Cab301 Algorithms and complexity

2021 semester 1 assignment (applied project)

By James Norman n10256989

Contents

Introduction:

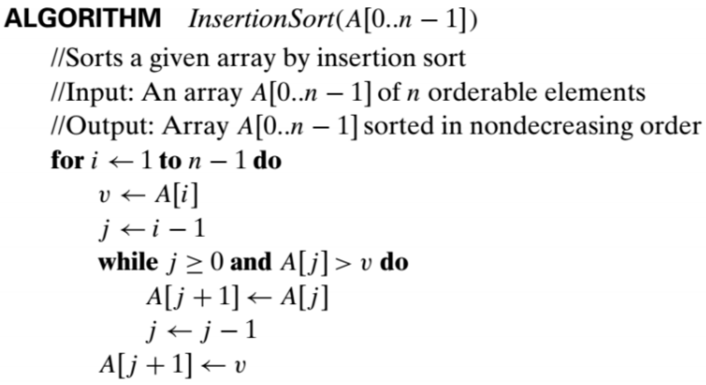
This report is based on the 2021 Cab301 assignment “Development of a Software Application for a Tool Library” in this project we were tasked with creating a program to solve a real-world problem. This program solved the problem of a Tool library, using data structures and algorithms covered in the unit. In the process designing algorithms to solve computational problems and analysing the time efficiency of algorithms. The purpose of this report is to display an understanding of the algorithms inside of this program and to understand and show their complexity and efficiency. This report will cover algorithm design, algorithm analysis, software test plan and test results.

Algorithm design:

Note: this is all a hypothetical as my program did not include a top three function as I was unable to implement it. However, I have done research on which algorithm to use and designed the approach I would have taken had I implemented it.

The process of designing an algorithm to return the top three most popular tools was a rather interesting one with two sorting algorithms being clear favourites to choose. These algorithms were the heap sort algorithm and the insertion algorithm. Both algorithms had their own strengths which made them seem like the best for the job. Let us start with some assumptions It is to be assumed that the data set passed through would be an array with multiple tools comprising of all the tool types in a single array, thus making the array random and not sorted. The array will contain the tool name, the quantity, the available quantity, and the number of borrowings. The final product will contain at most 3 tools with the list being re-sorted every time a new tool is added or borrowed. The heap sort algorithm is the first design that will be talked about in the design section, more specifically the max heap method. The max heap sort algorithm is the most efficient algorithm when it comes to finding the max values in a random array with an average and worst case run time of O(n*log(n)) and a best run time of O(n). which makes it an efficient method to find the top value in a random array. After this method has been run the array that contains the top three tools will be printed with all elements of the array. However, the insertion sort algorithm become much more efficient when you consider the scope of the project. The insertion sort algorithm is much more efficient than the heap sort when you consider that the insertion algorithm is the fastest at searching through a nearly sorted array, which the array will be after the first time it is sorted O(n). Add in the fact that the array is always callable with no modifications giving it a request time of O (1).

Algorithm analysis:



```
ALGORITHM InsertionSort(A[0..n − 1])
    //Sorts a given array by insertion sort
    //Input: An array A[0..n − 1] of n orderable elements
    //Output: Array A[0..n − 1] sorted in nondecreasing order
    for i ← 1 to n − 1 do
        v ← A[i]
        j ← i − 1
        while j ≥ 0 and A[j] > v do
            A[j + 1] ← A[j]
            j ← j − 1
        A[j + 1] ← v
```

heap sort will not be spoken about in the algorithm analysis as it was shown above that it was less efficient than the insertion sort.

The algorithm for the insertion sort is shown above, this algorithm will loop through the array starting from the first element A [1] to A[n-1] A [0] is excluded due to being part of a sorted sub list. A[i] is also stored in the value of v temporarily for each iteration of the loop. The while loop decrements through the sub list and checks that the current tool has a higher value for number of borrowings than the previously calculated tools, if so, the tool stored in A[j] is then moved to A[j+1] lowering it on the list. However, if v has a lower number of borrowings it is inserted into the index after A[j] in the sorted sub list.

The best case for this algorithm is 0(n) if the array is already sorted as the second loop would have no reason to execute. Because the elements are already in order, the basic operation which is the A[j] < v comparison, is only executed once for each iteration of the for loop.

$$C_{best}(n) = \sum_{i=1}^{n-1} 1 = n - 1 \in \theta(n)$$

In terms of the worst-case scenario for the time complexity, it would be 0(n^2) as if the array is initially unsorted or sorted in the wrong order both loops will be executed throughout every iteration of the algorithm.

$$C_{worst}(n) = \sum_{i=1}^{n-1}\sum_{j=0}^{i-1} 1 = \sum_{i=1}^{n-1} i = \frac{(n-1)n}{2} \in O(n^2)$$

The algorithm to print the top 3 looks like this

```
Algorithm TopThree()
//iterates through the sorted array and prints the top 3 elements
For k 0 to 3 do
  PrintA[i].Tool.name
  PrintA[i]Tool.noBorrowings
```

Each part of this statement has a time complexity of O(1) as it executes 3 times it will have an overall time complexity of O(n).

