

Wonder Walls





## Centre Number - 71307

## Candidate Number - 4141

Contents

[***Background*** 2](#_Toc5793030)

[***Target Audience – Staff and Competitors*** 2](#_Toc5793031)

[***User requirements*** 3](#_Toc5793032)

[***Data Requirements*** 5](#_Toc5793033)

- [***Inputs*** 7](#_Toc5793034)

- [***Storage*** 7](#_Toc5793035)

- [***Outputs*** 8](#_Toc5793036)

- [***Reports*** 8](#_Toc5793037)

[***Navigation Drawings*** 9](#_Toc5793038)

- [***Draft Drawings*** 9](#_Toc5793039)

- [***Digital Navigation Drawings*** 12](#_Toc5793040)

[***Designing Algorithms*** 15](#_Toc5793041)

- [***Success Criteria*** 15](#_Toc5793042)

- [***Logging on:*** 15](#_Toc5793043)

- [***Menu System:*** 16](#_Toc5793044)

- [***Assigning An Auto Number:*** 16](#_Toc5793045)

- [***Add Competitor:*** 16](#_Toc5793046)

- [***Reaction Time Tests:*** 17](#_Toc5793047)

[***Error Trapping Proposals*** 20](#_Toc5793048)

- [***Logging On*** 20](#_Toc5793049)

- [***First Name Validation*** 21](#_Toc5793050)

- [***Postcode Validation*** 22](#_Toc5793051)

[***Testing The Program*** 24](#_Toc5793052)

- [***Corrective Testing*** 31](#_Toc5793053)

[***Evaluation*** 33](#_Toc5793054)

- [***User requirements:*** 33](#_Toc5793055)

- [***Performance during testing:*** 34](#_Toc5793056)

- [***Refinements following testing:*** 34](#_Toc5793057)

- [***Robustness of the system:*** 34](#_Toc5793058)

# ***Background***

Wonder Walls is a new climbing range in Northern Ireland which provides a variety of climbing and building scaling opportunities such as the Eiffel Tower because many of their new Banbridge centre’s climbing walls are based around such structures. Climbers can also climb skyscrapers and mountain faces in their new centre as they have many themed walls available for the customer to climb and challenge themselves on. They offer facilities for parties, private bookings, work parties and group based activities. They have four party rooms, their own restaurant and a range of themed climbing walls. Wonder Walls charges £15 per person in parties and charges £25 for lessons. Wonder Walls are running a competition for climbers aged 14-16. each competitor must take two climbs and a reaction time assessment. Results are then published on the Wonder Walls website two hours after the climbs. Wonder Walls needs to have a system that enables Wonder Walls staff to access the system through an introductory logon screen in which they can record competitor details and an assigned number to which the competitor can be easily found, record performance details for each competitor’s climbs and preform a reaction time test. They also need facilities to produce reports on competitor details.

# ***Target Audience – Staff and Competitors***

The target audience for this program is the wonder walls staff who require certain facilities to effectively run their new competition. The Wonder Walls staff has asked for a system that will store various competitor details which are required to run the competition such as their name and their climb times throughout the competition this is to make data input easy and effective for them and will therefore allow staff to enter in competitor details and performance data for 3 climbs and a reaction time test which competitors have to take. The system will be made so that it is easily accessed by a member of the wonder walls staff and that they can quickly navigate to the desired operation within the system. The system will only be accessible to member of the Wonder Walls staff via a username and password tailored towards them. Competitors preform a reaction time test and climbs and then the Wonder Walls staff are required to enter in these results into the system, this allows all competitor data to be easily accessed by a member of staff to then be uploaded onto the website 2 hours after a competitor’s completed attempt. Although the system is primarily targeted towards staff and is made in a way that will make it effective for them to input and record data, the system is also targeted towards the competitors who as a part of the competition are required to perform a reaction time test which involves three visual tests such as pressing enter when a word appears on screen. The system must be tailored in a way to accommodate this and make it very simple for a competitor with the supervision of a member of staff to take this test. The system will store this data automatically allowing the process to be sped up for both the competitor and the member of staff as the competitor will not have to worry about what to do as there will be given clear instructions by the system and all it means that the member of staff does not have to waste time inputting the data as the system automatically does that. The system will also provide clear help and instructions to the member of staff so that they know how to use the system and input the data correctly therefore managing competitor details. The System will be made to accommodate the needs of the Wonder Walls staff and ensure that they can easily and effectively input data into the system which will therefore aide them in running their competition. The competitors are aged from 14-16 and these entries are then processed and recorded by the Wonder Walls staff and in order to effectively run the competition they require a platform on which to record these entries and related data. The system that will be made will ensure that this process is made simple for the staff and will correctly check for errors in the data and notify the member of staff who can then easily and quickly change the data. The system will also be tailored towards the young competitors and will be designed in a way in which they can simply complete a reaction time test as the system walks it through it. The audiences for this system will be closely looked at in order to make an exact and fully functional program for them to use effectively and easily

# ***User requirements***

The Wonder Walls staff require the program to have a various range of functions and options which they can use to input data into the system correctly. Including the program Wonder Walls also require a logo which can be seen at the start of the documentation. The main program must contain the following functions:

1. Firstly, the staff require an initial log on screen which will ask if they would like to either log onto the system or shut it down. When the user enters log on the system then prompts them to enter a username and password which is required to gain access to the system. The staff can request to have a set username and password such as WonderWalls and Admin1. The log on screen will also prevent the user from inputting an incorrect username and password a number of times and after three failed attempts the system will automatically shut off therefore preventing a person who is not a staff member from gaining access to the system.
2. Once the user has successfully logged onto the system they require a menu which will outline the range of functions that are available to them such as adding a competitor or completing a reaction time test. The user is then required to enter in a valid option to be taken to the desired process. If the user inputs an invalid entry, then the program will notify the user that this is correct and the program will prompt the user to enter in a valid option from the list shown to the user. The menu will also provide the user with an option to log off and return to the initial introductory screen. The staff at Wonder Walls require a number of functions to be available to them through the menu system, these include the following:
3. A function in which a competitor’s details can be entered by the member of staff. This function if chosen from the menu will prompt the user to enter in a competitor’s details such as a name, postcode and telephone number. The program will then automatically assign a competitor number to this person – this number will include both letters and characters and would look like this “WW1” As well as auto assigning a number to a competitor the system will also validate all data that is entered in as competitor details – this will ensure that for example the telephone number has the correct number of digits and the postcode is the right pattern of letters and numbers. This therefore limits the user’s time checking for errors in details as the program does it for them. A competitor can also be deleted from an option in the menu if the user so desires. The program then asks if the user wants to add another competitor – if so this process restarts. If not the program returns the user back to the menu system.
4. A function in which a competitor can undergo 3 separate visual reaction time tests. When the user selects this function from the main menu they can take 3 different tests. The user first has to enter in their name and competitor number for the system to find the correct person. The 3 tests are then taken.

* The first test involves the user pressing the enter key when a word is shown on screen. This test is then repeated 3 times and an average is taken from the three sets of results. This is shown to the user before it being inputted into a csv file with the rest of the competitor’s data.
* The second test will tell the user to press enter when a specific number appears on screen, the program then will show various numbers until the correct one is shown, their time is then recorded from the moment the number appears on screen. Like the other test this is repeated and averaged before being saved by the program along with the rest of the competitor’s data.
* The last of these tests requires the competitor to press a number of characters as fast as possible such as “a, t. h, m” Likewise this is then repeated, averaged and then recorded along with all the other competitor’s data. The program then returns the user to the main menu once this process has been repeated

1. A function in which the user can enter in competitor results for the 3 climbs. When chosen the user will be prompted to enter in the competitor’s assigned number and name so that the system can record the data to the right person. The user then enters the results of the three climbs separately as guided by the system. The system the validates this to ensure that the values entered are correct and that there will not be an error in their details. If incorrect the program will notify the user and prompt them to re-enter the data ensuring that it is correct. Once completed the program will ask whether the user wants to enter in another competitor’s scores before returning to the main menu.
2. A sub-menu where the staff members can create a number of reports based on competitor data. The sub menu will offer 4 main types of reports that the user can chose from.

* The first of which is a report on a single competitor’s data. This report will show their name and number and will also provide a summary of their results which includes the 3 reaction time tests which have been taken, the results of the 3 climbs which have been inputted into the system and an overall place in the competition.
* The Second report shows the whole file in order of competitor number starting at “WW1”
* The third and fourth report will be leader boards which each will be solely based on the first reaction time test and first climbs individually whole. These reports can be saved for later use by the user. This sub-menu will be easily exited back to the main menu by the user.

# ***Data Requirements***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Data Item** | **Data Type** | **Validation** | **Sample data** | **Function of data item** |
| Logging on | | | | | |
| 1 | Username | String | = to value | WonderWalls | Stores the staff members username to access the system |
| 2 | Password | String | = to value | Admin1 | Stores the staff members password to access the system |
| Menu System | | | | | |
| 3 | Chose | integer | Is a digit and a valid choice | 1 | Allows member of staff to access different functions by entering in the function number |
| Adding A Competitor | | | | | |
| 4 | CompetitorNumber | String | N/A | WW3 | Stores a unique set of letters and numbers to easily identify competitors |
| 5 | FirstName | String | Is made of letters | James | Stores a competitor’s first name for use in identifying them or in reports |
| 6 | LastName | string | Is made of characters | Ross | Stores a competitor’s last name for use in identifying them or in reports |
| 7 | Postcode | string | The first 2 and last 2 characters are letters and the rest are digits | BT422DX | Stores a competitor’s postcode into the csv file which can be used as personal contact information |
| 8 | PhoneNumber | integer | Contains only digits and is a certain length | 07946119013 | Stores a competitor’s phone number into the csv file which can be used as personal contact information |
| 9 | Email | String | Has the correct address endings | Jrross@Hotmail.com | Stores a competitor’s email address into the csv file which can be used as personal contact information |
| Reaction Time Tests | | | | | |
| 10 | Test 1 | float | N/A | 0.121 | Stores the first of a competitor’s 3 reaction time test to a csv file and can be used later in reports |
| 11 | Test 2 | float | N/A | 0.333 | Stores the second of a competitor’s 3 reaction time test to a csv file and can be used later in reports |
| 12 | Test 3 | float | N/A | 0.543 | Stores the third of a competitor’s 3 reaction time test to a csv file and can be used later in reports |
| Climb Times | | | | | |
| 13 | Climb 1 | float | Is a digit | 12.1 | Stores the first of a competitor’s 3 climb times to a csv file and can be used later in reports |
| 14 | Climb 2 | float | Is a digit | 23.5 | Stores the second of a competitor’s 3 climb times to a csv file and can be used later in reports |
| 15 | Climb 3 | float | Is a digit | 50.2 | Stores the third of a competitor’s 3 climb times to a csv file and can be used later in reports |

# ***Inputs***

The user will be required to enter in a number of competitor details such as their first and last names and their postcode. This will be done by the system prompting them to input a certain piece of data which the will have space below to enter it in. The user will also have to input their username and password when logging onto the system which the system will also prompt them to enter in the data. 3 different reaction times will also be under took by the competitor these will consist of 3 short tests in which the user will have to press a certain letter or number. The user will be required to complete this as fast as possible and their average score of 3 attempts per test will be shown to the competitor before being stored.

# ***Storage***

All competitor details will be stored into a single csv file by the system. This allows the system to use this information stored in the file to generate reports based on that information and identify and use an individual competitor’s details. The information that is stored in the csv file is the competitor’s:

* First Name
* Last Name
* Competitor Number
* Postcode
* Phone Number
* Email Address
* 3 Separate Climb Times

The competitor will also have their 3 separate reaction time scores recorded in the csv file along with the rest of their details. The 3 times are as follows:

* Press Enter When A Word Appears
* Press Enter When A Particular Number Appears In A Sequence
* Enter In A Series Of Characters

# ***Outputs***

# ***Reports***

The system will also produce a summary report of each competitor’s data which consists of a summary of their name, competitor number and results from both their reaction time tests and climb times.

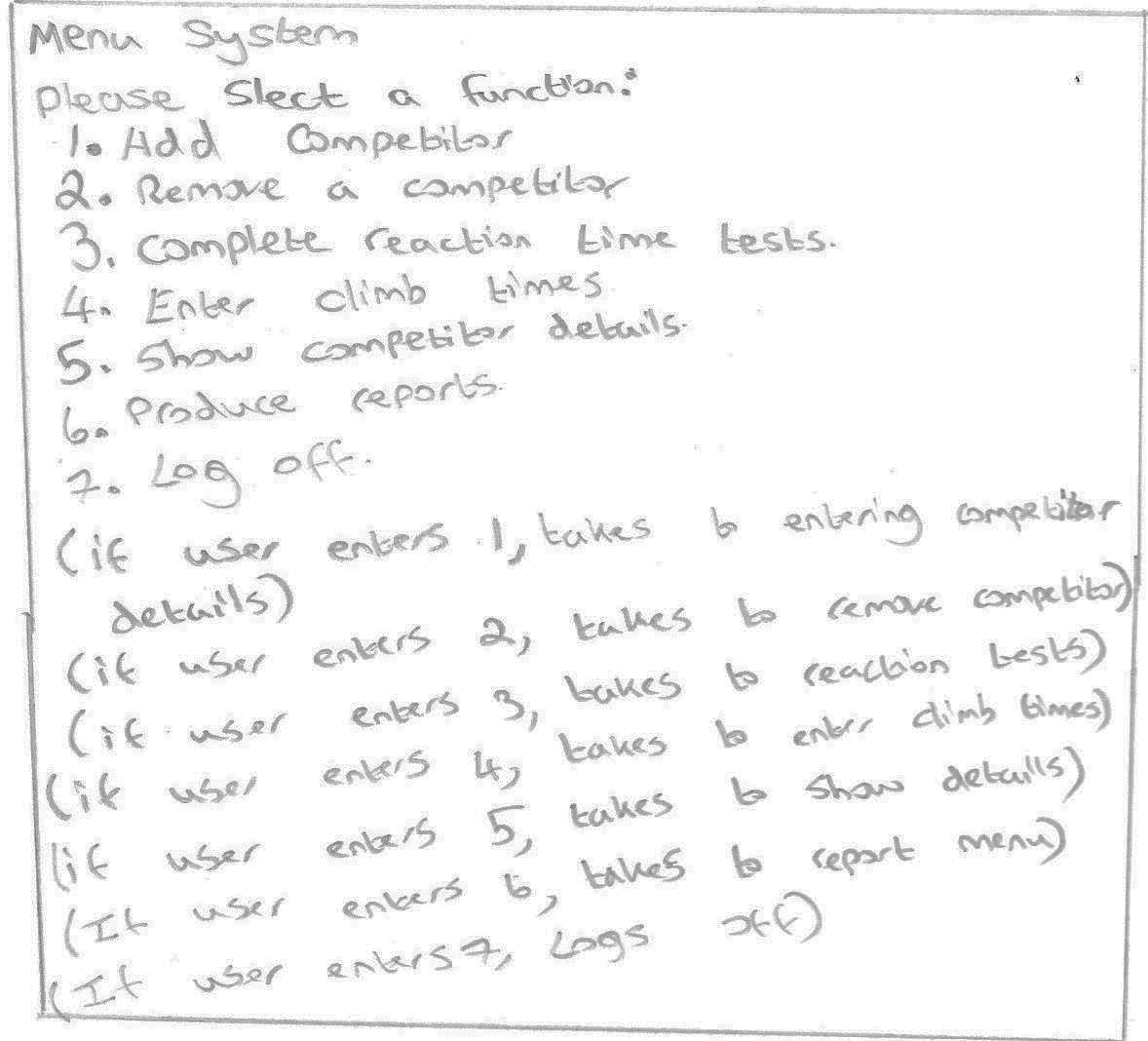
Another report is a leader board produced from a score from both the reaction times and the climb times. This is an overall score and the top will be the person with the highest and the bottom will be the person with the lowest.

