CAB301 PROJECT

Library Software to develop to manage

Name: Phuong Nam Ly
Student number: n10098097

Table of Contents

Algorithm Analysis

The overall complexity of DisplayTopTenMovies() is O(nlogn). Each individual functions within DisplayTopTenMovies() will be analysed to support the overall time complexity.

Algorithm DisplayTopTenMovies()

// Initialize a set of sorting data, specifically for displaying top ten movies

// Fill the sorting data by traveling through the binary search tree, through in-order traversal

// use the merge sort algorithm to sort the sorting data in a descending order in terms of borrowed times

// display the top ten most popular movies

SortingData sortingData = new SortingData(); // ϵ O(1)

InOrder(sortingData); // ϵ O(n)

Sorting.mergeSort(sortingData.MovieNum(), sortingData.BorrowedTimesArray(), sortingData.MovieArray()); // ϵ O(nlogn)

sortingData.DisplayMoviesInOrder(); // ϵ O(1)

Analysis

The time complexity of SortingData() is O(1); InOrder() is O(n); mergeSort() is O(nlogn); DisplayMoviesInOrder() is O(n);

The overall time complexity of DisplayTopTenMovies() is:

nlogn + n + 1 + 1 ϵ O(nlogn)

Algorithm SortingData()

// This constructor initializes empty movie array and int array, and set the number of movies to be 0

// the size of arrays is chosen to be an arbitrary number, 100. This number can be changed if the number of movies exceed 100.

movieArray[0,99] <- $\emptyset$ // movieArray is an array of movie, initially, the array is empty

borrowedTimesArray[0,99] <- $\emptyset$ // this array is an array of movieArray's borrowed times, initially, the array is empty

numberOfMovies <- 0

Analysis:

new[] in C# does not only allocated but also initializes. Therefore, for each of the array initialization, the time complexity is O(n) for array size n. With the size being 100, the time complexity is O(100) for the array initialization. There's no need to separate the cases since this algorithm does not depend on any input.

$C(n) = 100 + 100 + 1 \in O(1)$

Algorithm InOrder(SortingData sortingData)

// This function is from the Binary Search Tree class

// It calls the in-order traversal function from the root so that the traversal starts from the root

root.InOrderTraversal(sortingData)

The size of the input is n(T), the number of nodes within the tree.

Algorithm InOrderTraversal(SortingData sortData)

// This function is from the Node class

// data represents the movie the node contains

if Tleft != Ø

Tleft.InOrderTraversal

// A list of setters for sortData

sortData.setMovie(movieNum, data); // set a movie within the movie array to be data

sortData.setBorrowedTimes(movieNum, data.BorrowedTimes()); // set an int within the borrowed times array to be the borrowed times of the movie

sortData.setMovieNum(1); // increase the number of the movie by one

if Tright != Ø

Tright.InOrderTraversal

Analysis:

The size of the input is n(T), the number of nodes within the tree. Tleft != Ø and Tright != Ø are chosen as the basic operations in this case. The number of comparisons needed to check

whether a node in a tree is empty or not is C(n) = 2n + 1, as described in the textbook BSTree analysis (Levitin, 2011, p.209; p.210).

There is no need to separate average-case, worst-case and best-case because this algorithm will always make 2n + 1 comparisons for every input size of n node.

$C(n) = 2n + 1 \in O(n)$

Analysis from the textbook: "As to their [traversal] efficiency analysis, it is identical to the above analysis of the Height algorithm because a recursive call is made for each node of an extended binary tree." (Levitin, 2011, p.210).

ALGORITHM *Height*(T)

//Computes recursively the height of a binary tree
//Input: A binary tree T
//Output: The height of T
if $T = \varnothing$ **return** -1
else return $\max\{Height(T_{left}), Height(T_{right})\} + 1$

We measure the problem's instance size by the number of nodes $n(T)$ in a given binary tree T. Obviously, the number of comparisons made to compute the maximum of two numbers and the number of additions $A(n(T))$ made by the algorithm are the same. We have the following recurrence relation for $A(n(T))$:

$$A(n(T)) = A(n(T_{left})) + A(n(T_{right})) + 1 \quad \text{for } n(T) > 0,$$
$$A(0) = 0.$$

Before we solve this recurrence (can you tell what its solution is?), let us note that addition is not the most frequently executed operation of this algorithm. What is? Checking—and this is very typical for binary tree algorithms—that the tree is not empty. For example, for the empty tree, the comparison $T = \varnothing$ is executed once but there are no additions, and for a single-node tree, the comparison and addition numbers are 3 and 1, respectively.