Software test plan and test results:

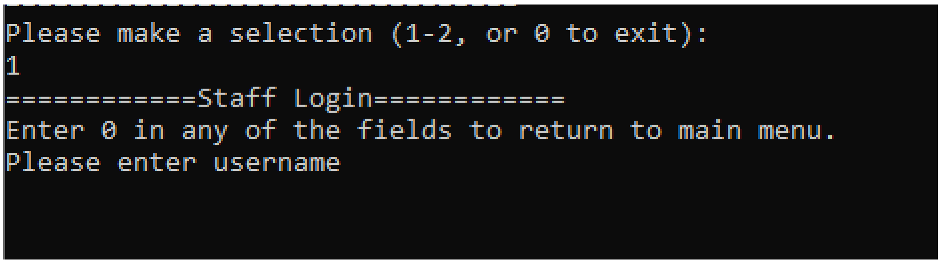
In this section a test plan and test results will be shown for the program that I have created, please note that not all tests go as plan and hence the tests. These tests will be broken up into three groups General checks, Staff menu checks and member menu checks.

General Checks:

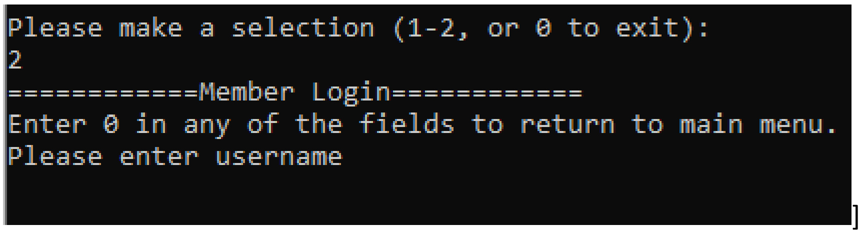
Test 1. Navigation of starting menu

Desired result: pressing 1 will result in staff login page being shown, pressing 2 will result in member login page being shown.

Result:



```
Please make a selection (1-2, or 0 to exit):
1
============Staff Login============
Enter 0 in any of the fields to return to main menu.
Please enter username
```



```
Please make a selection (1-2, or 0 to exit):
2
============Member Login============
Enter 0 in any of the fields to return to main menu.
Please enter username
```

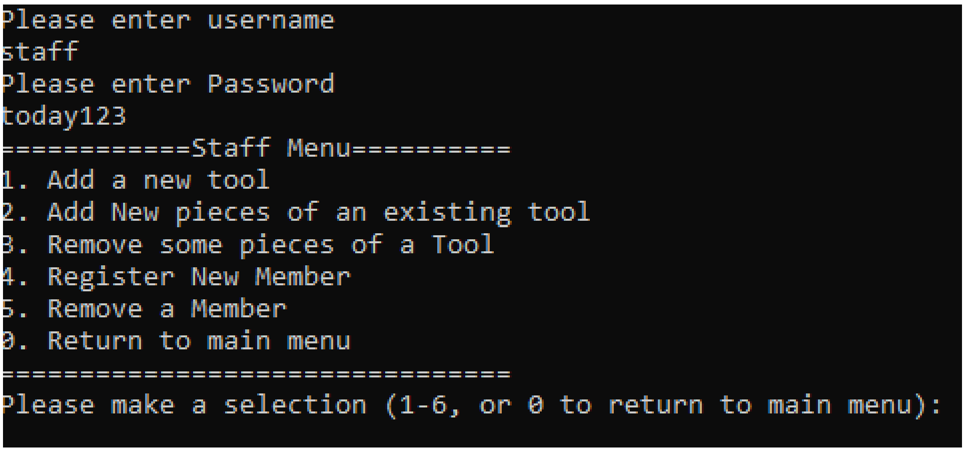
]

As shown both tests succeed.

Test 2. Checking staff login works

Desired result: When the correct username and password are entered the program enters the staff menu.

