System testing

Test Plan

Below is an adapted test plan of the one found in my design section which has been adapted to accommodate the changes that I have made during implementation and the aspects of my program that I missed out to test during the design:

|  |  |
| --- | --- |
| Test Series | Purpose of test series |
| 1 | Test the flow of control: Does the user interface work as it should? Does it take users to the correct form or option? (top-down testing) |
| 2 | Validation of input data performed correctly(bottom-up testing) |
| 3 | Iterations, decisions, calculations, searches and sorts performed correctly.(white box testing and desk checking) |
| 4 | Data is saved into the correct files.(system testing) |
| 5 | The system produces the required results as per the specification.(black box testing) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test series and number | Purpose | Description | Test data | Expected result | Actual Result | Evidence in system testing appendix |
| 1.1 | To check to see if the menu choices in the main menu take the user to the desired form.(top- down testing) | The menu choices must take the user to where they want to go, so that it is easier for the user to navigate and understand. | Button pressed = Quit.  (Typical Data)  Button pressed = Teacher’s Section.(Typical Data)  Kruskal’s algorithm  combo box item chosen = Run.(Typical Data)  Kruskal’s algorithm  combo box item chosen = Timed run.(Typical Data)  Kruskal’s algorithm combo box item chosen = Times Achieved.(Typical Data) | The program should end.  The program should show the teacher’s login page so that one can gain access to the teacher’s section.  Displays the form containing the run page for Kruskal’s algorithm.  Displays the form containing the timed run page for Kruskal’s algorithm.  Displays the form containing the times achieved page for Kruskal’s algorithm. | As expected  As expected  As expected  As expected  As expected | Fig.1.1a  Fig.1.1b  Fig.1.1c  Fig.1.1d  Fig.1.1e |
| 1.2 | To check to see if the menu choices in the teacher’s menu take the user to the desired form.(top-down testing) | The menu choices must take the teacher to where they want to go, so that it is easier for the teacher to navigate and understand. | Button pressed = Kruskal’s algorithm.(Typical Data)  Button pressed = Quit to Main Menu.(Typical Data) | The program should take the teacher to the page where the times achieved by students in timed runs of Kruskal’s algorithm are displayed and are able to be adapted.  The program should display the main menu. | As expected  As expected | Fig.1.2a  Fig.1.2b |
| 1.3 | To check to see if the other “Quit” buttons take the user back to the correct form.(top-down testing) | This is so that if the user wants to take a break or do something else in the program at any time then they can. (Except in the forename entry page because a forename must be entered.) | Button pressed = Quit to Main Menu on the page Kruskal’s: Run.(Typical Data)  Button pressed = Quit to Main Menu on the page Kruskal’s: Timed Run. (Typical Data)  Button pressed = Quit to Main Menu on the page Kruskal’s: Times Achieved. (Typical Data)  Button pressed = Quit to Main Menu on the page Teacher’s section: Login. (Typical Data)  Button pressed = Quit to Main Menu on the page Teacher’s section: Menu. (Typical Data)  Button pressed = Quit to Teacher’s Menu on the page Teacher’s section: Kruskal’s. (Typical Data) | The program should display the Main Menu.  The program should display the Main Menu.  The program should display the Main Menu.  The program should display the Main Menu.  The program should display the Main Menu.  The program should display the Teacher’s Menu. | As expected  As expected  As expected  As expected  As expected  As expected |  |
| 2.1 | Validate the forename that is entered by the user. I will be testing this by seeing if an error message or an acceptance message is displayed.(bottom-up testing) | The program should only not accept forenames if they contain no characters or contain more than 50 characters. An error message should be displayed in these two cases. An acceptance message should appear following anything else entered. | John(Typical Data)  1(Typical Data)  John Every(Typical data)  John.Every(Typical Data)  Enters no characters.(Erroneous Data)  Enter a name that is 50 characters long.(Boundary Data)  Enter a name that is 51 characters long.(Boundary Data, above)  Enter a name that is 49 characters long.(Boundary Data, below) | Accepted  Accepted  Accepted  Accepted  Error  Accepted  Error  Accepted | Accepted(acceptance message appeared)  Accepted(acceptance message appeared)  Accepted(acceptance message appeared)  Accepted(acceptance message appeared)  Error(error message appeared)  Accepted  Error  Accepted  (However the test for both 50 and 49 characters caused the time achieved to move out of line with the rest of the data because the names were too long.) | Fig.2.1a  Fig.2.1b  Fig.2.1c  Fig.2.1d  Fig.2.1e  Fig.2.1f  Fig.2.1f |
| 2.2 | To confirm the identity of the user being a teacher by means of validating a password entered by them.(bottom-up testing) | The program should only allow teachers to access the teacher’s section by means of entering a password and it matching the password contained in the code. The password is entered after the user has pressed the “OK” button. | Enters no characters  (Erroneous Data)  User enters incorrect password.(“Newton”)  (Erroneous Data)  User enters correct password.(“Euler”)  (Typical Data) | Error (error message will appear)  Error(error message will appear)  Accepted (proceeds to the teacher’s menu.) | Error(Error message appeared)  Error(Error message appeared)  Accepted(displayed Teacher’s Menu) | Fig.2.2a  Fig.2.2b  Fig.2.2c |