It helps in the analysis of tree algorithms to draw the tree's extension by replacing the empty subtrees by special nodes. The extra nodes (shown by little squares in Figure 5.5) are called ***external***; the original nodes (shown by little circles) are called ***internal***. By definition, the extension of the empty binary tree is a single external node.

It is easy to see that the *Height* algorithm makes exactly one addition for every internal node of the extended tree, and it makes one comparison to check whether

the tree is empty for every internal and external node. Therefore, to ascertain the algorithm's efficiency, we need to know how many external nodes an extended binary tree with n internal nodes can have. After checking Figure 5.5 and a few similar examples, it is easy to hypothesize that the number of external nodes x is always 1 more than the number of internal nodes n:

$$x = n + 1. \tag{5.2}$$

To prove this equality, consider the total number of nodes, both internal and external. Since every node, except the root, is one of the two children of an internal node, we have the equation

$$2n + 1 = x + n,$$

which immediately implies equality (5.2).

Note that equality (5.2) also applies to any nonempty ***full binary tree***, in which, by definition, every node has either zero or two children: for a full binary tree, n and x denote the numbers of parental nodes and leaves, respectively.

Returning to algorithm *Height*, the number of comparisons to check whether the tree is empty is

$$C(n) = n + x = 2n + 1,$$

and the number of additions is

$$A(n) = n.$$

The most important divide-and-conquer algorithms for binary trees are the three classic traversals: preorder, inorder, and postorder. All three traversals visit nodes of a binary tree recursively, i.e., by visiting the tree's root and its left and right subtrees. They differ only by the timing of the root's visit:

In the ***preorder traversal***, the root is visited before the left and right subtrees are visited (in that order).

In the ***inorder traversal***, the root is visited after visiting its left subtree but before visiting the right subtree.

In the ***postorder traversal***, the root is visited after visiting the left and right subtrees (in that order).

These traversals are illustrated in Figure 5.6. Their pseudocodes are quite straightforward, repeating the descriptions given above. (These traversals are also a standard feature of data structures textbooks.) As to their efficiency analysis, it is identical to the above analysis of the *Height* algorithm because a recursive call is made for each node of an extended binary tree.

Figure 1. BSTree analysis (Levitin, 2011, p.209; p.210)

Algorithm mergeSort(n, a[0..n-1], m[0..n-1])

// Sorts array a[0..n-1] and m[0..n-1] by recursive mergesort

```
// Input: n is the size of the arrays. a[0..n-1] is an int array. m[0..n-1] is a Movie array.
// Output: a[0..n-1] and m[0..n-1] sorted in decreasing order.
// If an array only contains one component, the array is already sorted.
// In this case, the following loop stops running and merge() starts to run.
if (n>1)
        n1 <- n/2
        n2 <- n – 1

        // a is broken down into two arrays, L and R
        // m is broken down into two arrays, mL and mR
        copy a[0.. n1 – 1] to L[0..n1 + 1]
        copy a[n1..n - 1] to R[0..n2 + 1]
        copy m[0..n1 – 1] to mL[0..n1 + 1]
        copy m[n1..n - 1] to mR[0..n2 + 1]

        // Uses mergeSort to further break down the left and right arrays
        // When arrays are broken into smaller array, only containing one array,
        // merge() is used to merge smaller array into bigger ones, until the size
         // reaches the original size, resulting in sorted int and Movie arrays.
        mergeSort(n1, L[0..n1 + 1], mL[0..n1 + 1])
        mergeSort(n2, R[0..n2 + n1], mR[0..n2 + 1]
        merge(n1, L[0..n1 + 1], n2, R[0..n2 + 1], a[0..n-1], mL[0..n1 + 1],  mR[0..n2 +
1], m[0..n-1])
```