Result:



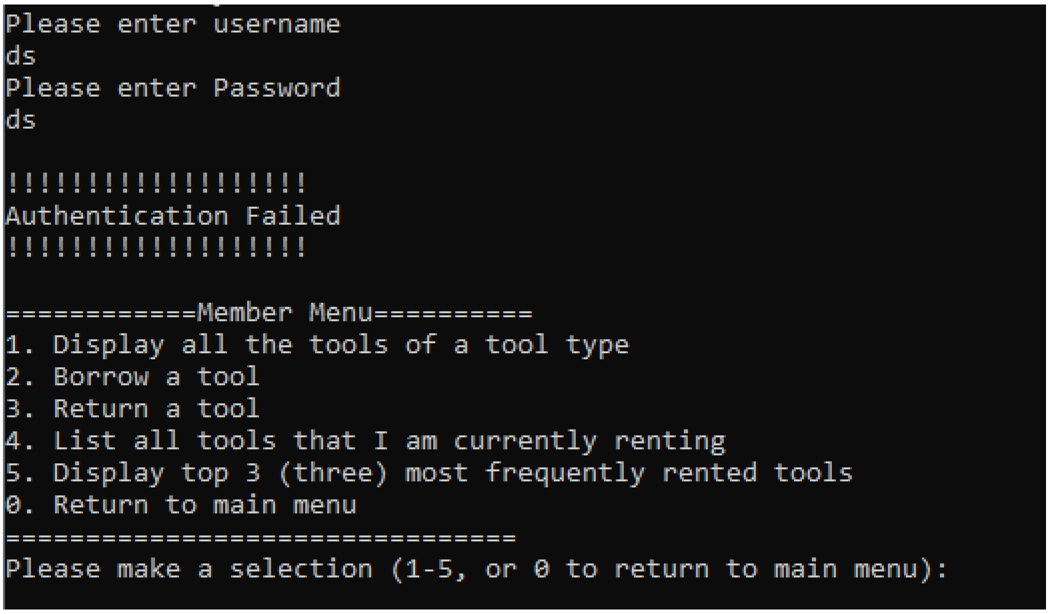
```
Please enter username
staff
Please enter Password
today123
===========Staff Menu=========
1. Add a new tool
2. Add New pieces of an existing tool
3. Remove some pieces of a Tool
4. Register New Member
5. Remove a Member
0. Return to main menu
================================
Please make a selection (1-6, or 0 to return to main menu):
```

As shown the test succeeds.

Test 3. Checking member login works

Desired result: When the correct username and pin are entered the program enters the member menu.

Result:



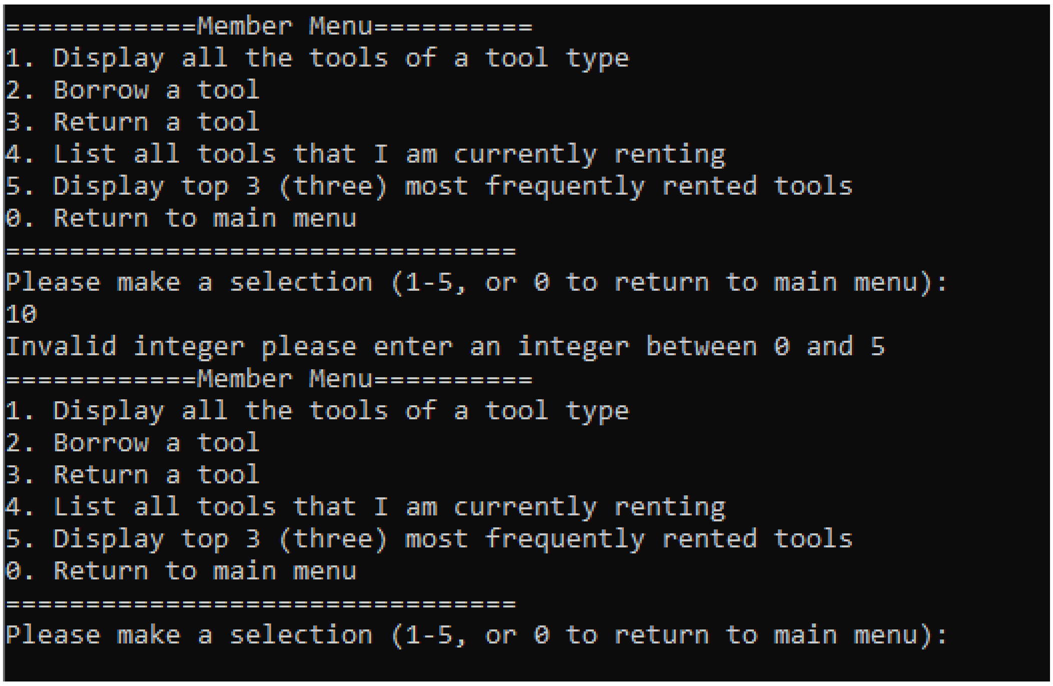
```
Please enter username
ds
Please enter Password
ds

!!!!!!!!!!!!!!!!!!!
Authentication Failed
!!!!!!!!!!!!!!!!!!!

===========Member Menu=========
1. Display all the tools of a tool type
2. Borrow a tool
3. Return a tool
4. List all tools that I am currently renting
5. Display top 3 (three) most frequently rented tools
0. Return to main menu
================================
Please make a selection (1-5, or 0 to return to main menu):
```

As shown the member login works and directs us to the member menu but as a function to detect usernames has not been written the authentication failed message will always come up. However, access to the menu is given else the program won't be able to run.

