CAB301 ALGORITHMS AND COMPLEXITY

Assignment 3

**Development of a Community Library**

**Movie DVDs Management Software**

**Group 22**

|  |  |
| --- | --- |
| **Team Member** | **Student Number** |
| Dulce Acevedo | N10370021 |
| Anthony Nguyen | N10625119 |
| Rodo Nguyen | N10603280 |

# Table of contents

[Table of contents 2](#_Toc105773834)

[1. Introduction 3](#_Toc105773835)

[2. Design and Analysis of an algorithm 4](#_Toc105773836)

[2.1 Pseudocode (Task B) 4](#_Toc105773837)

[2.2 Theoretical Analysis 4](#_Toc105773838)

[2.3 Empirical Analysis 5](#_Toc105773839)

[3. Software Test Plan and Test Results 7](#_Toc105773840)

[3.1 Test Plan 7](#_Toc105773841)

[3.2 Test Results 8](#_Toc105773842)

[3.2.1 Main Menu 8](#_Toc105773843)

[a) Staff Login Menu 9](#_Toc105773844)

[b) Member Login Menu 11](#_Toc105773845)

[3.2.2 Staff Menu 13](#_Toc105773846)

[Menu Option 0 14](#_Toc105773847)

[Menu Option 1 14](#_Toc105773848)

[Menu Option 2 19](#_Toc105773849)

[Menu Option 3 21](#_Toc105773850)

[Menu Option 4 24](#_Toc105773851)

[Menu Option 5 25](#_Toc105773852)

[Menu Option 6 26](#_Toc105773853)

[3.2.3 Member Menu 28](#_Toc105773854)

[Menu Option 0 29](#_Toc105773855)

[Menu Option 1 30](#_Toc105773856)

[Menu Option 2 31](#_Toc105773857)

[Menu Option 3 32](#_Toc105773858)

[Menu Option 4 34](#_Toc105773859)

[Menu Option 5 35](#_Toc105773860)

[Menu Option 6 36](#_Toc105773861)

# 1. Introduction

The goal of this project is to develop of a community library movie DVDs management software. This consists of designing effective algorithms to manage the library’s members, movies, and DVDs, and creating a console-based user interface to easily interact with.

In addition, an algorithm that gets the three greatest items from an array is requested. So, we have designed, analysed its efficiency by using empirical analysis approach and implemented it in the software as well.

The project has been split into three different parts, part one is about creating our objects for members, a way to store our members in a collection and methods to be able to interact with the member collection such as adding, removing, and listing information about a member in the member collection.

In part two, we created movie objects, movie collections and methods to be able to interact with the movie collection while using previous methods from part one.

And in part three, we combine both parts into a usable GUI and add further logics to the existing code to make a complete movie DVDs management software.

# 2. Design and Analysis of an algorithm

## 2.1 Pseudocode (Task B)

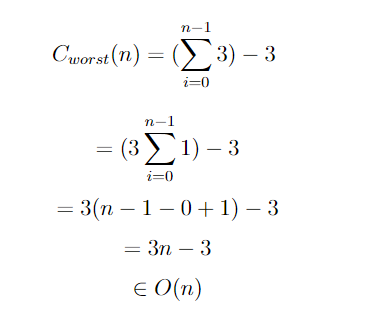
Text

Description automatically generated with medium confidence

## 2.2 Theoretical Analysis

The parameter characterising the size of the problem is determined by the length n of array A. The basic operation is the comparison between an element in A with an element in T. A basic operation occurs in three places in the algorithm*: i > T[0], i >T[1], i > T[2].* A different number of basic operations will occur depending on the elements of the array and its position.

In the worst-case scenario, the elements in array A are in descending order and they are all greater than or equal to 0. In this case, the first iteration will go through only the first basic operation. The second iteration will go through the first 2 basic operations. The rest of the iterations will reach all 3 basic operations. Thus, we have:



The efficiency of this algorithm is O(n).