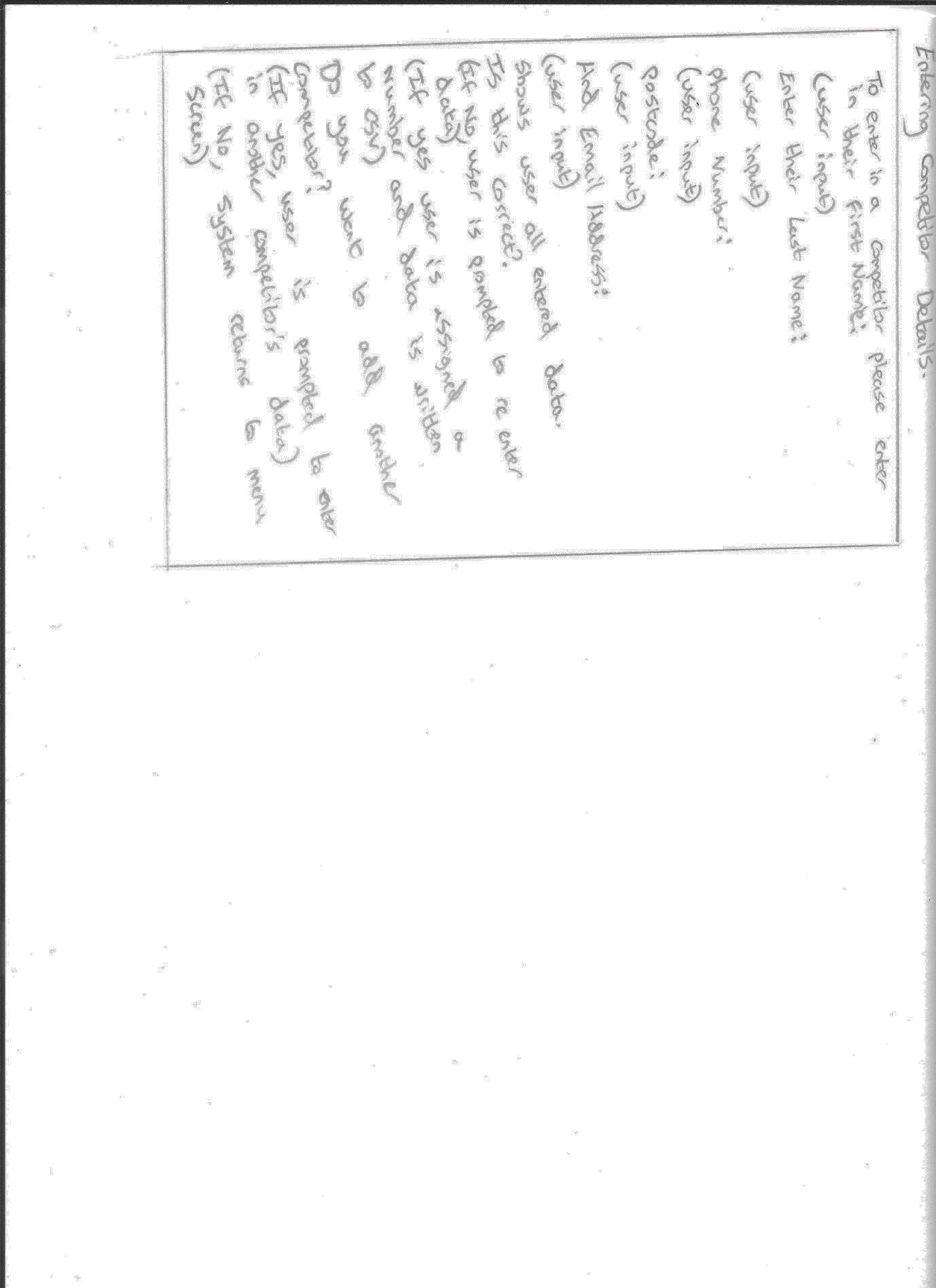
Another report will create a leader board that is based solely on the results of the reaction time tests and will therefore be a reaction test leader board

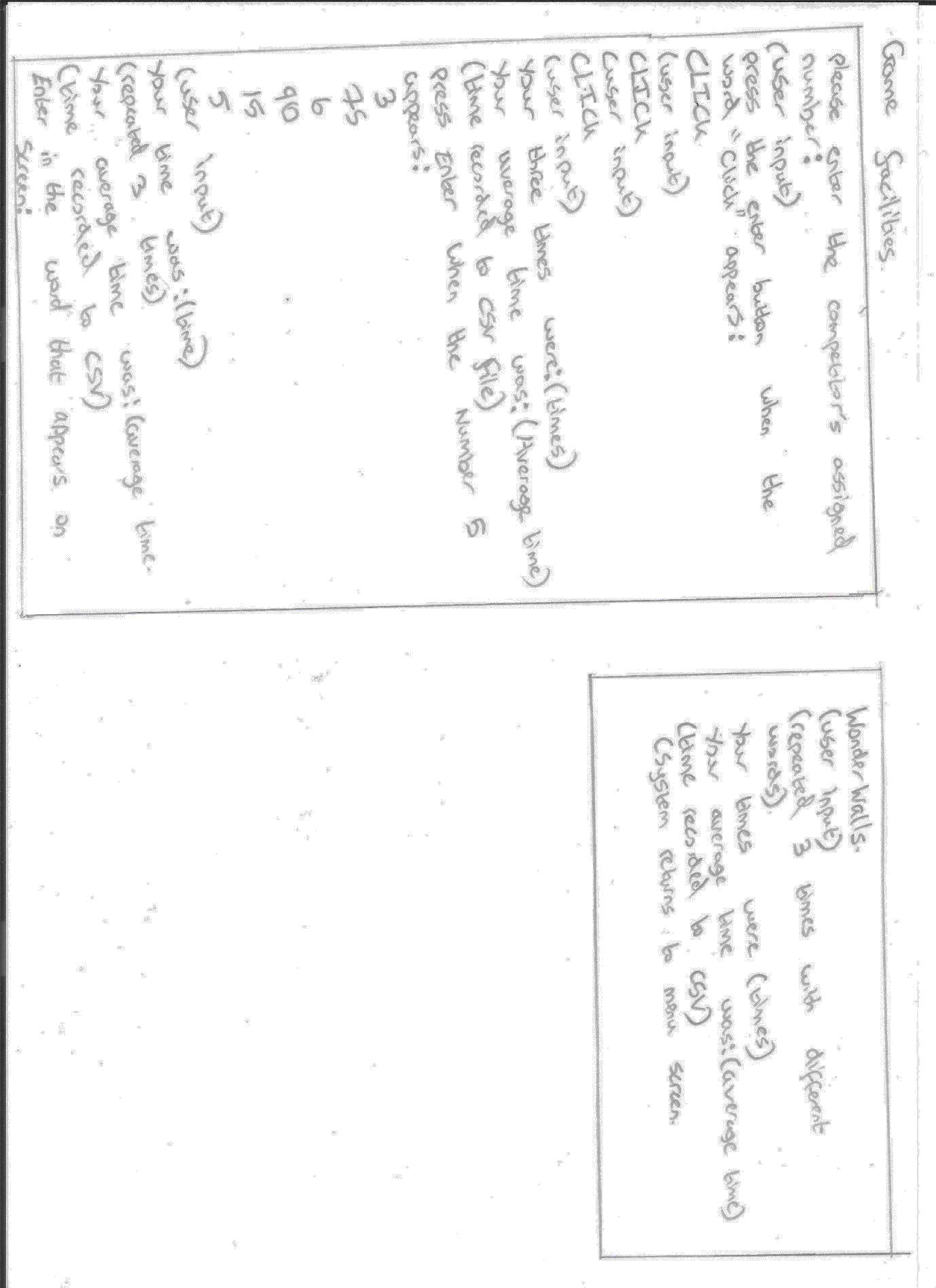
Similarly, the final report will be based on the results of the climbs and will not include the reaction time test. This allows competitors to see where they place in individual sections of the competition.