Test 4. Checking that the invalid selection works

Desired result: When an invalid number is chosen the program will raise an error message and bring up the previous screen to choose again

Result:



```
============Member Menu==========
1. Display all the tools of a tool type
2. Borrow a tool
3. Return a tool
4. List all tools that I am currently renting
5. Display top 3 (three) most frequently rented tools
0. Return to main menu
================================
Please make a selection (1-5, or 0 to return to main menu):
10
Invalid integer please enter an integer between 0 and 5
============Member Menu==========
1. Display all the tools of a tool type
2. Borrow a tool
3. Return a tool
4. List all tools that I am currently renting
5. Display top 3 (three) most frequently rented tools
0. Return to main menu
================================
Please make a selection (1-5, or 0 to return to main menu):
```

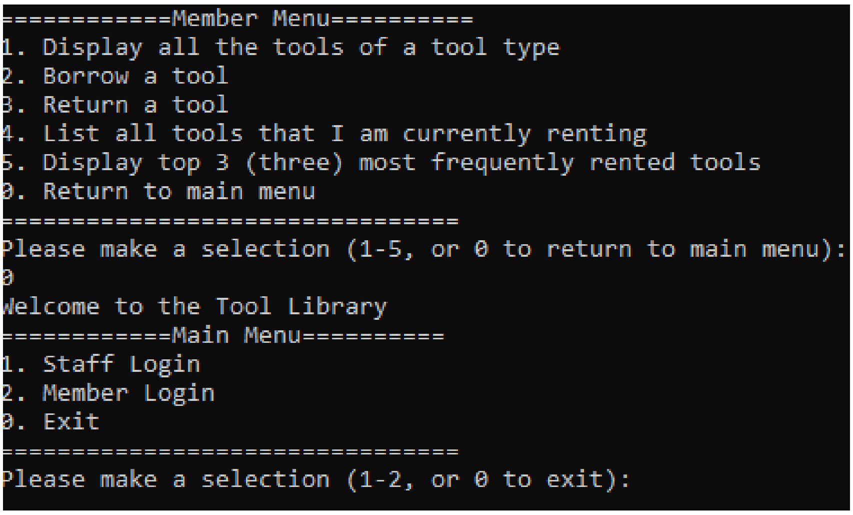
As you can see the invalid integer error shows up and reshows the menu, this error is applicable in any menu selection that requires, and integer listed on the side of the screen.

Test 5. Checking that the 0 key to return works

Desired result: When the 0 key is hit and entered the program will go back to the first page and if it is on the login page it will exit the program.

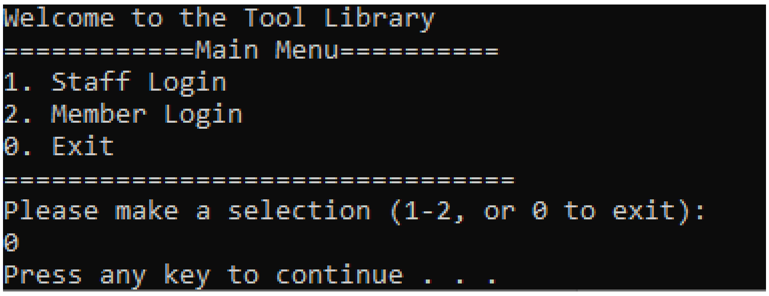
Result:

From Member menu



```
============Member Menu==========
1. Display all the tools of a tool type
2. Borrow a tool
3. Return a tool
4. List all tools that I am currently renting
5. Display top 3 (three) most frequently rented tools
0. Return to main menu
================================
Please make a selection (1-5, or 0 to return to main menu):
0
Welcome to the Tool Library
============Main Menu==========
1. Staff Login
2. Member Login
0. Exit
================================
Please make a selection (1-2, or 0 to exit):
```

From login page



```
Welcome to the Tool Library
===========Main Menu=========
1. Staff Login
2. Member Login
0. Exit
================================
Please make a selection (1-2, or 0 to exit):
0
Press any key to continue . . .
```

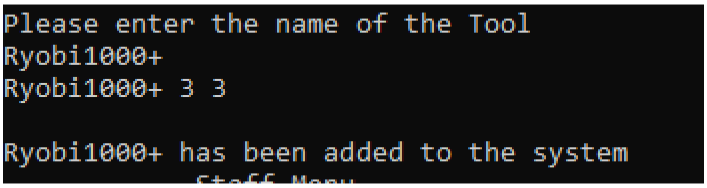
As you can see when the 0 key is entered in the member menu it takes the user back to login and when entered on the login page it takes the user out of the program.