| 3.1 | To check if the arc weights produced by the program are correct and that all the integers in the boundary are produced by the system once or more in one or more timed runs.(white box testing) | The arc weights for each arc of the network are produced from random numbers and then multiplied by 20. The arc weights must be integers and in the range of 0<arc weights<=20. | 5(Typical Data)  1(Typical Data)  0(Boundary Data, on)  -1(Boundary Data, below)  1(Boundary Data, above)  19(Boundary Data, below)  20(Boundary Data, on)  21(Boundary Data, above  1.1(Erroneous Data)  4,4(Typical Data) | Was produced  Was produced  Wasn’t produced  Wasn’t produced  Was produced  Was produced  Was produced  Wasn’t produced  Wasn’t produced  Was produced | As expected  As expected  As expected  As expected  As expected  As expected  As expected  As expected  As expected  As expected |  |
| 3.2 | To check that the minimum connector produced by the run of Kruskal’s algorithm is correct.(white box testing) | This is to show that my program performs Kruskal’s algorithm on a network correctly for the run. | The minimum connector produced by the run. Because the network and arc weights for the run are the same as one of the examples in the D1 revision guide. Therefore if the minimum connector produced by the run is the same as that in the revision guide then the algorithm is being performed correctly and has produced the correct result.(Typical Data) | Produces correct result. | Correct result was produced as it matched the solution to the corresponding example in the revision guide. |  |
| 3.3 | To check that the minimum connector produced by timed runs of Kruskal’s algorithm are correct.(white box testing) | This is to show that my program performs Kruskal’s algorithm on a network correctly for timed runs. | The minimum connector produced by the program. This is because the program produces the solution to the question by colouring arcs in the network. Therefore if I was to perform the algorithm on the network on paper and then check that the minimum connector produced by me on paper matches that produced by the program then it would be correct.(Typical Data) | Produces correct solution. | Produced correct solution. However the method was slightly different; this is explained in my System testing appendix. | Fig.3.3a |
| 4.1 | The data is saved into the correct binary random file correctly.(system testing) | Once the acceptance message has appeared the student’s forenames and times achieved are saved into the “Leaderboard.dat” file for timed runs. These need to be saved each time after some one completes a timed run. Where they need to be loaded and displayed in other forms. In both the update times for Kruskal’s algorithm and the Times achieved forms the data is sorted into order using the bubble sort; where a rank is generated for each student. This is done every time these forms are loaded and the results are saved to the file. | Forename, time achieved and generated rank. I will be using the forename “Sebastian” and the rank and time achieved will be determined by the program. However I don’t know what rank I am before I open the file. The time achieved turned out to be 0:9:1. This equals 0 minutes, 9 seconds and 1 millisecond.(Typical data) | Looking in the Kruskal’s times achieved page or the update times achieved page in the teacher’s section. The forename is there the same as it was entered. Along with the time achieved and the correct generated rank. | On both forms the correct record was displayed in the list box where both the forename and time achieved matched what was shown on the forename entry page. It also produced a generated rank which was as it should be this being 25. Therefore the bubble sort has worked correctly. | Fig.4.1a |
| 4.2 | The changes made to the data in the binary random file by a teacher are saved to the file correctly.(system testing) | In the Teacher’s section: Kruskal’s page the teacher deletes records form the leader board by selecting records and pressing the delete button. To save these changes the user has to press the save button once the delete button has been pressed. Where the program should then input or save only the records that weren’t deleted by the teacher. | I have saved records in the file using the forename entry page and then deleted two of these saved records and pressed the save button to make the changes permanent. (Deleted records = have forenames Ash and Sebastian)(Typical data) | In the Times Achieved page and the Teacher’s section: Kruskal’s all of the records should be shown in the correct order and only the two that were deleted shouldn’t be shown. | As expected |  |
| 5.1 | Timer works correctly, so that it stops at 9:59:99.(black box testing) | The program should only time up until the timer reads 9:59:99 and it must not go beyond this. Once it has reached this time it must stop and tell them that they have failed to complete a timed run in under the required time. Also if the user completes a timed run of an algorithm then the timer must stop and the program must ask the user to enter their forename. | Letting the timer run until 9:59:99.(Boundary data, on)  The user completes the timed run of an algorithm between 1 and 2 minutes.(Typical Data)  Letting the timer run until 3 minutes. (Typical Data)  Presses the “Quit to Main Menu” button at any time.(Typical Data) | The program stops and then displays a message box that states they have failed to complete the timed run in the required time. Then the user is taken back to the main menu.  Stops and takes the user to the forename entry page.  The timer will continue.  Stops, doesn’t ask for the user’s forename and returns the user to the main menu. | As expected  As expected  As expected  As expected | Fig.5.1a  Fig.5.1b |