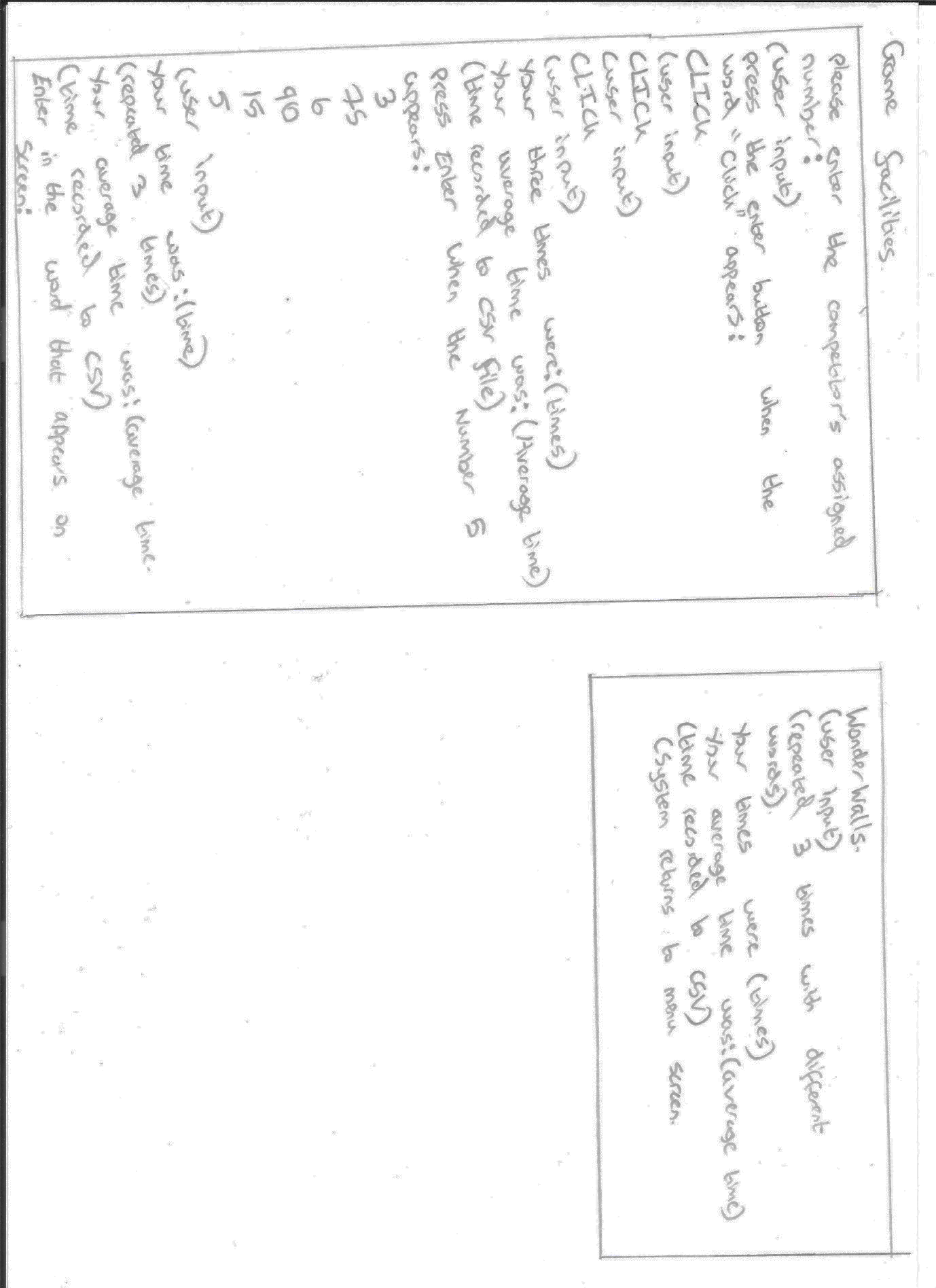
# ***Navigation Drawings***

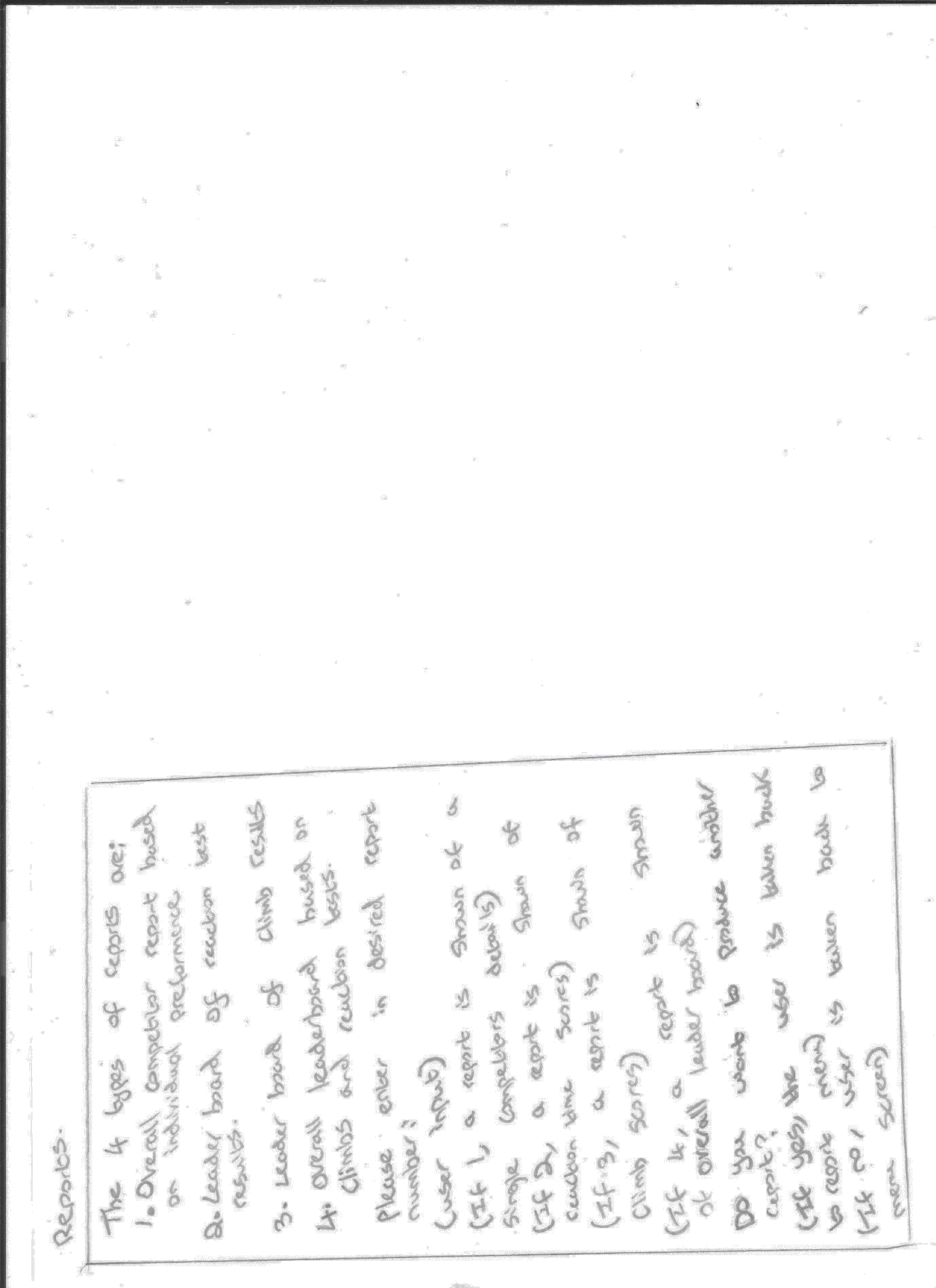
# ***Draft Drawings***











# ***Digital Navigation Drawings***

Menu Screen:

Please select a function:

1. Add A Competitor
2. Remove A Competitor
3. Complete Reaction Time Tests
4. Enter Climb Times
5. Show Competitor Details
6. Produce Reports
7. Log Off

(If user enters 1, takes user to Entering Competitor Details)

(if user enters 2, takes user to Removing Details)

(if user enters 3, takes user to Reaction Tests)

(if user enters 4, takes user to Entering in Climbs)

(if user enters 5, takes user to Showing Competitor Details)

(if user enters 6, takes user to Report Menu)

(if user enters 7, takes user back to Logging off)

Logging in:

Do you want to log in or shut down?

(User Input)

(if user inputs shut down)

Shutting down system…

(End program)

(if user inputs log in)

Username:

(User Input)

Password:

(if user inputs incorrect username or password, user is prompted to enter in Username and Password)

(If user incorrectly enters username or password 3 times)

You have entered in an incorrect username or password too many times

You have been locked out

(End Program)

(If correct username and password is entered)

(takes user to menu screen)

Entering Competitor Details:

To enter in a competitor, please enter in their first name:

(User input)

Enter their last name:

(User input)

Phone number:

(User input)

Postcode:

(User input)

And Email Address:

(User input)

Shows user all entered data

Is this correct?

(If no, the user is prompted to renter data)

(if yes, the user is assigned a number and data is stored to the CSV file)

Do you want to add another competitor?

(if yes, the user is prompted to enter in another competitor’s details)

(if no, system returns to menu screen)

Reaction Time Tests:

Please enter the competitor’s assigned number:

(User input)

Press the Enter button when the word “CLICK” appears:

CLICK

(User input)

CLICK

(User input)

CLICK

(User input)

Your three times were: (times)

Your average time was: (Average time)

(time recorded to CSV file)

Press enter when the Number 5 appears:

3

Reports:

The 4 types of reports are:

1. Overall competitor report based on individual performance
2. Leader board of reaction test results
3. Leader board of climb times
4. Overall leader board based on reaction and climb times

(if user enters 1, a report is shown of a single competitor’s details)

(if user enters 2, a report is shown of reaction time scores)

(if user enters 3, a report is shown of climb scores)

(if user enters 4, a report is shown of an overall leader board)

Do you want to produce another report?

(if yes, the user is taken back to the start of the report menu)

(if no, the user is taken back to the menu screen)

75

6

90

15

5

(User input)

Your time was: (time)

(repeated 3 times)

Your average time was: (Average time)

(Time recorded to CSV)

Enter In the word that appears on screen:

WonderWalls

(User input)

(repeated 3 times with different words)

Your times were: (times)

Your average time was: (Average time)

(time recorded to CSV)

(System returns to menu screen)

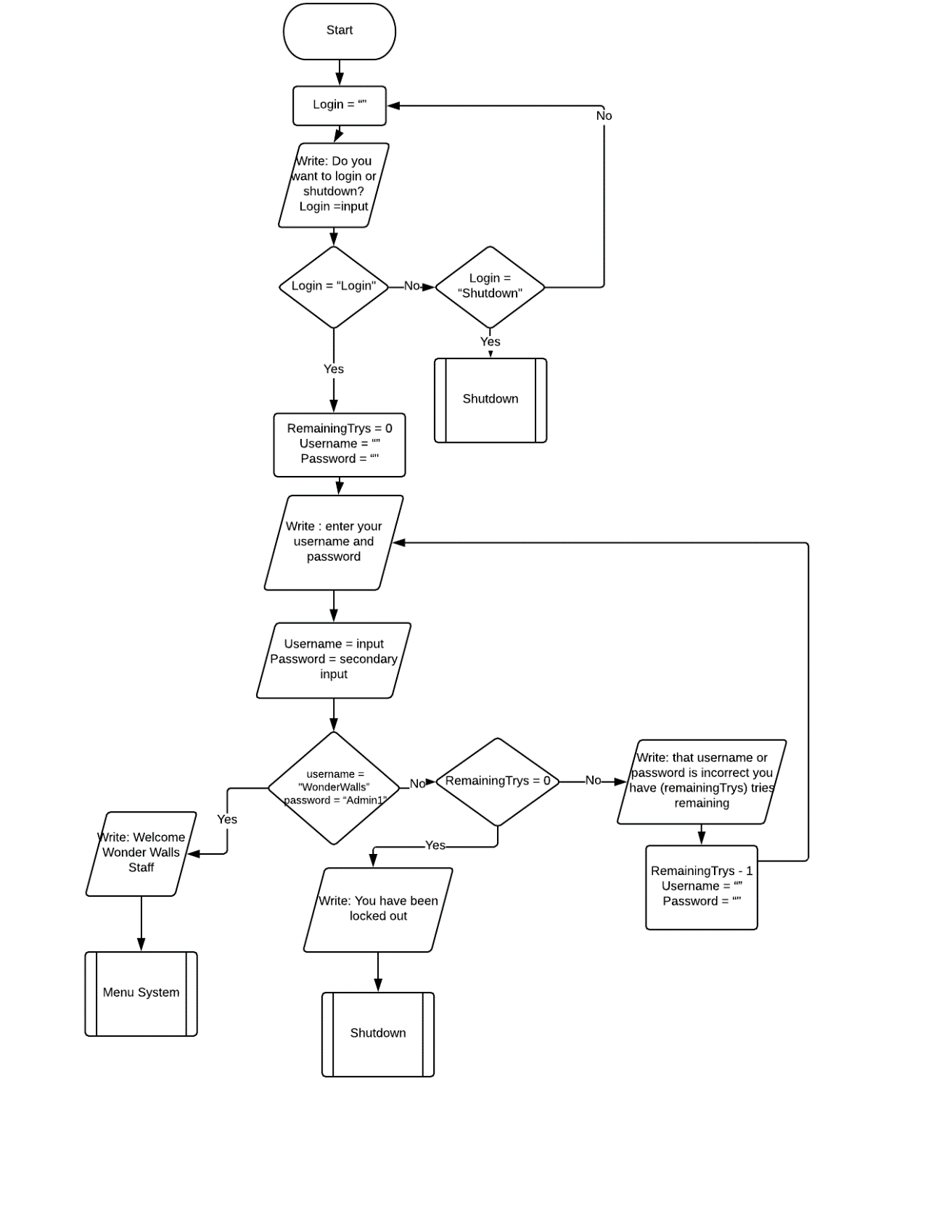
Some details of these plans have been changed throughout development of my program such as the reports which have changed.

# ***Designing Algorithms***

# ***Success Criteria***

1. Logging On
2. Menu System
3. Assigning An Auto Number
4. Add Competitor
5. Reaction Time Tests

# ***Logging on:***



# ***Menu System:***

# ***Assigning An Auto Number:***

file = open("Customer Details.csv", "r")

text = read file

length =read number of lines in file

customerNum = length

with open("Customer Details.csv", "a") as csvWrite:

csvWrite.write to file(CustomerNum)

# ***Add Competitor:***

firstName = input("To add a competitor please enter their first name: ")

lastName = input("please enter their Last name: ")

postcode = input("please enter their postcode: ")

email = input("please enter their email address: ")

phone = Input("please enter their phone number: ")

Details = [firstName, lastName, postcode. Email. Phone]

with open("Customer Details.csv", "a") as csvWrite:

csvWrite.write to file(Details)

# ***Reaction Time Tests:***

plays = 0

score1 = 0

score2 = 0

score3 = 0

while plays <= 2:

Write ("press enter when the word 'CLICK' appears:\n")

wait(random.randint(1,8))

start = time.time()

test = input("CLICK\n")

end = time.time()

total = end - start

Write("Your score was: " + total)

if plays == 0:

score1 = total

elif plays == 1:

score2 = total

elif plays == 2:

score3 = total

plays += 1

Write("Your three scores are:” + score1 + score2 + score3))

aver = average([score1,score2,score3])

finalScore1 = str(aver)

Write("Your final score and average is:", finalScore1)

plays = 0

score1 = 0

score2 = 0

score3 = 0

while plays <= 2:

correct = True

Write(“Type 66 when the number 66 when it appears appears:\n")

numbers = (random.randint(1,10))

numb = 0

while correct == True:

wait(random.randint(1,5))

Write(random.randint(1,65))

numb += 1

if numb >= numbers:

wait(random.randint(1,5))

start = time.time()

test = input("66\n")

if test == "66":

end = time.time()

total = end - start

Write("Your score was: " + total)

if plays == 0:

score1 = total

elif plays == 1:

score2 = total

elif plays == 2:

score3 = total

plays += 1

correct = False

else:

Write("that is not a correct entry the test will now repeat")

numb = 0

correct = True

Write("Your three scores are:" + score1 +score2 + score3)

aver = average ([score1,score2,score3])

finalScore2 = aver

Write("Your final score and average is:", finalScore2)

plays = 0

score1 = 0

score2 = 0

score3 = 0

correct = True

while plays <= 2 and correct == True:

Write("Enter in the letters 't i w w' when the word 'GO' appears:\n")

wait(random.randint(1,8))

start = time.time()

test = input("GO\n")

if test == "t i w w":

end = time.time()

total = end - start

Write("Your score was: " + total)

if plays == 0:

score1 = total

elif plays == 1:

score2 = total

elif plays == 2:

score3 = total

correct = False

plays += 1

else:

Write("This is not a valid entry and the test will now be reapeated")

correct = True

Write("Your three scores are: score1 + score2 +score3)

aver = average([score1,score2,score3])

finalScore3 = aver

Write("Your final score and average is:", finalScore3)

# ***Error Trapping Proposals***

# ***Logging On***

1. username = “”
2. password = ““
3. tries = 0
4. while (not username):
5. username = input("Username: ")
6. while (not password):
7. password = input("Password: ")
8. if trys <= 3 and username == "WonderWalls" and password == "Admin1":
9. write("Welcome Wonder Walls Staff")
10. Menu System
11. elif trys <= 3:
12. tries += 1
13. write("That Username or Password is incorrect. You have " 3-Tries "tries remaining")
14. username = ""
15. password = ""

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Line Number | Username | Password | tries | Input | Output |
| 1 | “” |  |  |  |  |
| 2 | “” | “” |  |  |  |
| 3 | “” | “” | 0 |  |  |
| 4 | “” | “” | 0 |  |  |
| 5 | WonderWalls | “” | 0 | WonderWalls | Username: |
| 6 | WonderWalls | “” | 0 |  |  |
| 7 | WonderWalls | Admin1 | 0 | Admin1 | Password: |
| 8 | WonderWalls | Admin1 | 0 | Admin1 |  |
| 9 | WonderWalls | Admin1 | 0 |  | Welcome Wonder Walls Staff |
| 10 | WonderWalls | Admin1 | 0 |  | Menu System |
| 1 | “” |  |  |  |  |
| 2 | “” | “” |  |  |  |
| 3 | “” | “” | 0 |  |  |
| 4 | “” | “” | 0 |  |  |
| 5 | James | “” | 0 | James | Username: |
| 6 | James | “” | 0 |  |  |
| 7 | James | Ross123 | 0 | Ross123 | Password: |
| 8 | James | Ross123 | 0 |  |  |
| 9 | James | Ross123 | 0 |  |  |
| 10 | James | Ross123 | 0 |  |  |
| 11 | James | Ross123 | 0 |  |  |
| 12 | James | Ross123 | 1 |  |  |
| 13 | James | Ross123 | 1 |  | That Username or Password is incorrect. You have 3 tries remaining |
| 14 | “” | Ross123 | 1 |  |  |
| 15 | “” | “” | 1 |  |  |
|  |  |  |  |  |  |
| 5 | WonderWalls | “” | 1 | WonderWalls | Username: |
| 6 | WonderWalls | “” | 1 |  |  |
| 7 | WonderWalls | Admin1 | 1 | Admin1 | Password: |

# ***First Name Validation***

1. firstName = (input("To add a customer please enter your first name: "))

2. checkName = False

3. while checkName == False:

4. for char in firstName:

5. if not char.isalpha():

6. print("not a valid entry, please enter a valid entry")

7. firstName = input("To add a customer please enter their name: ")

8. checkName = False

9. else:

10. checkName = True

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Line Number | firstName | checkName | Input | Output |
| 1 | James |  | James | To add a customer please enter your first name: |
| 2 | James | False |  |  |
| 3 | James | False |  |  |
| 4 | James | False |  |  |
| 5 | James | False |  |  |
| 6 | James | False |  |  |
| 7 | James | False |  |  |
| 8 | James | False |  |  |
| 9 | James | False |  |  |
| 10 | James | False |  | Continues with program |
| 1 | Jam1 |  | Jam1 | To add a customer please enter your first name: |
| 2 | Jam1 | False |  |  |
| 3 | Jam1 | False |  |  |
| 4 | Jam1 | False |  |  |
| 5 | Jam1 | False |  |  |
| 6 | Jam1 | False |  | not a valid entry, please enter a valid entry |
| 8 | Jam1 | False |  |  |
| 1 | James |  | James | To add a customer please enter your first name: |

# ***Postcode Validation***

1. postcode = input("Please enter their postcode: ")
2. checkPost = False
3. while checkPost == False:
4. for char in postcode[0:2]:
5. if not char.isalpha():
6. print("not a valid entry, please enter an appropriate entry")
7. postcode = input("Please enter in their postcode: ")
8. checkPost = False
9. if not postcode[2:5].isdigit():
10. print("not a valid entry, please enter an appropriate entry")
11. number = input("Please enter in their postcode: ")
12. checkPost = False
13. for char in postcode[5:7]:
14. if not char.isalpha():
15. print("not a valid entry, please enter an appropriate entry")
16. postcode = input("Please enter in their postcode: ")
17. checkPost = False
18. else:
19. checkPost = True

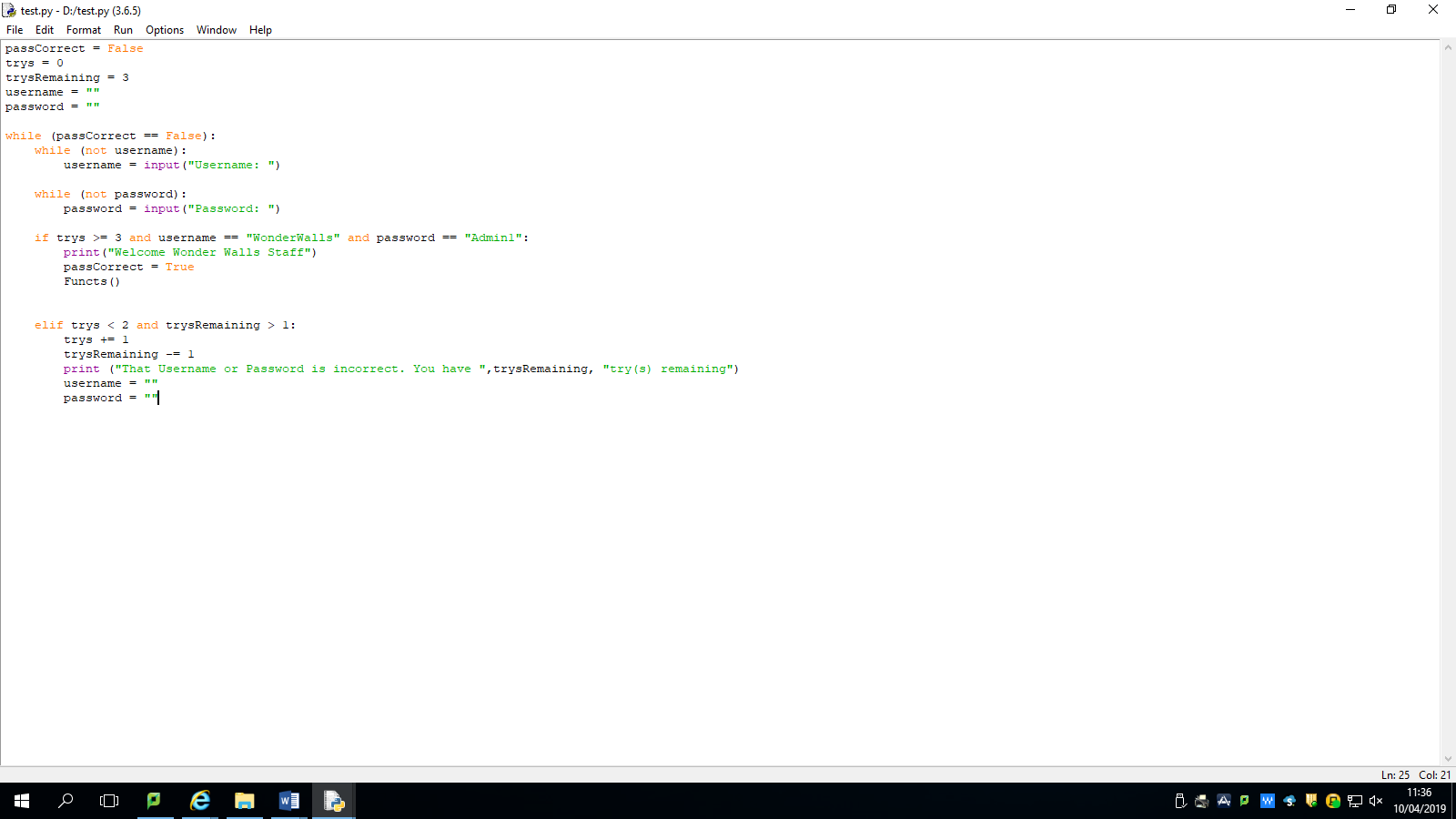
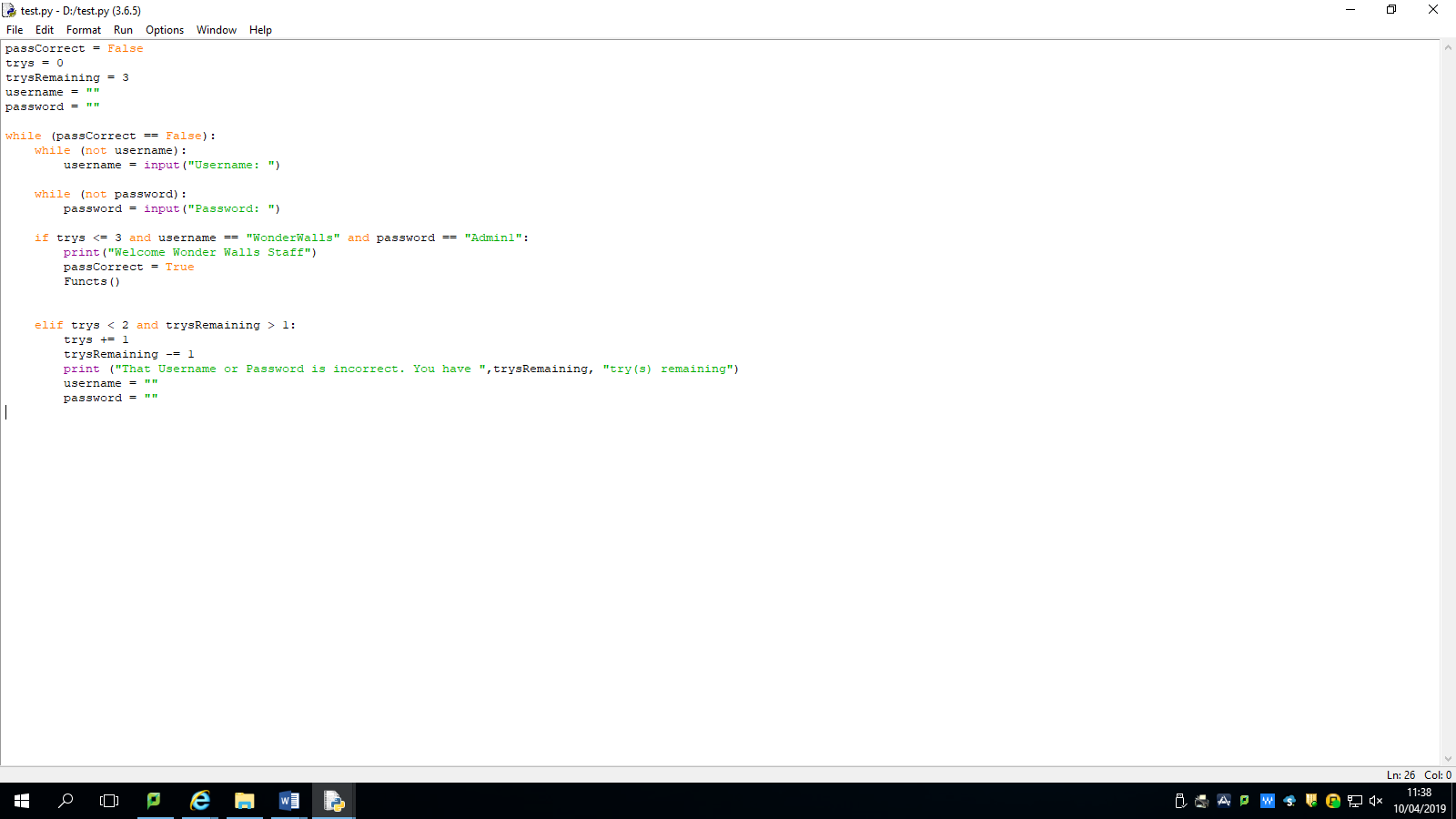
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Line Number | Postcode | checkPost | Input | Output |
| 1. | BT422DX |  | BT422DX | Please enter their postcode: |
| 2. | BT422DX | False |  |  |
| 3. | BT422DX | False |  |  |
| 4. | BT422DX | False |  |  |
| 5 | BT422DX | False |  |  |
| 6 | BT422DX | False |  |  |
| 7 | BT422DX | False |  |  |
| 8 | BT422DX | False |  |  |
| 9 | BT422DX | False |  |  |
| 10 | BT422DX | False |  |  |
| 11 | BT422DX | False |  |  |
| 12 | BT422DX | False |  |  |
| 13 | BT422DX | False |  |  |
| 14 | BT422DX | False |  |  |
| 15 | BT422DX | False |  |  |
| 16 | BT422DX | False |  |  |
| 17 | BT422DX | False |  |  |
| 18 | BT422DX | False |  |  |
| 19 | BT422DX | True |  |  |
| 1 | 42BT2DX |  | 42BT2DX | Please enter their postcode: |
| 2 | 42BT2DX | False |  |  |
| 3 | 42BT2DX | False |  |  |
| 4 | 42BT2DX | False |  |  |
| 5 | 42BT2DX | False |  |  |
| 6 | 42BT2DX | False |  | not a valid entry, please enter an appropriate entry |
| 7. | BT422DX | False | BT422DX | Please enter in their postcode: |

# ***Testing The Program***

Testing the program allows us to easily see the errors that have been made in the code and how to quickly and effectively correct them so that the program does not cause errors while the user is trying to do functions and ensures that the requirements of the user have been met.