Staff menu checks:

Test 1. Checking that the add tool function works

Desired result: When the add tool, function is chosen from the staff menu it reads an input for the tool name and then adds the tool into the chosen tool array.

Result:



```
Please enter the name of the Tool
Ryobi1000+
Ryobi1000+ 3 3

Ryobi1000+ has been added to the system
```

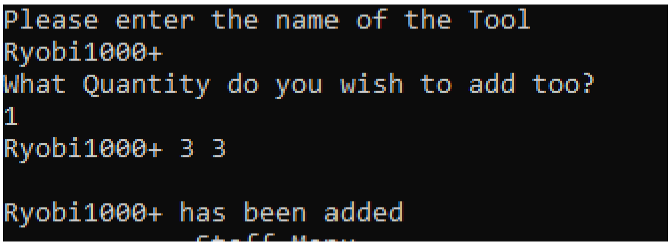
As shown the add tool function does indeed add the tool into the specified tool array and this can be shown by both the print line of the tool array or the message down the bottom.

Test 2. Checking that the add new pieces of an existing tool function works

Desired result: When the add tool pieces of an existing tool function is chosen the menu will ask for a quantity and a tool name and then add that quantity of pieces tool the already existing tool, if the tool does not exist it will throw an error.

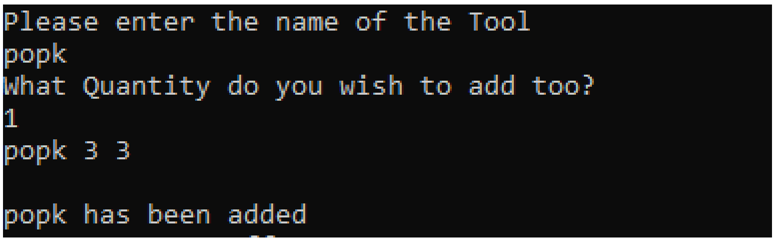
Result:

Valid tool



```
Please enter the name of the Tool
Ryobi1000+
What Quantity do you wish to add too?
1
Ryobi1000+ 3 3

Ryobi1000+ has been added
```

Invalid tool



```
Please enter the name of the Tool
popk
What Quantity do you wish to add too?
1
popk 3 3

popk has been added
```

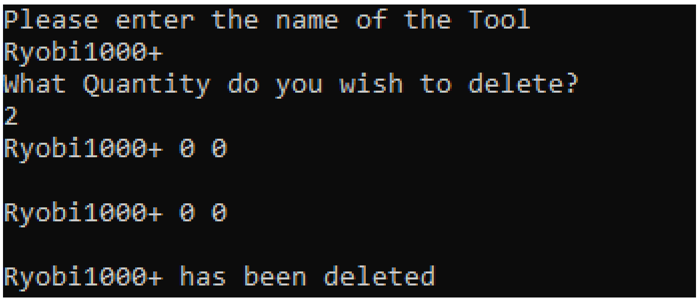
As shown by the results above the add piece's function does not operate as intended as in the valid tool test it just adds a new tool of the same name instead of adding pieces of a tool to the array and in the invalid tool test it once again just adds a new tool with the invalid name into the tool array. It also does not bring up any error message.

Test 3. Checking that the remove some pieces of an existing tool function works

Desired result: when the remove pieces function is called and a valid tool is entered pieces of that tool will be removed, when an invalid tool is entered the program will show an error message.

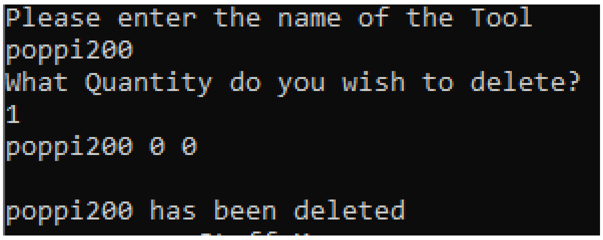
Result:

Valid tool



```
Please enter the name of the Tool
Ryobi1000+
What Quantity do you wish to delete?
2
Ryobi1000+ 0 0

Ryobi1000+ 0 0

Ryobi1000+ has been deleted
```