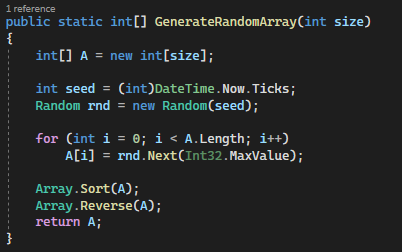
## 2.3 Empirical Analysis

An empirical analysis is performed with the purpose of finding the time efficiency of the algorithm by analysing its worst case and checking the accuracy of our initial theoretical assertion about the efficiency.

The basic operation identified is the comparison between an element in A with an element in T. A basic operation occurs in three places in the algorithm*: i > T[0], i >T[1], i > T[2].* A different number of basic operations will occur depending on the elements of the array and its position.

The efficiency metric used is the number of times the algorithm's basic operation is executed. Thus, the program increases ‘count’ by 1 every time the identified basic operation is executed.

The input is a randomly generated array of descending integers, none of which is less than 0. The sizes of the array are 1000, 2000, 3000, …, 20000; so, there are 20 runs in total. The experiment involves running a program in C#. Below is the function to generate our array:



The results of our test are:

|  |  |
| --- | --- |
| **Size** | **Basic operation count** |
| 1000 | 2997 |
| 2000 | 5997 |
| 3000 | 8997 |
| 4000 | 11997 |
| 5000 | 14997 |
| 6000 | 17997 |
| 7000 | 20997 |
| 8000 | 23997 |
| 9000 | 26997 |
| 10000 | 29997 |
| 11000 | 32997 |
| 12000 | 35997 |
| 13000 | 38997 |
| 14000 | 41997 |
| 15000 | 44997 |
| 16000 | 47997 |
| 17000 | 50997 |
| 18000 | 53997 |
| 19000 | 56997 |
| 20000 | 59997 |

Let the efficiency function t(n) = cg(n), where c is a constant and g(n) is a simple function. Firstly, we calculate the ratio of t(2n) and t(n) using the above experimental results:

Therefore, we have:

Because , which is exactly what we have in (1)

So, the efficiency function t(n) = cn belongs to O(n). This shows that our initial assertion about the algorithm efficiency in section 2.2 Theoretical Analysis is accurate.

# 3. Software Test Plan and Test Results

## 3.1 Test Plan

|  |  |  |
| --- | --- | --- |
| **Menu** | **Menu Option** | **Functionality to Test** |
| Main Menu |  | * Choose an action |
| Staff Login Sub-Menu |  | * Login as staff |
| Member Login Sub-Menu |  | * Login as member |
| Staff Menu |  | * Choose an action |
| 0 | * Return to the main menu |
| 1 | * Add a new movie if a new DVD movie is not in the system. * Add DVDs to a movie that already exists |
| 2 | * Remove a number of DVD copies from a movie that is less than the number of copies available. * Delete the movie from the system if all the copies are deleted. |
| 3 | * Register a new member |
| 4 | * Remove a registered member from the system |
| 5 | * Display a member's contact phone number, given the member's name |
| 6 | * Display all members who are currently renting a particular movie |
| Member Menu |  | * Choose an action |
| 0 | * Return to the main menu |
| 1 | * Display information about all movie DVDs in the library in alphabetical order, including the number of the movie DVDs currently available. |
| 2 | * Display all the information about a movie, given the movie's title |
| 3 | * Borrow a movie DVD |
| 4 | * Return a movie DVD |
| 5 | * Display List current movies that are currently borrowed by the registered member |
| 6 | * Display the top three (3) most frequently borrowed movies by members in the descending order of their frequency. The display should include the title of the movie and frequency (the number of times that the movie has been borrowed by registered members by now) |