| 5.2 | To check to see if the program recognises when the correct arc is chosen. (black box testing) | When the user clicks on the correct arc the algorithm moves onto the next stage and displays an acceptance message, thereby allowing the user to complete the timed run and recognise that they clicked on the correct arc. Also if the wrong one is chosen then an error message is displayed that tells the user that it was the wrong one. | Correct arc is chosen by the user.(Typical Data)  The incorrect arc is chosen by the user.(Typical Data) | Accepted (the correct message is displayed.)  Error (the incorrect message is displayed in a label.) | As expected  As expected |  |
| 5.3 | This is to check to see if the program produces an error message and stops saving records in the file once the number of records goes beyond 100.(black box testing) | If the user tries to save a record in the file which will cause the number of records in it to exceed 100 the program must not save any more records in the file and therefore display an error message which states that they must contact my end user who can delete records from the file using the program. This should not affect the opening of both the Times Achieved page and the Teacher’s: section: Kruskal’s page. | Save 99 records to the file.(Boundary Data, below)  Save 100 records to the file.(Boundary Data, on)  Try to save 101 records into the file. (Boundary Data, above) | The program saves all of the records to the file without any problems.  The program saves all of the records to the file without any problems.  The program should produce a message box which states that the maximum records for the file has been reached and that the user must contact the end user to save more files. This is produced after the user has entered a forename and pressed “OK”, where in this case the record is not saved to the file. After this the user should be taken back to the Main Menu and the user and subsequent users should be stopped from saving more records to the file until the problem has been resolved by the end user. This shouldn’t affect the opening of both the Times Achieved page and the Teacher’s section: Kruskal’s page, so that users can still see the current leader board and teachers can make changes to the leader board. | As expected  As expected  As expected, however to make this happen as expected the code needed to do this created a blank record at the bottom of the leader board on both the Times Achieved page and the Teacher’s section: Kruskal’s page. However the blank record was not saved to the file. This doesn’t affect anything else in my program. |  |
| 5.4 | This is to check that the delete button on the Teacher’s section: Kruskal’s page only deletes the record selected from the list box and not the file itself. (black box testing) | The “Delete” button must only delete the selected record from the list box because the “Save” button makes this change permanent. Also if the teacher makes a mistake then at least the mistakes have not been saved. This deleted item must be deleted from the list box and then return when the same page is opened again if the “Save” button is not pressed. | Delete the record which has forename “John” using the delete button. (Typical Data)  No item is selected.(Typical Data) | When I go back into the same page to look for the record using the “Quit to Main Menu” button the record is still there and hasn’t been deleted permanently.  Nothing happens | As expected  Nothing happened |  |
| 5.5 | This is to check that the “Save” button on the page Teacher’s section: Kruskal’s saves the changes made to the leader board by the end user or another Maths teacher.(black box testing) | The “Save” button must only save changes to the file once the delete button has been pressed and records have been deleted from the list box. | The delete button is pressed and records are deleted from the list box. The save button is then pressed.(Typical Data)  The save button is pressed when nothing has been deleted/ selected.(Typical Data) | When going back into the Times Achieved page or the Teacher’s section: Kruskal’s page the records that were deleted permanently are not displayed in both forms.  Nothing happens | As expected  Nothing happened |  |
| 5.6 | This is to check that when the help button is pressed in both the timed run page and the run page that the list box is displayed or hidden depending on whether it is already displayed.(black box testing) | When the user presses the help button the first time the list box should appear if the user presses the button again then the list box should be hidden on both the timed run page and the run page. | The list box is not currently displayed and the help button is pressed on the run page.(Typical Data)  The list box is currently displayed and the help button is pressed on the run page.(Typical Data)  The list box is not currently displayed and the help button is pressed on the timed run page.(Typical Data)  The list box is currently displayed and the help button is pressed on the timed run page.(Typical Data) | The list box is displayed.  The list box is hidden.  The list box is displayed.  The list box is hidden. | The list box was displayed.  The list box was hidden.  The list box was displayed.  The list box was hidden. |  |

I am also going to have some beta testing performed on my program; this will be done by students who have completed the D1 module in my Further Mathematics class and by my end user.

They will be asked to test whether my program works as it should by testing for errors (this will be done more by my end user), seeing if my program does the job it was designed to do this being to teach students D1 and finally to test whether I have met the objectives that I set and agreed upon with my end user in my analysis section. The results of this will be shown and analysed in my appraisal.