Invalid tool



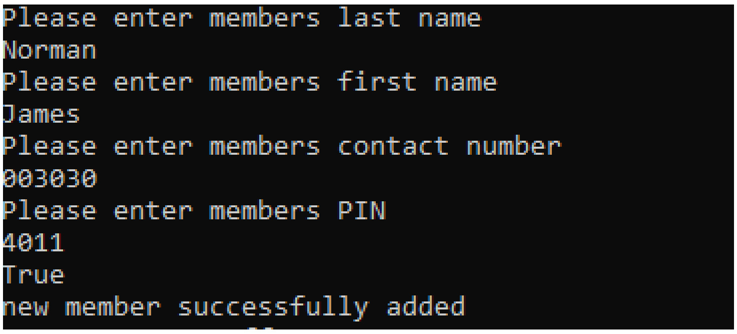
```
Please enter the name of the Tool
poppi200
What Quantity do you wish to delete?
1
poppi200 0 0

poppi200 has been deleted
```

As shown in the results the delete piece's function does not work as much like above when parsed a valid tool it just creates a new tool with the same name that contains 0 pieces and when it is passed an invalid tool no error is shown and once again it creates a new tool with 0 pieces.

Test 4. Checking that the Register new member function works

Desired result: when the register new member function is chosen it will bring up the prompt to enter the users' details and then add the user to the system.

Result:



```
Please enter members last name
Norman
Please enter members first name
James
Please enter members contact number
003030
Please enter members PIN
4011
True
new member successfully added
```

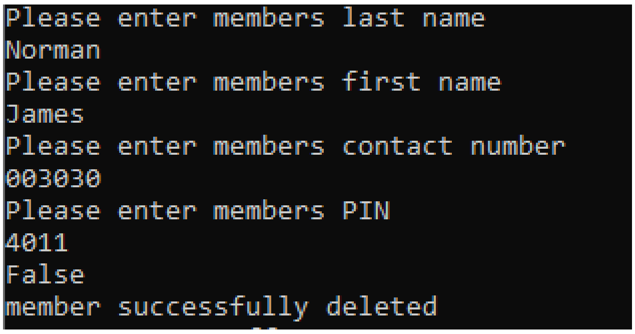
As shown the add member function work as the members details are entered and the search member function is called at the end to confirm that the member is in the tree and then a confirmation message is shown.

Test 5. Checking that the delete member function works

Desired result: when the delete member, function is chosen it will bring up the prompt to enter the users' details and then delete the user to the system if the member is valid and if the member is invalid it will raise an error that the user is not in the system.

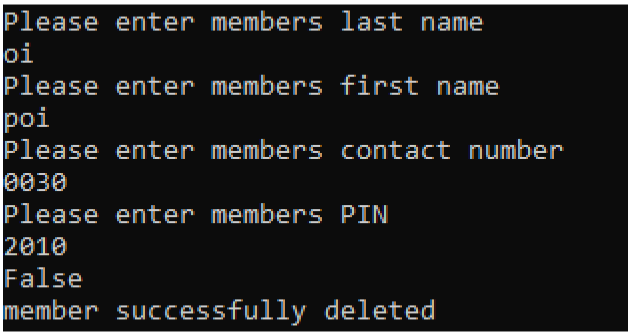
Result:

Valid



```
Please enter members last name
Norman
Please enter members first name
James
Please enter members contact number
003030
Please enter members PIN
4011
False
member successfully deleted
```

Invalid



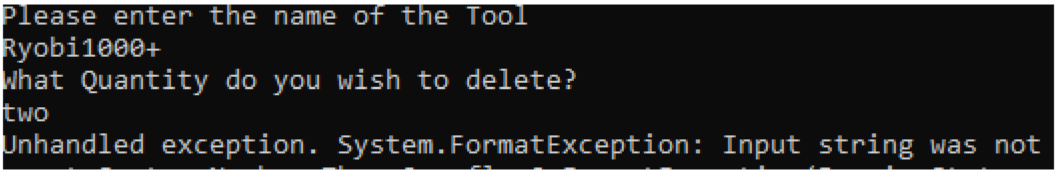
```
Please enter members last name
oi
Please enter members first name
poi
Please enter members contact number
0030
Please enter members PIN
2010
False
member successfully deleted
```

As shown in the tests above the delete member function does not work as intended as the bool to search for the member returns false even with a valid member meaning the member has not been deleted yet the confirmation still shows up, the invalid test also does not work as it does not show an error.

Test 6. Checking that invalid input into the add and remove pieces of a tool function give an error.

Desired result: when the add or remove piece’s function is chosen in the area where the program asks for a quantity If the user parses anything that is not an int the program gives an error warning.