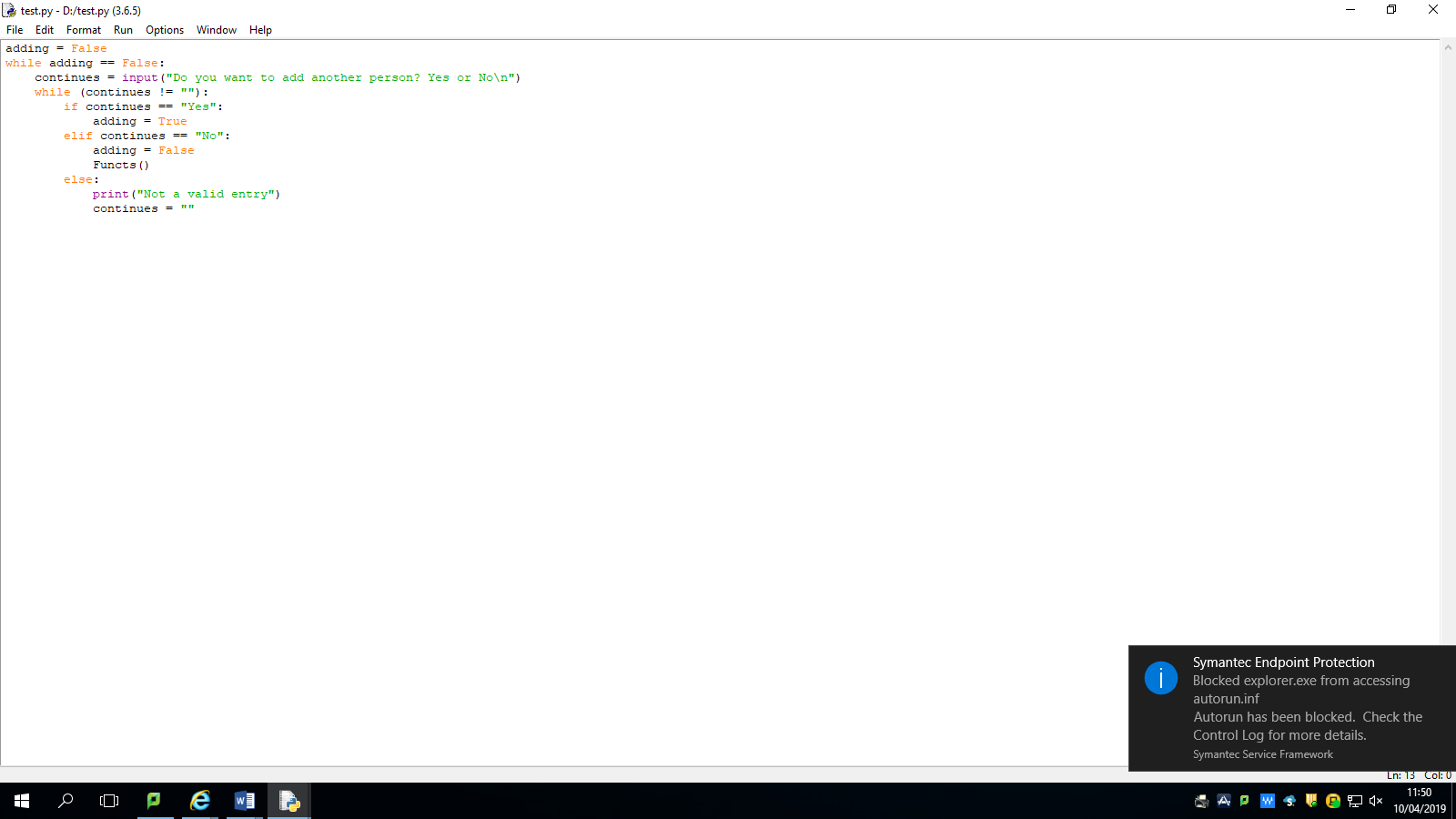
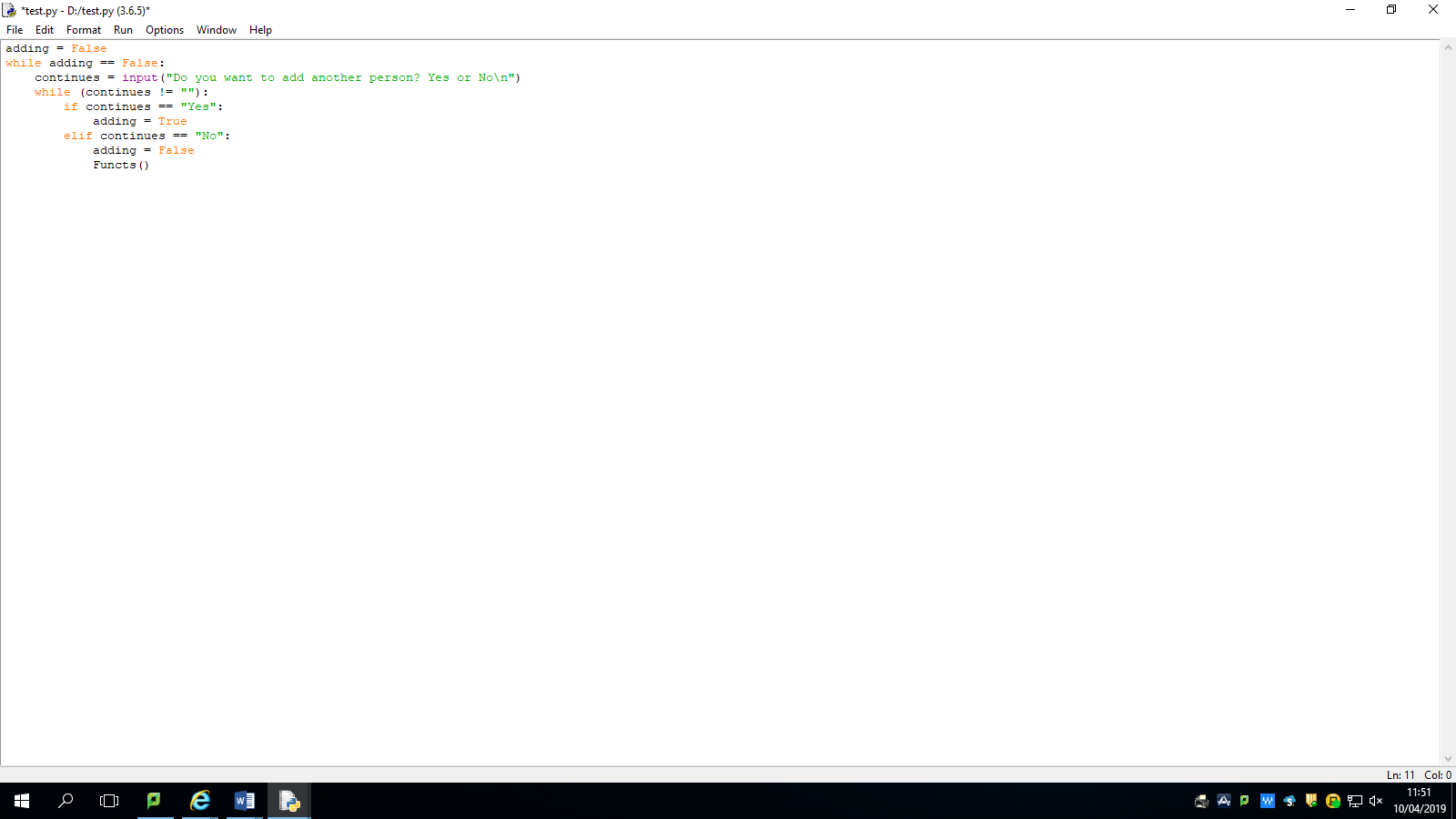
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Number | Test Function | Test Data | Expected Result | User Requirement | Actual Result |
| 1 | Log on or Log Off | Valid Data – 1 | Username:  (Allows user to log on) | 1 | Username:  (Allows user to log on) |
| 2 | Log on or Log Off | Null Data | Do you want to either:  1:login  2:shut down | 1 | Do you want to either:  1:login  2:shut down |
| 3 | Log on or Log Off | Invalid Data - James | Do you want to either:  1:login  2:shut down | 1 | Do you want to either:  1:login  2:shut down |
| 4 | Enter Username | Valid Data – Admin1 | Password: | 1 | Repeatedly asks for input |
| 5 | Enter Username | Null Data | Password: | 1 | Password: |
| 6 | Enter Username | Erroneous Data – Wonder Walls Staff | Password: | 1 | Password: |
| 7 | Enter Password | Valid Data – WonderWalls | Please Select a function:  (Allows access to menu system) | 1 | Please Select a function:  (Allows access to menu system) |
| 8 | Enter Password | Null Data | That Username or Password is incorrect.  Username: | 1 | That Username or Password is incorrect.  Username: |
| 9 | Enter Password | Invalid Data – Letmein123 | That Username or Password is incorrect.  Username: | 1 | That Username or Password is incorrect.  Username: |
| 10 | Choose Function | Valid Data – 1 | Allows user to access desired function | 2 | Allows user to access desired function |
| 11 | Choose Function | Null Data | Function: | 2 | Function: |
| 12 | Choose Function | Invalid Data – Function1 | Function: | 2 | Function: |
| 13 | Competitor First Name | Valid Data – James | Please enter their last name: | 3 | Please enter their last name: |
| 14 | Competitor First Name | Null Data | not a valid entry, please enter a valid entry. To add a customer please enter their first name: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their first name: |
| 15 | Competitor First Name | Invalid Data – James123 | not a valid entry, please enter a valid entry. To add a customer please enter their first name: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their first name: |
| 16 | Competitor Last Name | Valid Data – Ross | Please enter their postcode: | 3 | Please enter their postcode: |
| 17 | Competitor Last Name | Null Data | not a valid entry, please enter a valid entry. To add a customer please enter their last name: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their last name: |
| 18 | Competitor Last Name | Erroneous Data – Ross123 | not a valid entry, please enter a valid entry. To add a customer please enter their last name: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their last name: |
| 19 | Competitor Postcode | Accepted Data – BT422DX | Please enter their Gender - M/F: | 3 | Please enter their Gender - M/F: |
| 20 | Competitor Postcode | Null Data | not a valid entry, please enter a valid entry. To add a customer please enter their postcode: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their postcode: |
| 21 | Competitor Postcode | Invalid Data – B2422R1 | not a valid entry, please enter a valid entry. To add a customer please enter their postcode: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their postcode: |
| 22 | Competitor Gender | Valid Data – M | Please enter in their phone number: | 3 | Please enter in their phone number: |
| 23 | Competitor Gender | Null Data | not a valid entry, please enter a valid entry. To add a customer please enter their gender: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their gender: |
| 24 | Competitor Gender | Invalid Data – X | not a valid entry, please enter a valid entry. To add a customer please enter their gender: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their gender: |
| 25 | Competitor phone number | Valid Data – 07946119013 | Please enter their email address: | 3 | Please enter their email address: |
| 26 | Competitor phone number | Null Data | not a valid entry, please enter a valid entry. To add a customer please enter their phone number: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their phone number: |
| 27 | Competitor phone number | Invalid Data – 0111233 | not a valid entry, please enter a valid entry. To add a customer please enter their phone number: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their phone number: |
| 28 | Competitor email | Valid Data – jrross@hotmail.co.uk | This is the customer you are trying to add:  Is this correct? | 3 | This is the customer you are trying to add:  Is this correct? |
| 29 | Competitor email | Null Data | not a valid entry, please enter a valid entry. To add a customer please enter their email: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their email: |
| 30 | Competitor email | Invalid Data – jamesRoss@hotmail | not a valid entry, please enter a valid entry. To add a customer please enter their email: | 3 | not a valid entry, please enter a valid entry. To add a customer please enter their email: |
| 31 | Add competitor | Valid Data – Yes | Do you want to add another person? Yes or No | 3 | Do you want to add another person? Yes or No |
| 32 | Add competitor is correct | Null Data | Is this correct? | 3 | Is this correct? |
| 33 | Add competitor is correct | Invalid Data – 12345 | Is this correct? | 3 | Is this correct? |
| 34 | Add another competitor | Valid Data – No | Returns to Menu System | 3 | Returns to Menu System |
| 35 | Add another competitor | Null Data | Not a valid entry  Do you want to add another person? Yes or No | 3 | Not a valid entry  Do you want to add another person? Yes or No |
| 36 | Add another competitor | Invalid Data – 12345 | Not a valid entry  Do you want to add another person? Yes or No | 3 | Program ends |
| 37 | Find Competitor | Valid Data – 1 | Runs next process | 6 | Runs next process |
| 38 | Find Competitor | Null Data | What is your assigned number? (Only the number not the letters): | 6 | What is your assigned number? (Only the number not the letters): |
| 39 | Find Competitor | Invalid Data – WW5 | What is your assigned number? (Only the number not the letters): | 6 | Program ends |
| 40 | Reaction Test 1 | Valid Data – enter button pressed | Your score was: + score | 4 | Your score was: + score |
| 41 | Reaction Test 1 | Null Data | Invalid data. The test will now repeat | 4 | Invalid data. The test will now repeat |
| 42 | Reaction Test 1 | Invalid Data – click | Invalid data. The test will now repeat | 4 | Invalid data. The test will now repeat |
| 43 | Reaction Test 2 | Valid Data – 66 | Your score was: + score | 4 | Your score was: + score |
| 44 | Reaction Test 2 | Null Data | Invalid data. The test will now repeat | 4 | Invalid data. The test will now repeat |
| 45 | Reaction Test 2 | Invalid Data – 65 | Invalid data. The test will now repeat | 4 | Invalid data. The test will now repeat |
| 46 | Reaction Test 3 | Valid Data – t I w w | Your score was: + score | 4 | Your score was: + score |
| 47 | Reaction Test 3 | Null Data | Invalid data. The test will now repeat | 4 | Invalid data. The test will now repeat |
| 48 | Reaction Test 3 | Invalid Data – t I z z | Invalid data. The test will now repeat | 4 | Invalid data. The test will now repeat |
| 49 | Write reaction Times to file | Valid Data – 0.23, 0.33, 0.44 | Your reaction times have been successfully added to the file: to add another set of times select Reaction Time Tests from the menu | 4 | Writes reaction times to a new line |
| 50 | Write reaction Times to file | Null Data | Your reaction times have been successfully added to the file: to add another set of times select Reaction Time Tests from the menu | 4 | Your reaction times have been successfully added to the file: to add another set of times select Reaction Time Tests from the menu |
| 51 | Write reaction Times to file | Invalid Data – number, 033, james | Your reaction times have been successfully added to the file: to add another set of times select Reaction Time Tests from the menu | 4 | Your reaction times have been successfully added to the file: to add another set of times select Reaction Time Tests from the menu |
| 52 | Write competitor details to file | Valid Data – James, Ross, WW1 | Your Details have been successfully added to the file | 3 | Assigns auto number to 0 |
| 53 | Write competitor details to file | Null Data | Your Details have been successfully added to the file | 3 | Your Details have been successfully added to the file |
| 54 | Write competitor details to file | Invalid Data – James1, R0ss, 1WW | Your Details have been successfully added to the file | 3 | Your Details have been successfully added to the file |
| 55 | Remove competitor from file | Valid Data – No Data | Competitor successfully removed |  | Competitor successfully removed |
| 56 | Remove competitor from file | Null Data | Competitor successfully removed |  | Competitor successfully removed |
| 57 | Remove competitor from file | Invalid Data – James1, Ross,WW1 | Competitor has not been removed |  | Competitor has not been removed |
| 58 | Show File | Valid Data – No Data | Shows complete file | 6 | Shows complete file |
| 59 | Show File | Null Data | Shows complete file | 6 | Shows complete file |
| 60 | Show File | Invalid Data – James1, Ross,WW1 | Shows complete file | 6 | Shows complete file |
| 61 | Show competitor details | Valid Data – 1 | Shows competitor’s details | 6 | Shows competitor’s details |
| 62 | Show competitor details | Null Data | Competitor not found. Please enter a competitor’s number: | 6 | Competitor not found. Please enter a competitor’s number: |
| 63 | Show competitor details | Invalid Data –WW1 | Competitor not found. Please enter a competitor’s number: | 6 | Competitor not found. Please enter a competitor’s number: |
| 64 | Create file | Valid Data – list of fieldnames | File created with correct fieldnames |  | File created with nothing inside |
| 65 | Create file | Null Data | Blank file created |  | Blank file created |
| 66 | Create file | Invalid Data 132242434 | File created with erroneous data |  | File created with erroneous data |
| 67 | Entering climb time 1 | Valid Data – 22.2 | Please enter your second climb time: | 5 | Please enter your second climb time: |
| 68 | Entering climb time 1 | Null Data | Incorrect entry. Please enter your first climb time: | 5 | Incorrect entry. Please enter your first climb time: |
| 69 | Entering climb time 1 | Invalid Data climb23443 | Incorrect entry. Please enter your first climb time: | 5 | Incorrect entry. Please enter your first climb time: |
| 70 | Entering climb time 2 | Valid Data – 22.2 | Please enter your third climb time: | 5 | Please enter your third climb time: |
| 71 | Entering climb time 2 | Null Data | Incorrect entry. Please enter your second climb time: | 5 | Incorrect entry. Please enter your second climb time: |
| 72 | Entering climb time 2 | Invalid Data -climb23443 | Incorrect entry. Please enter your second climb time: | 5 | Incorrect entry. Please enter your second climb time: |
| 73 | Entering climb time 3 | Valid Data – 22.2 | Adds the times to file | 5 | Adds the times to file |
| 74 | Entering climb time 3 | Null Data | Incorrect entry. Please enter your third climb time: | 5 | Incorrect entry. Please enter your third climb time: |
| 75 | Entering climb time 3 | Invalid Data -climb23443 | Incorrect entry. Please enter your third climb time: | 5 | Incorrect entry. Please enter your third climb time: |
| 76 | Write times to file | Valid Data – 22.2,33.3,44.4 | Writes correct times to file | 5 | Writes correct times to file |
| 77 | Write times to file | Null Data | Writes no times to file | 5 | Writes no times to file |
| 78 | Write times to file | Invalid Data -climb56788,gg,44e | Writes incorrect times to file | 5 | Writes incorrect times to file |
| 79 | Report menu | Valid Data – 1 | Do you want to view the whole file? | 6 | Do you want to view the whole file? |
| 80 | Report menu | Null Data | That is not a valid entry. Report type: | 6 | That is not a valid entry. Report type: |
| 81 | Report menu | Invalid Data – hello | That is not a valid entry. Report type: | 6 | That is not a valid entry. Report type: |
| 82 | Report 1 | Valid Data – no data | Shows whole file | 6 | Shows whole file |
| 83 | Report 1 | Null Data | Shows whole file | 6 | Shows whole file |
| 84 | Report 1 | Invalid Data – show1 | That is not a valid entry. Report type: | 6 | That is not a valid entry. Report type: |
| 85 | Report 2 | Valid Data – 1 | Shows details for number 1 | 6 | Shows details for number 1 |
| 86 | Report 2 | Null Data | That is not a valid entry. Wonder Walls number: | 6 | That is not a valid entry. Wonder Walls number: |
| 87 | Report 2 | Invalid Data – show2 | That is not a valid entry. Wonder Walls number: | 6 | That is not a valid entry. Wonder Walls number: |
| 88 | Report 3 | Valid Data – no data | Shows file ordered for first reaction test | 6 | Shows file ordered for first reaction test |
| 89 | Report 3 | Null Data | Shows file ordered for first reaction test | 6 | Shows file ordered for first reaction test |
| 90 | Report 3 | Invalid Data – show3 | That is not a valid entry. Shows file ordered for first reaction test | 6 | That is not a valid entry. Shows file ordered for first reaction test |
| 91 | Report 4 | Valid Data – no data | Shows file ordered for first climb time | 6 | Shows file ordered for first climb time |
| 92 | Report 4 | Null Data | Shows file ordered for first climb time | 6 | Shows file ordered for first climb time |
| 93 | Report 4 | Invalid Data – show4 | That is not a valid entry. Shows file ordered for first climb time | 6 | That is not a valid entry. Shows file ordered for first climb time |
| 94 | Log off | Valid Data – log me off | Logging off | 1 | Logging off |
| 95 | Log off | Null Data | Not a valid entry. Please choose a function: | 1 | Not a valid entry. Please choose a function: |
| 96 | Log off | Invalid Data – 123 | Not a valid entry. Please choose a function: | 1 | Not a valid entry. Please choose a function: |
| 97 | Shut down | Valid Data – shut down | Shutting down | 1 | Shutting down |
| 98 | Shut down | Null Data | Not a valid entry. Please choose a function: | 1 | Not a valid entry. Please choose a function: |
| 99 | Shut down | Invalid Data – 342 | Not a valid entry. Please choose a function: | 1 | Not a valid entry. Please choose a function: |