## 3.2 Test Results

## 3.2.1 Main Menu

**Functionality Under Test**: Choose Action

|  |  |
| --- | --- |
| Test | Enter a valid input |
| Test Data | * Input: 1 (the same for 0 and 2) |
| Expected Result | The Staff Login Menu appears |
| Actual Result |  |
| Test Passed | | |

|  |  |
| --- | --- |
| Test | Enter an invalid input (an integer) |
| Test Data | * Input: 4 |
| Expected Result | No action is selected. Warning Message is displayed.  Refresh the menu. |
| Actual Result |  |
| Test Passed | | |

|  |  |
| --- | --- |
| Test | Enter an invalid input (string) |
| Test Data | * Input: “\n” |
| Expected Result | No action is selected. Warning Message is displayed.  Refresh the menu. |
| Actual Result |  |
| Test Passed | |

### a) Staff Login Menu

**Functionality Under Test**: Login as staff

|  |  |
| --- | --- |
| Test | Enter correct Password and Username |
| Test Data | * Username: staff * Password: today123 |
| Expected Result | The Staff Menu appears |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter wrong Password |
| Test Data | * Username: staff * Password: today |
| Expected Result | Login is unsuccessful. A warning message is displayed.  The login form is refreshed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Unsuccessful Login: Wrong Username |
| Test Data | * Username: STAFF * Password: today |
| Expected Result | Login is unsuccessful. A warning message is displayed.  The login form is refreshed. |
| Actual Result |  |
| Test Passed | |

### b) Member Login Menu

**Functionality Under Tes**t: Login as member

|  |  |
| --- | --- |
| Test | Enter correct password and names of a register member |
| Test Data | * First name: Anthony * Last Name: Nguyen * Password: 1234 |
| Expected Result | The Member menu appears. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an incorrect Password |
| Test Data | * First name: Anthony * Last Name: Nguyen * Password: 123 |
| Expected Result | Login is unsuccessful. Warning Message is displayed.  The login form is refreshed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an incorrect First Name |
| Test Data | * First name: Anthonyyyy * Last Name: Nguyen * Password: 123 |
| Expected Result | Login is unsuccessful. Warning Message is displayed.  The login form is refreshed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an incorrect Last Name |
| Test Data | * First name: Anthony * Last Name: Nguyennnnn * Password: 123 |
| Expected Result | Login is unsuccessful. Warning Message is displayed.  The login form is refreshed. |
| Actual Result |  |
| Test Passed | |

## 3.2.2 Staff Menu

**Functionality Under Test**: Choose Action

|  |  |
| --- | --- |
| Test | Enter a valid choice (integers from 0 to 6) |
| Test Data | * Choice: 1 |
| Expected Result | An according form appears. In this test case, “Add new DVDs” form appears |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an invalid choice (an integer) |
| Test Data | * Choice: 7 |
| Expected Result | No menu option was selected. Warning message is displayed.  The form is refreshed. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 0

**Functionality Under Test**: Return to menu

|  |  |
| --- | --- |
| Test | Return to main menu |
| Test Data | * Choice: 0 |
| Expected Result | The main menu is displayed |
| Actual Result |  |
| Test Passed | |

#### Menu Option 1

**Functionality Under Test**: Add a new movie if a new DVD movie is not in the system.

|  |  |
| --- | --- |
| Test | Add a new movie with valid information to the system. |
| Test Data | * Title: Movie A * Genre: 1 * Classification: 2 * Duration: 100 * Copies: 8 |
| Expected Result | Movie is added. Successful message will be displayed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an invalid genre choice (not a number between 1-5, other string) |
| Test Data | * Title: Hi * Genre: 6, 13i |
| Expected Result | Warning message is displayed.  User will be asked to re-enter their choice of movie genre. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an invalid movie classification choice (not a number between 1 and 4, other string) |
| Test Data | * Title: Movie B * Genre: 4 * Classification: 5, PG |
| Expected Result | Warning message is displayed.  User will be asked to re-enter their movie classification choice. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter invalid input for movie duration (integer that <= 0, a string). |
| Test Data | * Title: Movie B * Genre: 4 * Classification: 3 * Duration: 0, -99, hello |
| Expected Result | Warning message is displayed.  User will be asked to re-enter the movie duration. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter invalid input for number of copies (integer that <= 0, a string) |
| Test Data | * Title: Movie B * Genre: 4 * Classification: 3 * Duration: 120 * Copies: abc, -1, 0 |
| Expected Result | Warning message is displayed. User will be asked to re-enter the number of copies. |
| Actual Result |  |
| Test Passed | |

