’ score file opens, data is read into an array which is sorted high to low, and displayed | Should happen at point of form load, in ‘lstScores’ | On load, lstScores will display saved data from the ‘Scores.txt’ file and order it | Displays a sorted list on load | Works as intended |
| DH 2 | Test that ‘newGame’ function resets variables. | Press of ‘new game’ button on load and on main form | Bushels = 2800,  Population = 100,  Land = 1000. | On button press, variables for population, bushels and land are reset to original values | Works as intended |
| DH 3 | BushelsTest and Landtest auto adjust with inputted values, after passing through leave box function | Input -100 & 100 into txtSellLand, leave box, check resulting calculation is correct to expected | Inputting -100 takes land down 100, and adjusts bushels up to land value x100. Opposite with 100. | Calculation works as intended, result displayed for user | Works as intended. |
| DH 4 | yearPass function calls relevant functions, takes inputted info and calculates outcomes | Input -100 into sell land, 2000 into feeding and 800 into sowing. Should take bushels to 0 + 100x land value, then add harvested bushels to create new total | Land value is 20, should get 2000+ sowed land bushels. Land should be 900, population starved should be 0, population should increase with influx | 2800 bushels – 800 farmed, 2000 from selling land, with 0 being consumed. Population went up by 3, land went down to 900 | Works as intended, pictureboxs and graph updated correctly also |
| Dh 5 | Random functions work as intended – plague(10%), bushels consumed(% of stock), influx(1-10), Price (17-26) | Pass 3 years of the game as usual, check outcomes are as expected. Use stops to check function values | Population should go up year on year between 1-10, function should show a random number between given values | Random functions work on given parameters, results are shown on main form | Works as intended. |
| DH 6 | Starved population (20 per person) functions as intended on yearPass | yearPass with 0 bushels, 1000, and 2000 on a new game | 0 – message box shows game loss  1000 – 50 people starve  2000 – 0 starvations | 0 – 100 starve  1000 – 50 starve  2000 – 0 starve  1500 – 25 starve | Works as intended |
| DH 7 | keepCount function stores data of variables year on year | Use a stop to check the value held in the 2d array in the data class | Data in the array per year should reflect the variable value at the point it was captured | Stops show value in array to be true to the value at the time of write | Works as intended |
| DH 8 | Final score is calculated correctly based on given values | Play a full game, use stops to check the variables before final score calculated, check result | Final score should come out correct to what has been calculated based on captured values | PopPerAcre – 989  Pop starved - 0 (+100 to score)  Popl.starved – 0  Final score – 1089 | Works as intended |
| DH 9 | Data held to display when a picture box is clicked is accurate | Pass a year, click a picture box, check held data vs expected | Data held should reflect expected values | Data held reflects expected values based on calculations | Works as intended |

Speed Tests

Speed of programme load, function calls, and other parameters tested.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test number | Test | Method | Expected | Actual | Notes |
| ST 1 | Programme loading test speed | Load the installed programme from desktop, time load | N/A, as long as software boots within a reasonable time (sub 10 seconds) | <1 sec load up time on debug, sub 2 seconds on published software | Acceptable |
| ST 2 | Programme calculating test speed | Perform a yearPass call by clicking ‘Make it so’, time how long it takes to call | N/A, max 5 second time expected | <1 Sec to perform function call on both debug and published application | Acceptable |
| ST 3 | Programme doesn’t stutter during normal play | Pass a normal game with 10 years, ensure no transaction goes beyond acceptable threshold | No method call should take beyond 5 seconds, on debug and published | <5 secs per year pass | Acceptable |

File Operation tests

Tests the programme can open, read, and save to external files.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test number | Test | Method | Expected | Actual | Notes |
| FO 1 | Tests the software can fine and read the ‘Scores.txt’ file on loading | Run the programme on both debug and install, modify the file in between to ensure read is correct | Programme reads the contents of the file. | Programme reads file on load | Works as intended |
| FO 2 | Tests the software can write to the ‘Scores.txt’ file, then re-read next cycle. | Run the programme to the end, check the value of final score, re-open programme to check value is read. | Programme writes the final score to new line of file, on open score is read | Score is written to file, and re-reads on next game load | Works as intended |
| FO 3 | Test programme icon is saved for both task bar and programme window | Run programme on debug, install and on new machine, ensure logo is present in all tests | Icon is present on all tests | Icon is present on all tests | Works as indended |

Testing references –

(1) <http://softwaretestingfundamentals.com/unit-testing/>