# ***Corrective Testing***

Failed Test Code Corrected Test Code

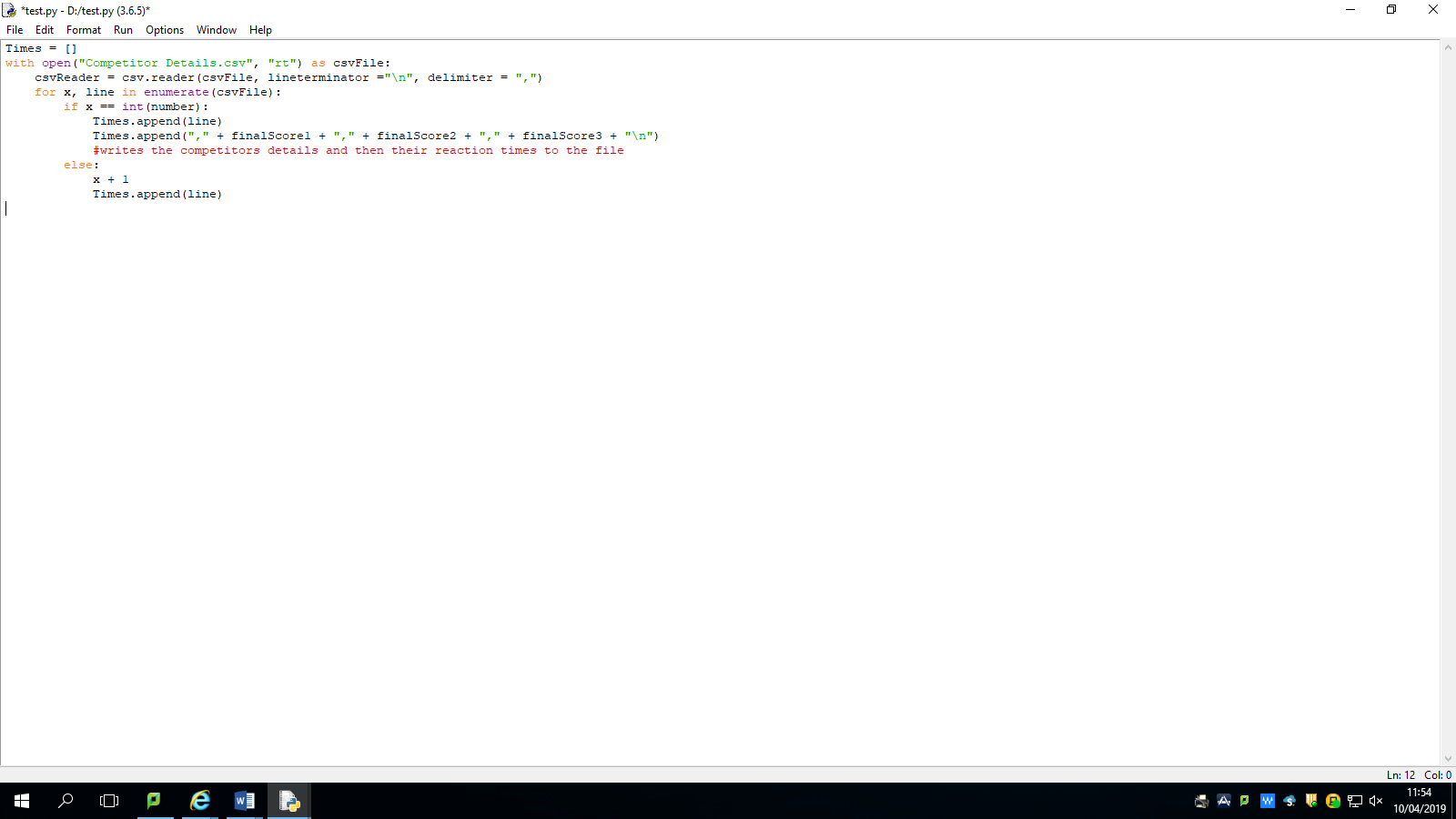
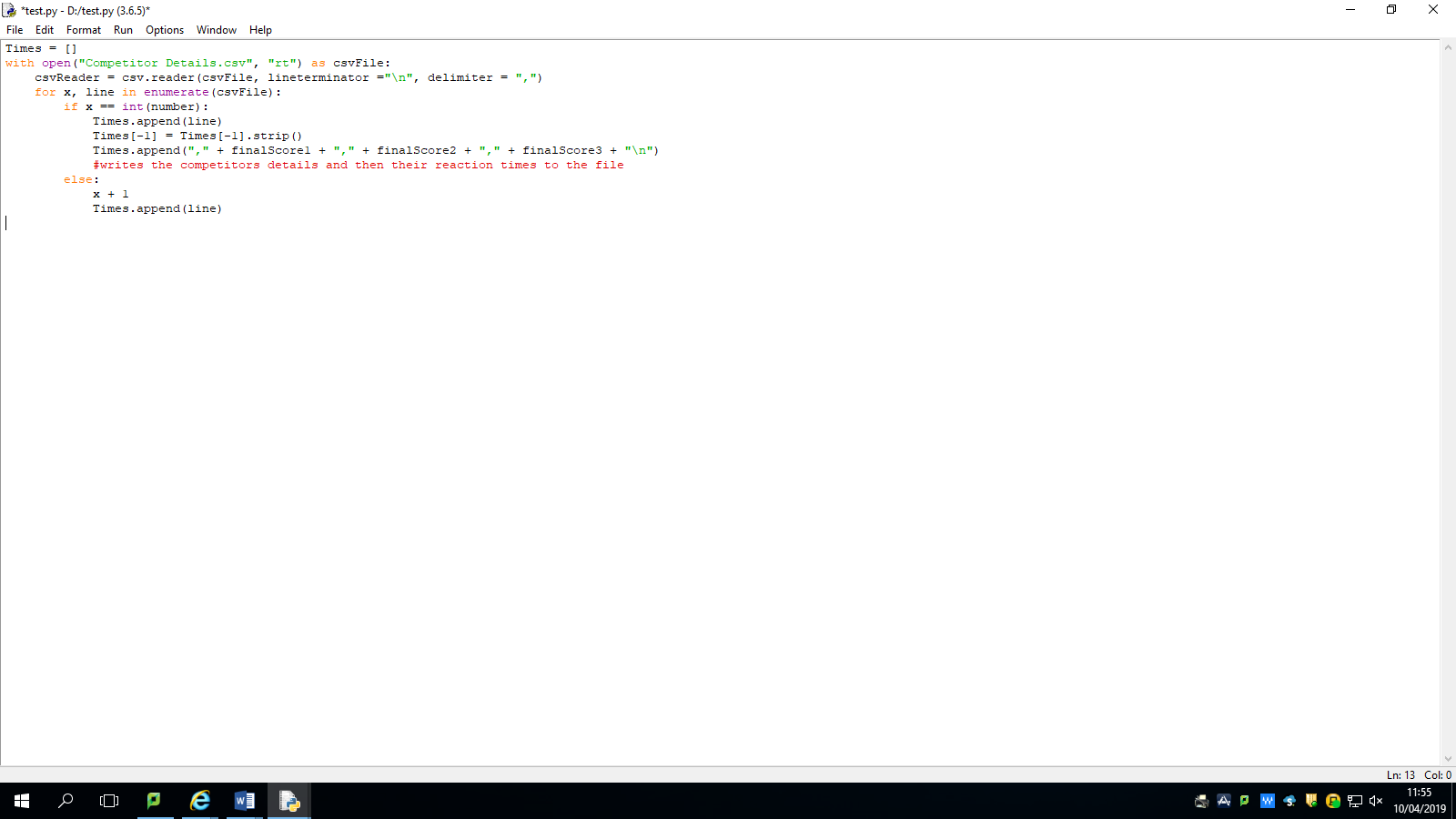
To fix this I had to change the operator in the if statement so it is a less than rather than greater than.

When entering username and password, the program would continue to ask for inputs after correct ones given. Test number 4.