Algorithm merge(n1, L[0..n1 - 1], , n2, R[n2..n - 1], a[0..n-1], mL[0..n1 - 1], mR[n2..n - 1], m[0..n-1])

```
// Merges two sorted int arrays into one array and two sorted Movie arrays into one
                                                                        array
// Input:
         /// n1: the size for the left arrays</param>
```

```
        /// L[0..n1 + 1],: the left int array</param>
        /// n2: the size for the right arrays</param>
        /// R[0..n2 + 1]: the right int array</param>
        /// a[0..n-1]: the int array</param>
        /// mL[0..n1 + 1]: the left Movie array</param>
        /// mR[0..n2 + 1]: the right Movie array </param>
        /// m[0..n-1]: the Movie array</param>
    // Sorted arrays a[0..n-1] of the elements of L and R and m[0..n-1] of elements of mL
                                                                    and mR
    // inf is a flag, having a large negative value
    // to ensure that when sorting, the last values of the int arrays,L[n1] and R[n2]
    // are the smallest values, which is true because both the arrays are in the
descending order
    // and the last component of the array should be the smallest value, hence the large
negative value.
    inf <- -1000000001;
    L[n1] = inf;
    R[n2] = inf;
    i <- 0; j <- 0; k <-0
    while i < n1 and j < n2 do
        if L[i] > R[j]
            a[k] <- L[i]; m[k] <- mL[i]; i<-i+1
        else a[k] = R[j]; m[k] <- mR[j]; j<j+1
        k <- k+1
    if i = n1
        copy R[j..n2+1] to a[k..n-1]; mR[j..n2+1] to m[k..n -1]
    else copy L[i..n1+1] to a[k..n – 1]; copy mR[j..n2+1] to m[k..n-1]
```

Analysis:

The recurrence relation for the number of key comparisons in mergesort is:

C(n) = 2C(n/2) + C_{merge}(n) for n > 1, C(1) = 0

C_{merge}(n) is the number of key comparisons performed during the merging stage. For the worst case, C_{worse} (n) = n – 1, as described in the textbook (Levitin, 2011, p.199). For the best case, the C_{best} (n) = n/2. This happens when the smallest element of one sorted sub-list is larger than the first element of the opposing list. Only one element from the opposing list is compared, which reduces the number in each merge step to n/2 (Quiles, n.d.).

For C_{worse} (n) = n – 1 for C_{merge}(n), C_{worse} (n) = n$log_2 n$- n + 1 ϵ O(nlogn), as shown in the following workings.

The complexity of worst-case Merge Sort is:

$$T(N) = 2T(N/2) + N - 1 \quad (2.1)$$
$$T(N) = 2[2T(N/4) + N/2 - 1] + N - 1 \quad (2.2)$$
$$T(N) = 4[2T(N/8) + N/4 - 1] + 2N - 3 \quad (2.3)$$
$$T(N) = 8T(N/8) + N + N + N - 4 - 2 - 1 \quad (2.4)$$

$$T(N) = 2^k T(\frac{N}{2^k}) + kN - (2^k - 1) \quad (2.5)$$

In Eq. 2.5, a new variable called "k" is introduced. This variable represents the depth of the recursion. When the sort is recursively dividing the input list, the recursion stops when the list contains a single element. *A single element list is already sorted.*

$$T(1) = 0 \quad (2.6)$$
$$2^k = N \quad (2.7)$$
$$k = \log_2 N \quad (2.8)$$
$$T(N) = N \log_2 N - N + 1 \quad (2.9)$$

Figure 2. Mathematical analysis for the worse case (Quiles, n.d.)

For C_{best} (n) = n/2 for C_{merge}(n), C_{best} (n) = $\frac{n}{2} log_2 n$ ϵ O(nlogn), as shown in the following workings.

$$T(N) = 2T(N/2) + N/2 \quad (2.10)$$
$$T(N) = 2[2T(N/4) + N/4] + N/2 \quad (2.11)$$
$$T(N) = 4[2T(N/8) + N/8] + N \quad (2.12)$$
$$T(N) = 8T(N/8) + 3N/2 \quad (2.13)$$
$$T(N) = 2^k T(\frac{N}{2^k}) + \frac{kN}{2} \quad (2.14)$$
$$T(N) = \frac{N}{2} \log_2 N \quad (2.15)$$

Eq. 2.14 is produced from 2.10 using the same substitution process (2.11-2.13) as before. Like Eq. 2.9 from the worst case, Eq. 2.15 shows that Merge Sort has *O(Nlog(N))* complexity with the best possible input.

Figure 3. Mathematical analysis for the best case (Quiles, n.d.)

The average case is bounded from above by the worse case function C_{worse} (n) = n$log_2 n$ - n + 1 ,and from below by the best-case function C_{best} (n) = $\frac{n}{2} log_2 n$, so the average case

C_{avg} (n) ϵ O(nlogn).