Result



```
Please enter the name of the Tool
Ryobi1000+
What Quantity do you wish to delete?
two
Unhandled exception. System.FormatException: Input string was not
```

As shown the string was not handled properly and causes the program to shut down.

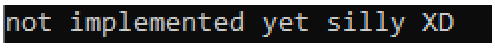
Tests to handle the input for the other functions are not needed as all of their inputs are strings which is read line () standard form.

Member menu checks:

Test 1. Display all tools of a tool type

Desired result: when a tool type is selected the tools that are currently in that tool type are shown.

Result



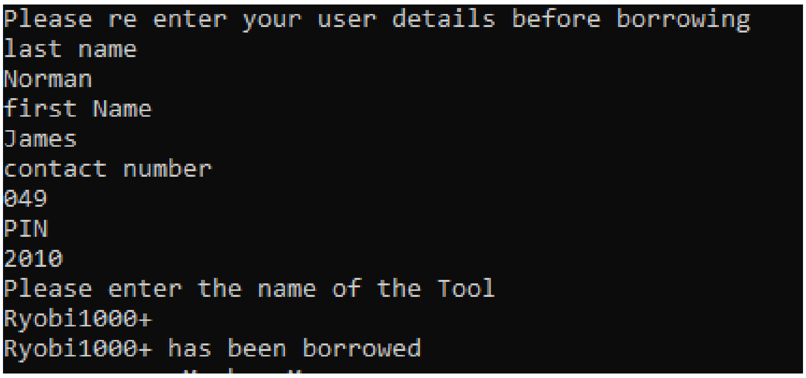
```
not implemented yet silly XD
```

As shown the above function has not yet been implemented

Test 2. Borrow a tool

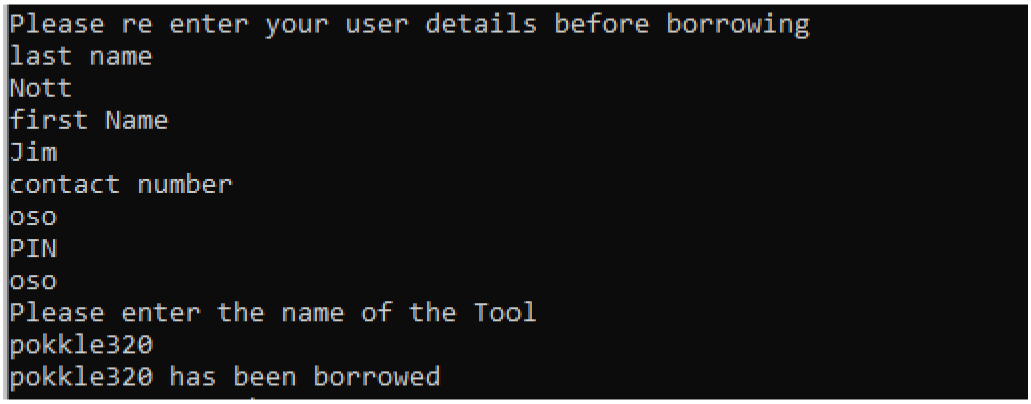
Desired result: When the Borrow tool function is selected the program asks the user to re-enter their details and then the name of the tool they wish to borrow and then borrows the tool. When the tool is not valid It gives the user an error message.

Valid



```
Please re enter your user details before borrowing
last name
Norman
first Name
James
contact number
049
PIN
2010
Please enter the name of the Tool
Ryobi1000+
Ryobi1000+ has been borrowed
```

Invalid



```
Please re enter your user details before borrowing
last name
Nott
first Name
Jim
contact number
oso
PIN
oso
Please enter the name of the Tool
pokkle320
pokkle320 has been borrowed
```

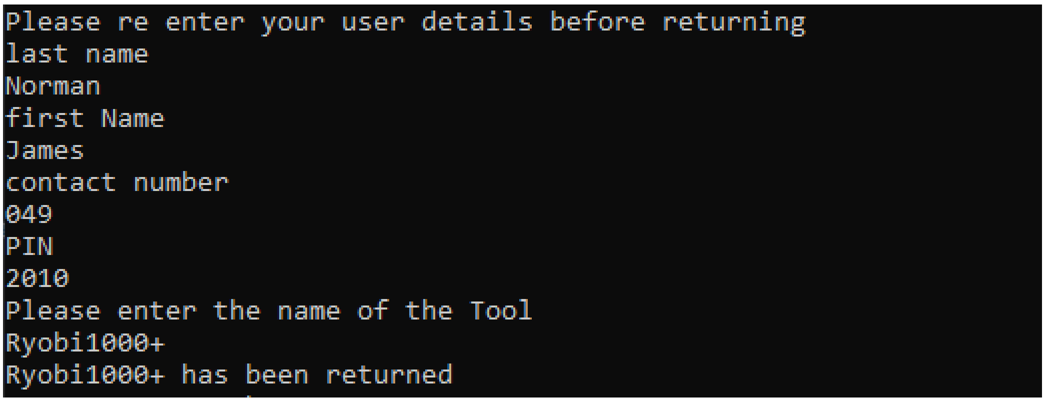
As seen in the results the program works when the member borrows a valid tool but does not work when the user enters in the wrong member details or the wrong tool as neither of these errors show any error message and instead show the success message.