**Functionality Under Test**: Add DVDs to a movie that already exists

|  |  |
| --- | --- |
| Test | Enter a valid input for the number of addition DVDs copies |
| Test Data | * Title: Movie 1 (with current number of copies: 2) * Number of DVDs to add: 3 |
| Expected Result | The new total number of DVD copies (5) is displayed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter invalid input for the number of DVDs to add (integer that <= 0, a string) |
| Test Data | * Title: Movie 1 * Total copies: 1 * Number of DVDs to add: abc, -2, 0 |
| Expected Result | Warning message is displayed. User will be asked to re-enter the number of copies. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 2

**Functionality Under Test**: Remove a number of DVD copies from a movie

|  |  |
| --- | --- |
| Test | Enter a number less than the number of DVDs available was removed. |
| Test Data | * Title: Movie 4 (total copies: 3, available copies: 3) * Number of DVDs to delete: 2 |
| Expected Result | DVD copies of the movie is successfully removed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an invalid input for the number of DVD copies to be removed (integer <= 0 or greater than the number of available DVDs, string) |
| Test Data | * Title: Movie 1 (total copies: 10, available copies: 10) * Number of DVDs to delete: -3, ninety-nine, 99 |
| Expected Result | Warning message is displayed.  User will be asked to re-enter their input.  The form is stopped if a positive integer is entered and a message notifying successful/unsuccessful action. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an existed Movie. |
| Test Data | * Title: NOT A MOVIE |
| Expected Result | Warning message will display: “DVD copies of NOT A MOVIE cannot be deleted because that movie is not in the database”. |
| Actual Result |  |
| Test Passed | |

**Functionality Under Test:** Delete the movie from the system if all the copies are deleted.

|  |  |
| --- | --- |
| Test | The movie is deleted when all of its DVD copies are not currently borrowed and are deleted. |
| Test Data | * Title: Movie 2 (total copies: 2, available copies: 2) * Number of DVDs to delete: 2 |
| Expected Result | Movie successfully deleted |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Unable to remove a movie if some of its DVDs are currently borrowed. |
| Test Data | * Title: Movie 4 * Total copies: 10 * Available copies: 8 * Number of DVDs to delete: 10 |
| Expected Result | Movie was not deleted. Warning message will display. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 3

**Functionality Under Test**: Register a new member

|  |  |
| --- | --- |
| Test | Add a member with valid input information. |
| Test Data | * First Name: Tom * Last Name: Math * Phone Number: 0123123123 * Pin: 1231 |
| Expected Result | Member is added to the database.  The message “Member is successfully added” is displayed |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an invalid phone Number (phone number does not start with zero and not have 10 digits, phone number is not an integer) |
| Test Data | * First Name: Tim * Last Name: Lardy * Phone Number: 1234567891, 123, string |
| Expected Result | Phone number entered will not be accepted.  Warning message is displayed.  User will be asked to re-enter their phone number. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an invalid PIN. (Pin does not have a length of 4-6 digits, pin is not an integer) |
| Test Data | * First Name: Tim * Last Name: Lardy * Phone Number: 0123456789 * Pin: 1234567, 123, stringPIN |
| Expected Result | Pin entered will not be accepted.  Warning message is displayed.  User will be asked to try re-enter their PIN. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Add an existed member. |
| Test Data | * First Name: Tim * Last Name: Lardy * Phone Number: 0123456789 * Pin: 1234 |
| Expected Result | Member will not be added  A message is displayed to inform user that this member is existed |
| Actual Result |  |
| Test Passed | |