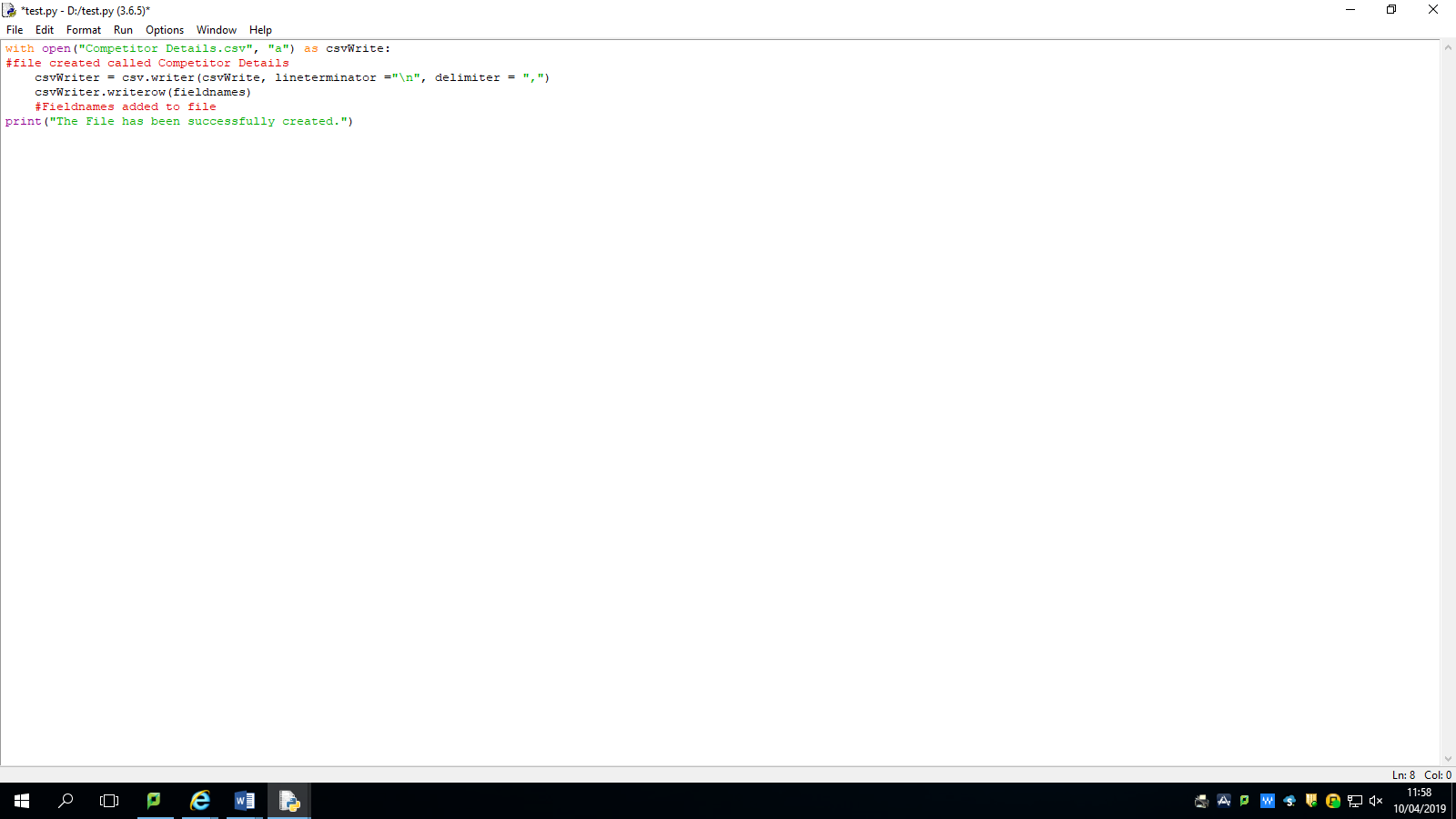
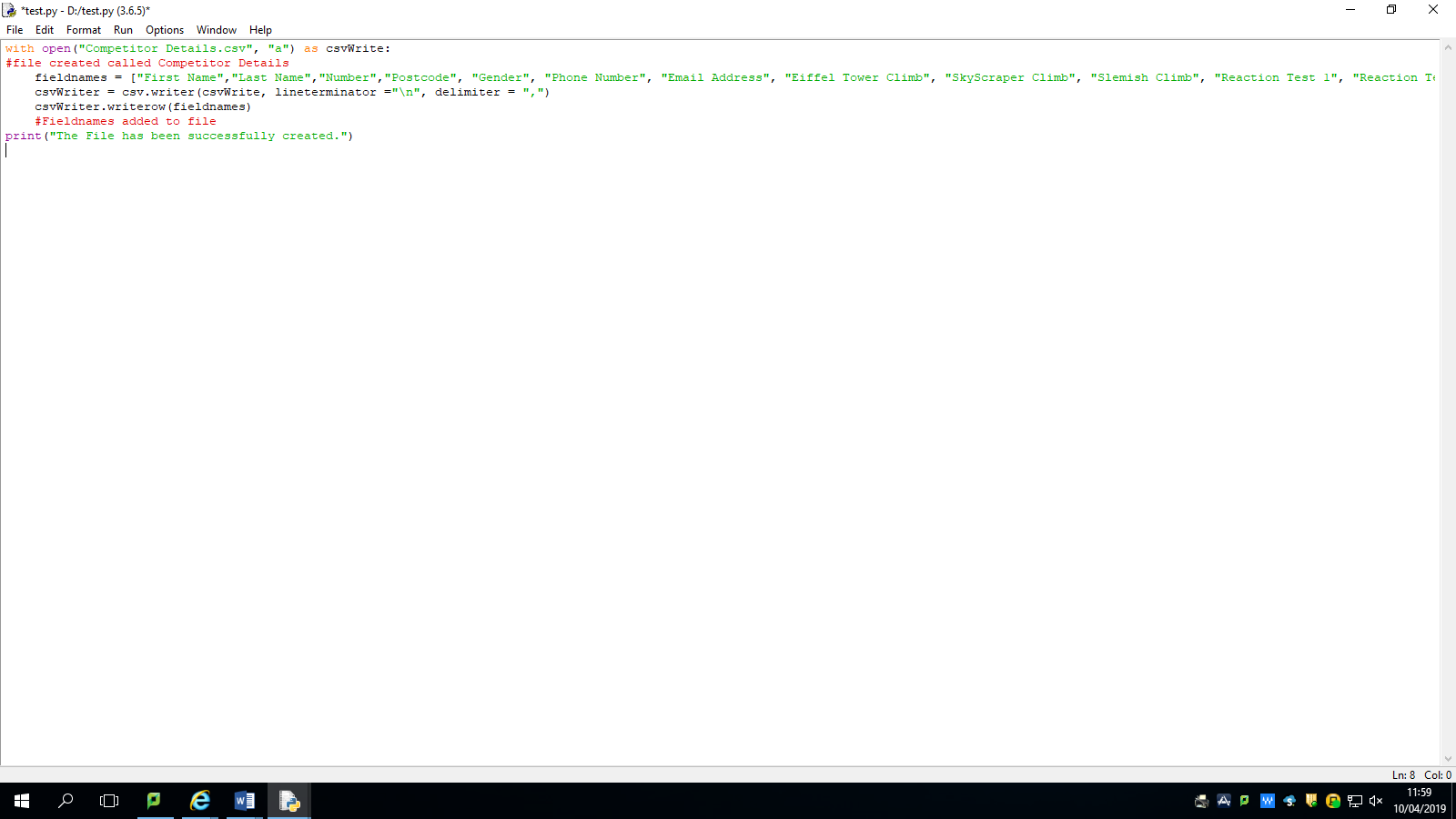
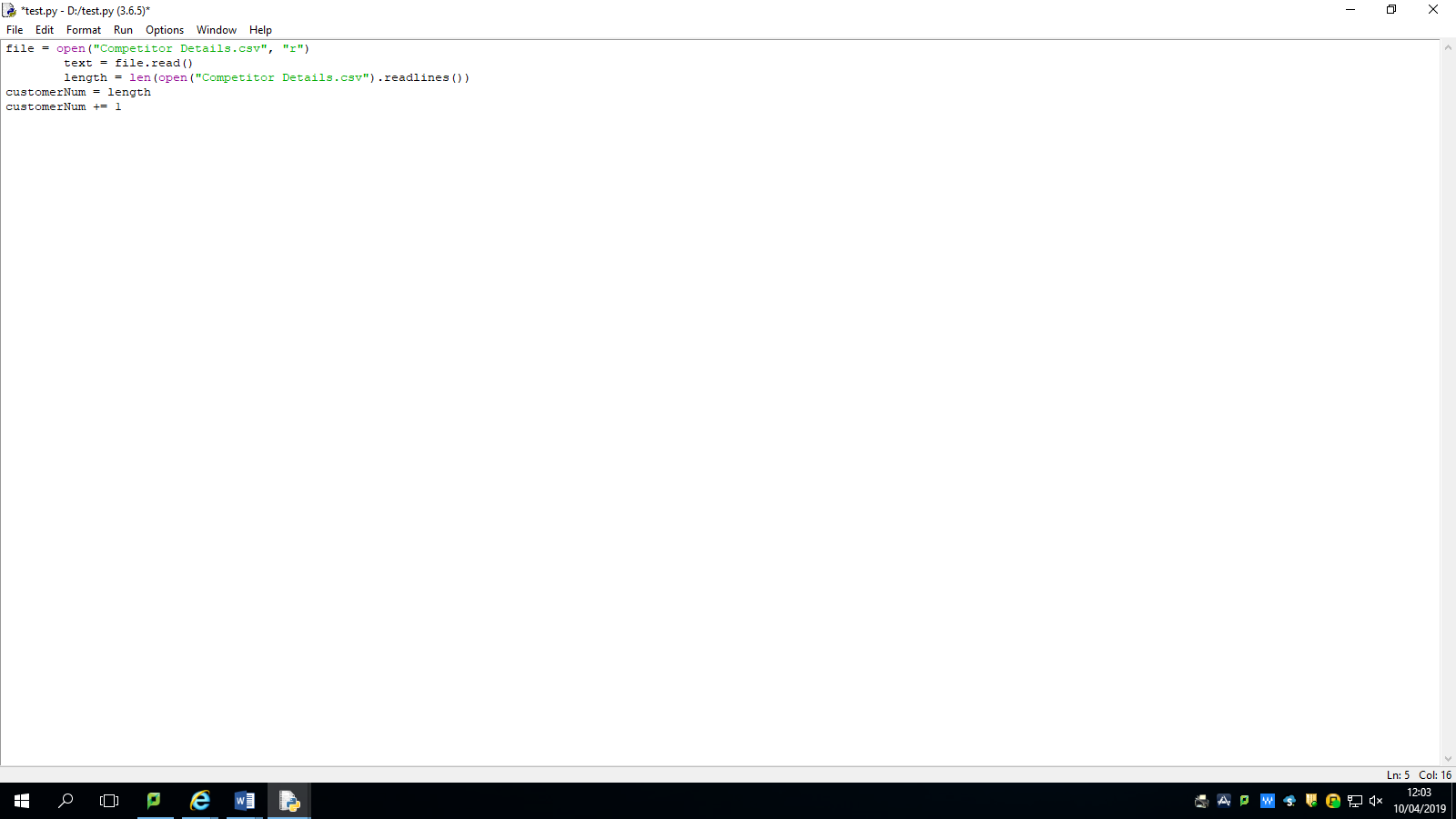
If incorrect entry given when asking if the user wanted to add another competitor program would end. Test number 36.

This simple piece of code askes the user if they want to add another competitor, to avoid the program from ending if the wrong inputs are given, if the user enters in any other value other than Yes or No then the user is prompted to renter the data so that a correct value is given.



When adding reaction times to file, times would be added onto a new line. Test number 49.

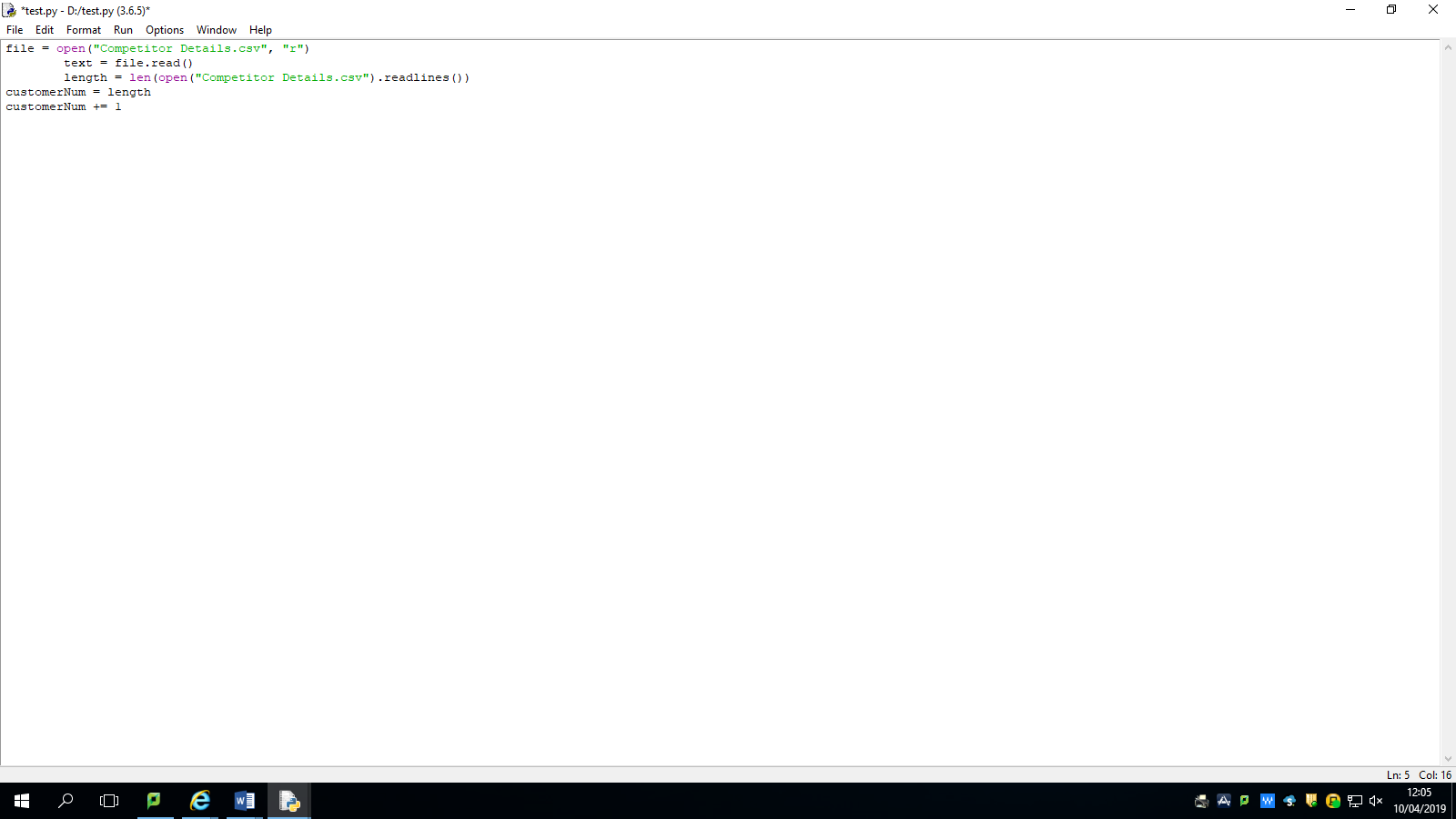
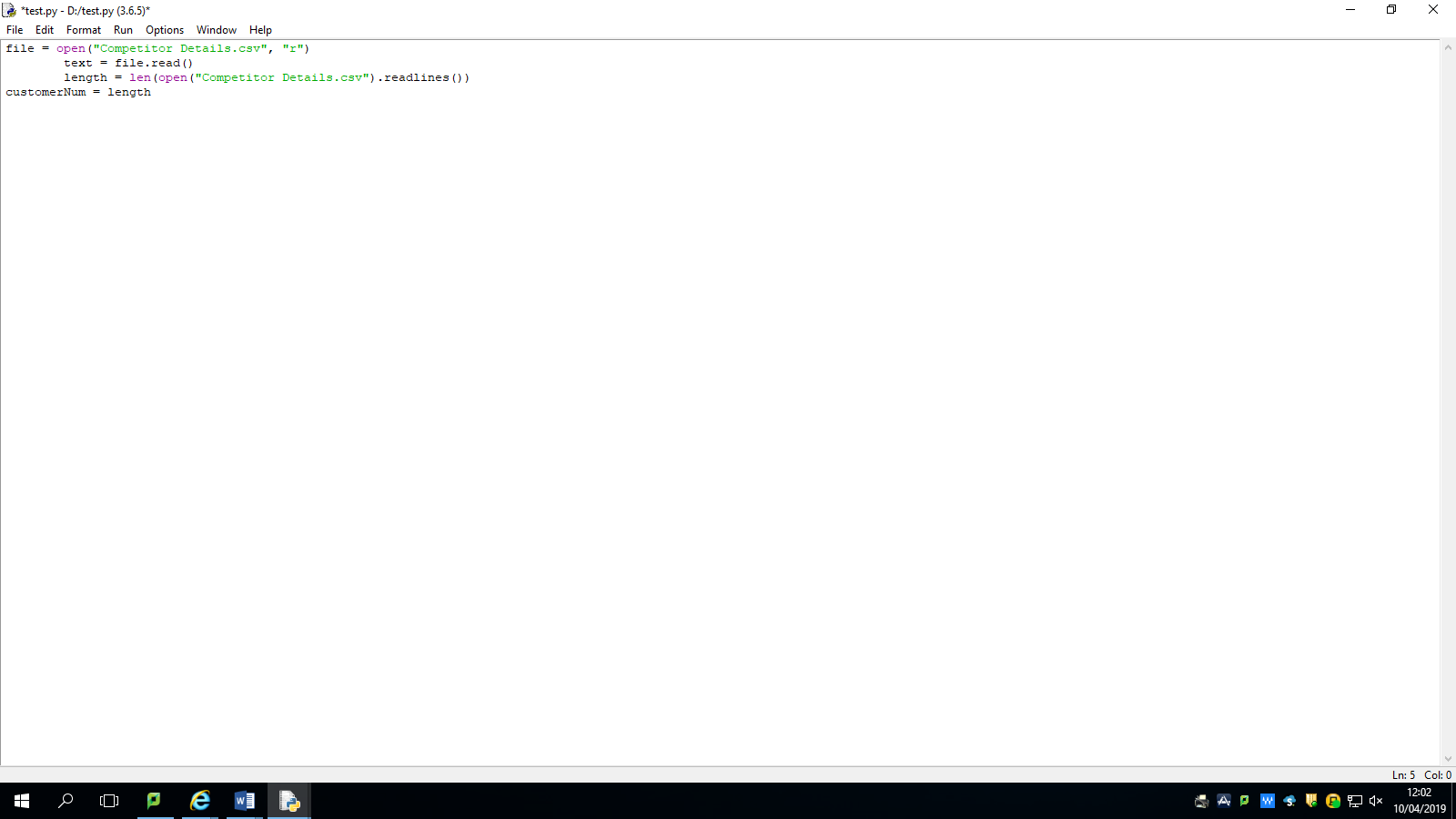
The system first adds the existing line to the list called Times and then using the strip command removes the comma at the end allow the program to amend to the file the 3 scores from the reaction test onto the same line as the competitor’s details.