Test 3. Return a tool

Desired result: when the return a tool function is selected the user is prompted to re-enter their details and then the name of the tool that they are returning and when that is valid, they will receive a message that their tool has been returned. When either the member or tool is invalid an error message will appear.

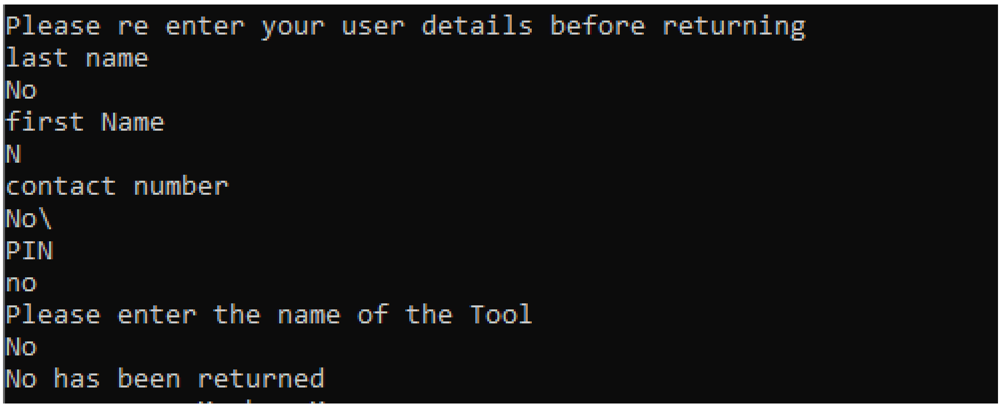
Result:

Valid



```
Please re enter your user details before returning
last name
Norman
first Name
James
contact number
049
PIN
2010
Please enter the name of the Tool
Ryobi1000+
Ryobi1000+ has been returned
```

Invalid



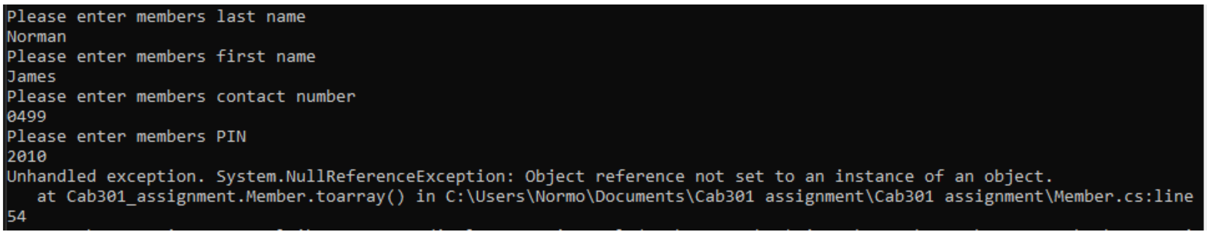
```
Please re enter your user details before returning
last name
No
first Name
N
contact number
No\
PIN
no
Please enter the name of the Tool
No
No has been returned
```

As shown in the results the valid entrance works as the user is shown the tool has been returned message but in the invalid input nothing

Test 4. List all tools the user is currently renting

Desired result: when the list tools currently borrowing function is chosen the program will prompt the user to re-enter their user details and then will present them with a list of the tools they are currently borrowing.

Result



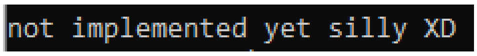
```
Please enter members last name
Norman
Please enter members first name
James
Please enter members contact number
0499
Please enter members PIN
2010
Unhandled exception. System.NullReferenceException: Object reference not set to an instance of an object.
   at Cab301_assignment.Member.toarray() in C:\Users\Normo\Documents\Cab301 assignment\Cab301 assignment\Member.cs:line
54
```

As seen by the result the function does not work due to the unhandled exception.

Test 5. Show top three tools

Desired result: when the top three tools function is chosen the program will display the 3 most popular tools.

Result



```
not implemented yet silly XD
```

As seen by the result the function has not been implemented.

Conclusion:

To conclude this report has taken a look at the algorithm design, algorithm analysis and the test plan and test results from the assessment task. Throughout the assessment the Tool Library application has been created using various data structures and algorithms.