#### Menu Option 4

**Functionality Under Test:** Remove a registered member from the system

|  |  |
| --- | --- |
| Test | Delete an existed member. |
| Test Data | * First Name: Dulce * Last Name: Acevedo * Movies Borrowing: None |
| Expected Result | Member will be deleted successfully.  An according message is also displayed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Delete a member currently borrowing DVDs. |
| Test Data | * First Name: Anthony * Last Name: Nguyen * Movies Borrowing: 2 |
| Expected Result | Member is not deleted because they are currently borrowing movies. An according message is also displayed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Delete a non-existed member |
| Test Data | * Member that are not in the system. * First Name: Angelina * Last Name: Jolie |
| Expected Result | Member cannot be deleted as they do not exist.  An according message is also displayed. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 5

**Functionality Under Test**: Display a member's contact phone number, given the member's name

|  |  |
| --- | --- |
| Test | Display an existed member’s phone number. |
| Test Data | * First Name: Dulce * Last Name: Acevedo |
| Expected Result | The member’s phone number is displayed |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Display a non-existed member’s phone number. |
| Test Data | * First Name: Brad * Last Name: Pitt |
| Expected Result | Number will not be displayed as the member does not exist.  An according message is also displayed. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 6

**Functionality Under Test**: Display all members who are currently renting a particular movie

|  |  |
| --- | --- |
| Test | Display all borrowers of a movie which is borrowed by some members |
| Test Data | * Title: Movie 4 * Borrowers: Dulce Acevedo, Rodo N and Anthony Nguyen. |
| Expected Result | The current borrowers of the movie will be successfully displayed |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Display all borrowers of a movie which is not borrowed by any member |
| Test Data | * Title: Movie 1 * No Borrowers: 0 |
| Expected Result | No borrower is displayed. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Display all borrowers of a non-existed movie |
| Test Data | * Movie name entered: Advengers (not existed in the system) |
| Expected Result | No borrowers will be displayed. |
| Actual Result |  |
| Test Passed | |

## 3.2.3 Member Menu

**Functionality Under Test**: Choose Action

|  |  |
| --- | --- |
| Test | Enter a valid choice (an integer from 0 to 6) |
| Test Data | * Choice: 3 |
| Expected Result | Corresponding sub-menu is displayed. In this case, it’s “Borrow a movie”. |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Enter an invalid choice (an integer not from 0 to 6, string) |
| Test Data | * Choice: 019, lalala |
| Expected Result | Warning Message is displayed followed by refreshed member menu |
| Actual Result |  |
| Test Passed | |

#### Menu Option 0

**Functionality Under Test**: Return to menu

|  |  |
| --- | --- |
| Test | Return to main menu |
| Test Data | * Choice: 0 |
| Expected Result | The main menu will be displayed |
| Actual Result |  |
| Test Passed | |

#### Menu Option 1

**Functionality Under Test**: Display information about all movie DVDs in the library in alphabetical order, including the number of the movie DVDs currently available.

|  |  |
| --- | --- |
| Test | Display all movies’ information |
| Test Data | * Title: Movie 1, Genre: Western, Classification: G, Duration: 101, Total Copies: 1, Available copies: 1 * Title: Movie 2, Genre: Action, Classification: G, Duration: 102, Total Copies: 2, Available copies: 1 * Title: Movie 3, Genre: History, Classification: PG, Duration:103, Total Copies: 5, Available copies:2 * Title: Movie 4, Genre: History, Classification: PG, Duration: 104, Total Copies: 10, Available copies:7 |
| Expected Result | All movies are successfully display with all the information |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Display all movies’ information while no movie is added |
| Test Data | none |
| Expected Result | A message is displayed to inform user that no movie is available. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 2

**Functionality Under Test**: Display all the information about a movie, given the movie's title.

|  |  |
| --- | --- |
| Test | Display a valid movie’s information |
| Test Data | * Title: Movie 1, Genre: Western, Classification: G, Duration: 101, Total Copies: 0 * Movie name entered: Movie 1 |
| Expected Result | The information of the movie with that name was displayed |
| Actual Result |  |
| Test Passed | |

Empty test

|  |  |
| --- | --- |
| Test | Display a non-existed movie |
| Test Data | * Title: NOT A MOVIE |
| Expected Result | The information of the movie with that name will not be displayed because it does not exist. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 3