Creating file – blank file created without fieldnames. Test number 64.

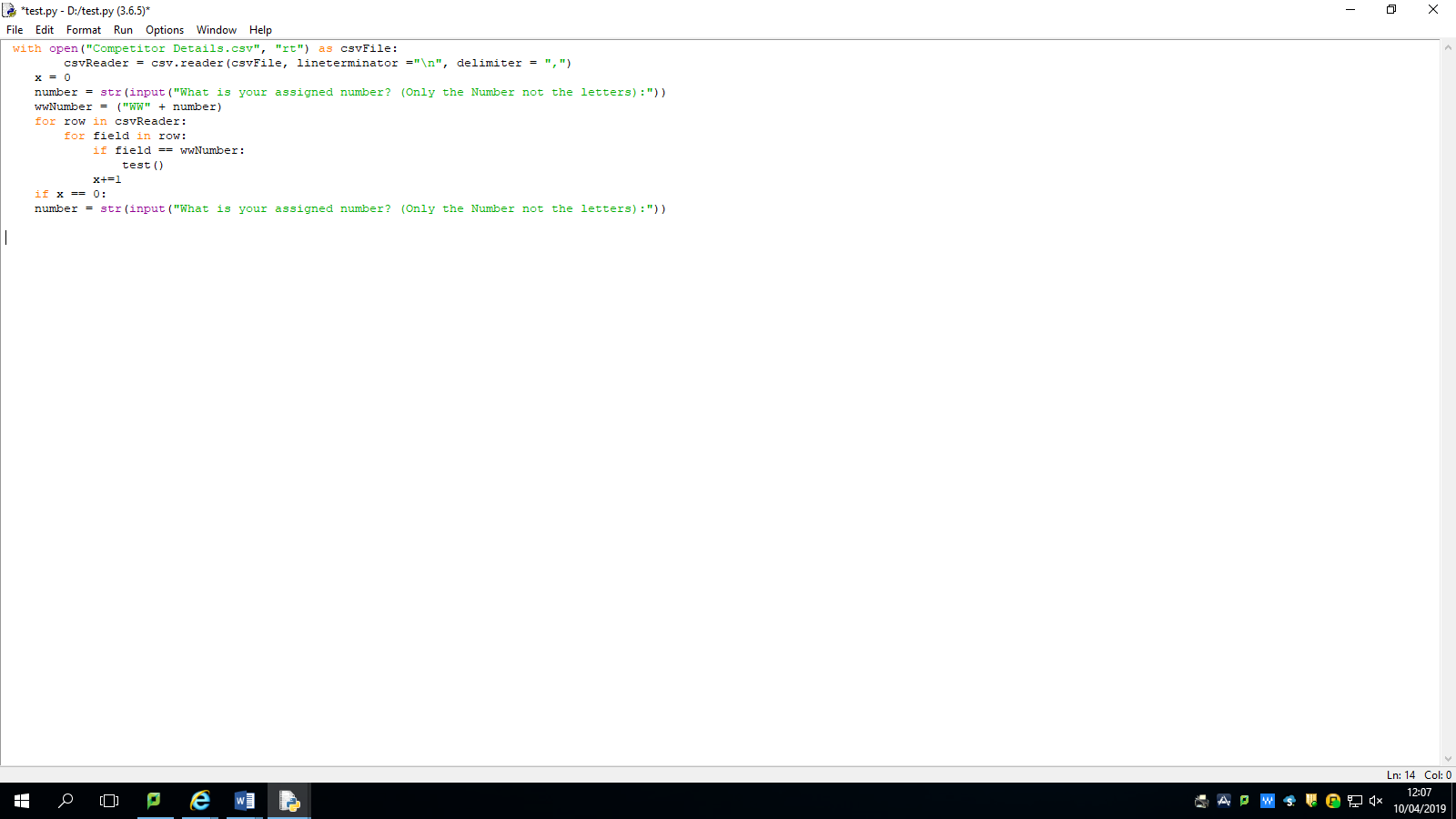
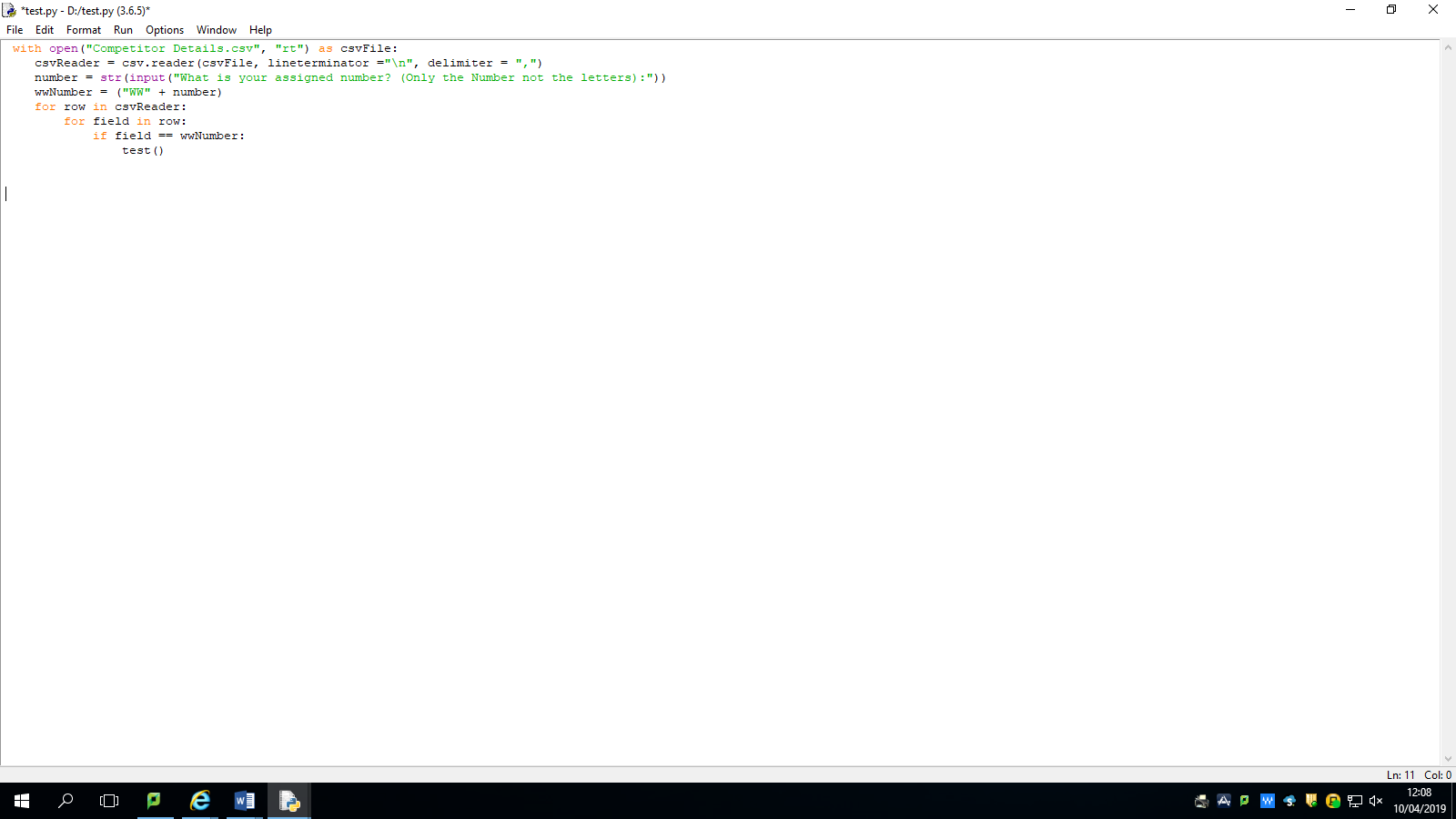
This opens the csv file Competitor details and creates a list called fieldnames – this has all of the column names in it. The file is created when it is opened as append by the program and it then also writes a new row with the contents of fieldnames.

Creating file – blank file created without fieldnames. Test number 64



Assigns Auto number to 0. Test number 52.

Adds 1 to the existing number so that the first number is 1 instead 0.



When asked for their Wonder Walls number, if user enters in a number which cannot be found, the program ends. Test number 39.

This opens the csv file again and asks the user to enter in their assigned number and sets the variable x to 0, the program then checks each row to see if it can find this number for example WW2. If the number is found, then the next function in this case – test is run and the variable x is incremented. If the number is not found x remains 0 and the user is prompted to renter their assigned number.

In conclusion testing is very important as it allows the programmer to test inputting various types of data to see if they give the desired results, if not then the programmer must change the code so that the program can accommodate for that type of data. The code can then be retested and checked to see if it matches the desired result once more. Testing is also related to the user requirements as it allows us to see what the user needs out of the system. For example, the user needs a specific input of competitor details however the first name of the competitor cannot be a number so the system needs to be able to recognise this and prompt the user to renter the data with the correct values. This helps the programmer to easily validate inputs as they can test a wide variety of different data types which the user can enter. It also helps us to clearly see the user requirements and which tests are part of certain user requirements. It ensures that these requirements and successfully met by the program and it can handle any sort of data that the user enters. My testing is extensive as to ensure all the user requirements have been met and that the system can correctly input validate and output data without causing errors

# ***Evaluation***

My solution has met the user requirements that have been set out by the Wonder Walls staff in the following ways:

# ***User requirements:***

The program is able to require the user to enter in a username and password into the system which in this case are WonderWalls and Admin1. If the user enters in the correct data, they are then taken to the main menu however, if they enter in an incorrect entry they will be required to renter both the username and password. If they incorrectly enter this three times, the system will automatically shut down as this prevents an infinite number of attempts. The system further fulfils the user requirements as it provides a menu system where the user can carry out 9 different functions such as entering in a competitor’s details and carrying out reaction time tests, this menu is required for the wonder walls staff to easily navigate in the program. My program also allows the user to enter in a competitor’s details such as their first and last names, their phone numbers and emails. Before automatically assigning the competitor an ID number which is used to locate a specific competitor in the system. This meets the requirements set out by staff that they must be able to add competitors to a file as well as having an ID number. The program also includes a reaction time test which each competitor is meant to take – it comprises of three separate tests one with a simple click, the next with entering in a number in a sequence and finally then one entering in a certain character. These three tests are individually averaged over three attempts at each and then are added to the csv file along with all the other details which have been previously saved by the program itself – this also matches the user requirements as it provides three separate reaction time results for each competitor that has done the test. After completing the reaction tests, the user can then enter in three climb times that the competitor has taken. This is added to the file by the system similar to the reaction tests. My program has met the user requirement for this as it easily allows Wonder Walls staff to enter a competitor’s climb times. The user requirements also state that the system must be able to produce at least four reports. My program has met this requirement as it provides four reports. The first one shows the whole csv file to the user. The second shows the details and results for an individual competitor. The third report orders competitors on their result in the first reaction time test. And the fourth report orders the competitors on the first climb time results. This meets the user requirement as it effectively presents the user with four different reports based on the competitors and their details. Although all the user requirements have been met in the ways above I have also added a number of other features to the program. I have added an option to shut down the program as to not keep it running all the time. Finally, I have added an option to create the csv file from the program rather than manually creating it. Although the user does not specifically require these functions to be in the program I have included them as I see them as beneficial to the user and it will make the program much more straightforward and effective for the user to use.

# ***Performance during testing:***

During my in depth testing the system had encountered a number of errors such as validation and writing to a file. Testing allowed me to see what exactly the program was doing to cause the error and it made them much easier to solve. One example of this would be trying to get the program to add the reaction times to the file. As the file would not allow me to simply choose a line to write data to – this would cause the file to rewrite itself with just the reaction tests I had to create a separate temporary list in the program to write all the data to (which included the reaction tests) this list was the used to write to the file. It replaced the data already in the file with that data plus the reaction time tests. I only knew how to do this through testing as it highlighted the error to me and when I tested it the file returned with just the reaction times and nothing else. Testing ensured that the program had met all the user requirements and that the program effectively worked and would not stop working for no reason. During testing I encountered a number of errors such as the one above which would have meant that the program would have not been able to achieve the requirements set out by staff.

# ***Refinements following testing:***

As a result of testing I was able to locate and eliminate any errors that may have caused my system to stop working and fail. Such failures could have meant the system randomly stopped or the data entered by the user had not been correctly validated and therefore data entered into the file by the system would have been inaccurate or incorrect. Following testing I have found a number of ways in which the program needed to be or could have been improved. The program needed to be improved in terms of validating as it incorrectly validated a competitor’s email and phone number during testing and I quickly rectified this. The program also needed to correctly append data to the csv file after testing and I have now managed to do so. The program also incorrectly ordered the csv file for two of the reports as a result of them starting from high to low instead of low to high. These issues have all now been fixed. However, there are a number of ways in which I could have improved upon the program. I could have added a feature where the user could change data already added to the file such as their phone number. The user also could have been able to see reports based on the other times provided or the program could have been able to produce a report based on all results given in the file. I would have added all these features if I had more time to do so.

# ***Robustness of the system:***

My program is robust in a variety of ways as it accounts for the user entering in both correct and incorrect data. For example, when entering in their choice of menu function the user has to enter in any number from 1-9. If the user does not enter this data in, then the system does not stop as it knows that when the user enters any character or number that is not in that range It will prompt the user to renter their choice but this time using the correct data. This process does not stop after one attempt but will go on indefinitely until the user exits the program or enters in a valid option. My program will do this for every piece of data that is entered in by the user to ensure that the program does not crash at the end if any similar circumstances occur such as the one above. This means that my system is very robust as it can easily handle a user inputting a random character when a specific number of integers should instead be entered in its place. This is very important that my system is robust as it means that the Wonder Walls staff will not encounter any errors within my program and it will be able to easily deal with unknown data given as it knows what data it is looking for and what to do if the user does not supply this data. Therefore, this means that the program has a high ability to deal with errors and exceptional circumstances that it may encounter when being used by staff to enter in competitor details or complete reaction time tests.