**Functionality Under Test**: Borrow a movie DV

|  |  |
| --- | --- |
| Test | Borrow a valid movie |
| Test Data | * Title: Movie 3, Total Copies: 5, Available copies:2 * Movie name entered: Movie 3 |
| Expected Result | The member successfully borrowed the movie selected |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Borrow a movie twice |
| Test Data | * Title: Movie 3, Total Copies: 5, Available copies:2 * Movie name entered: Movie 3 * Member already has a copy borrowed |
| Expected Result | The member cannot borrow the movie selected because they already did. An according message is displayed |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Borrowing a movie with no available copy left |
| Test Data | * Title: Movie 1, Total Copies: 1, Available copies: 0 * Movie name entered: Movie 1 |
| Expected Result | The member cannot borrow the movie selected because there is no copy available |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Borrow a non-existed movie |
| Test Data | * Movie name entered: NOT A MOVIE |
| Expected Result | The member cannot borrow the movie selected because the movie does not exist |
| Actual Result |  |
| Test Passed | |

#### Menu Option 4

**Functionality Under Test**: Return a movie DVD

|  |  |
| --- | --- |
| Test | Return a valid movie |
| Test Data | * Member has a copy of Movie 1 * Movie name entered: Movie 1 |
| Expected Result | The copy of the movie was successfully returned |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Return a DVD that was not borrowed before |
| Test Data | * Member does not have a copy of Movie 2. * Movie name entered: Movie 2 |
| Expected Result | No copy of the movie selected was returned as the member does not have any copy |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | Return a non-existed movie |
| Test Data | * Movie name entered: NOT A MOVIE * That movie does not exist in the system |
| Expected Result | No copy of the movie selected was returned as the movie does not exist |
| Actual Result |  |
| Test Passed | |

#### Menu Option 5

**Functionality Under Test**: Display List current movies that are currently borrowed by the registered member.

|  |  |
| --- | --- |
| Test | List all borrowing movies |
| Test Data | * Movies being borrowed by the member and the information:   + >>> Movie: Movie 1, Western, G, 101, 0   + >>> Movie: Movie 4, History, PG, 104, 7 |
| Expected Result | All movies that the member is currently borrowing are displayed |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | List the borrowing movies while the member is not currently borrowing any movies |
| Test Data | * Member is not currently borrowing any movies |
| Expected Result | < Empty > is printed to the console. |
| Actual Result |  |
| Test Passed | |

#### Menu Option 6

**Functionality Under Test**: Display the top three (3) most frequently borrowed movies by the members in the descending order of their frequency. The display should include the title of the movie and frequency (the number of times that the movie has been borrowed by registered members by now)

|  |  |
| --- | --- |
| Test | List the top 3 most frequently borrowed movies |
| Test Data | * Movie 1( borrowed 0 times) * Movie 3 (borrowed 3 times) * Movie 4 (borrowed 4 times) * Movie 2 (borrowed 0 times) |
| Expected Result | Top three most frequently borrowed movies displayed |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | List the top 3 most frequently borrowed movies while there are less than three movies available |
| Test Data | * Movie 1( borrowed 1 times) * Movie 4 (borrowed 3 times) * No more movies available |
| Expected Result | It will show not available for the third top 3, since there are only 2 movies that have been borrowed |
| Actual Result |  |
| Test Passed | |

|  |  |
| --- | --- |
| Test | List the top 3 most frequently borrowed movies while some movies have the same numbers of borrowed frequencies. |
| Test Data | * Movie 3 (borrowed 3 times) * Movie 4 (borrowed 3 times) * Movie 2 (borrowed 1 times) * Movie 1 (borrowed 0 times) |
| Expected Result | Top three most frequently borrowed movies will be displayed. They will be displayed in alphabetical/dictionary/lexicographic if some movies have the same borrowed frequency.  Order: Top 1: Movie 3, Top 2: Movie 4, Top 3: Movie 2. |
| Actual Result |  |
| Test Passed | |