Reference:

ALGORITHM *Mergesort*$(A[0..n-1])$

//Sorts array $A[0..n-1]$ by recursive mergesort
//Input: An array $A[0..n-1]$ of orderable elements
//Output: Array $A[0..n-1]$ sorted in nondecreasing order
if $n > 1$
    copy $A[0..\lfloor n/2 \rfloor - 1]$ to $B[0..\lfloor n/2 \rfloor - 1]$
    copy $A[\lfloor n/2 \rfloor..n-1]$ to $C[0..\lceil n/2 \rceil - 1]$
    Mergesort$(B[0..\lfloor n/2 \rfloor - 1])$
    Mergesort$(C[0..\lceil n/2 \rceil - 1])$
    Merge(B, C, A) //see below

The ***merging*** of two sorted arrays can be done as follows. Two pointers (array indices) are initialized to point to the first elements of the arrays being merged. The elements pointed to are compared, and the smaller of them is added to a new array being constructed; after that, the index of the smaller element is incremented to point to its immediate successor in the array it was copied from. This operation is repeated until one of the two given arrays is exhausted, and then the remaining elements of the other array are copied to the end of the new array.

ALGORITHM *Merge*$(B[0..p-1], C[0..q-1], A[0..p+q-1])$

//Merges two sorted arrays into one sorted array
//Input: Arrays $B[0..p-1]$ and $C[0..q-1]$ both sorted
//Output: Sorted array $A[0..p+q-1]$ of the elements of B and C
$i \leftarrow 0;\ j \leftarrow 0;\ k \leftarrow 0$
while $i < p$ **and** $j < q$ **do**
    if $B[i] \le C[j]$
        $A[k] \leftarrow B[i];\ i \leftarrow i+1$
    else $A[k] \leftarrow C[j];\ j \leftarrow j+1$
    $k \leftarrow k+1$
if $i = p$
    copy $C[j..q-1]$ to $A[k..p+q-1]$
else copy $B[i..p-1]$ to $A[k..p+q-1]$

The operation of the algorithm on the list 8, 3, 2, 9, 7, 1, 5, 4 is illustrated in Figure 5.2.

How efficient is mergesort? Assuming for simplicity that n is a power of 2, the recurrence relation for the number of key comparisons $C(n)$ is

$$C(n) = 2C(n/2) + C_{merge}(n) \quad \text{for } n > 1, \quad C(1) = 0.$$

Let us analyze $C_{merge}(n)$, the number of key comparisons performed during the merging stage. At each step, exactly one comparison is made, after which the total number of elements in the two arrays still needing to be processed is reduced by 1. In the worst case, neither of the two arrays becomes empty before the other one contains just one element (e.g., smaller elements may come from the alternating arrays). Therefore, for the worst case, $C_{merge}(n) = n - 1$, and we have the recurrence

$$C_{worst}(n) = 2C_{worst}(n/2) + n - 1 \quad \text{for } n > 1, \quad C_{worst}(1) = 0.$$

Hence, according to the Master Theorem, $C_{worst}(n) \in \Theta(n \log n)$ (why?). In fact, it is easy to find the exact solution to the worst-case recurrence for $n = 2^k$:

$$C_{worst}(n) = n \log_2 n - n + 1.$$

Figure 4. Analysis (Levitin, 2011, p.)

DisplayMoviesInOrder()

// Displays the first ten movies within the sorted movie array, using the sorted movie array.

// The function accounts for the situation that there are not less than ten movies within the array.

// This array will not display a movie's borrowed times if its borrowed times is zero.

```
i ← 0

    Foreach Movie m in movieArray

        If i<= 10

            If m != null

                If m.BorrowedTimes() != 0 // getter function getting the
                                                borrowed times of the movie m

                m.GetInfo() // a void function writing all info of movie m to the
                                                                    console

            ++i

        else
```

return

Analysis:

The basic operations in this function are the comparisons m != null and i <= 10. m != null is performed once every foreach loop, until the condition i <= 10 is no longer true. This algorithm does not depend on the any input size as it only depends on the comparison

i <= 10

C(n) = $\sum_{i=0}^{10} 1$ = 10 – 0 + 1 = 11

C(n) = 11 ϵ O(1)

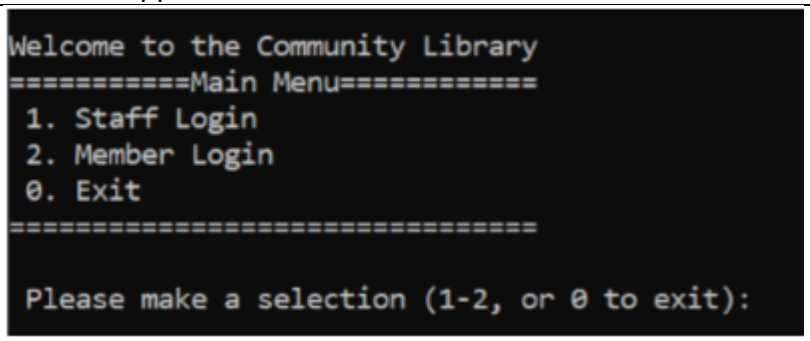
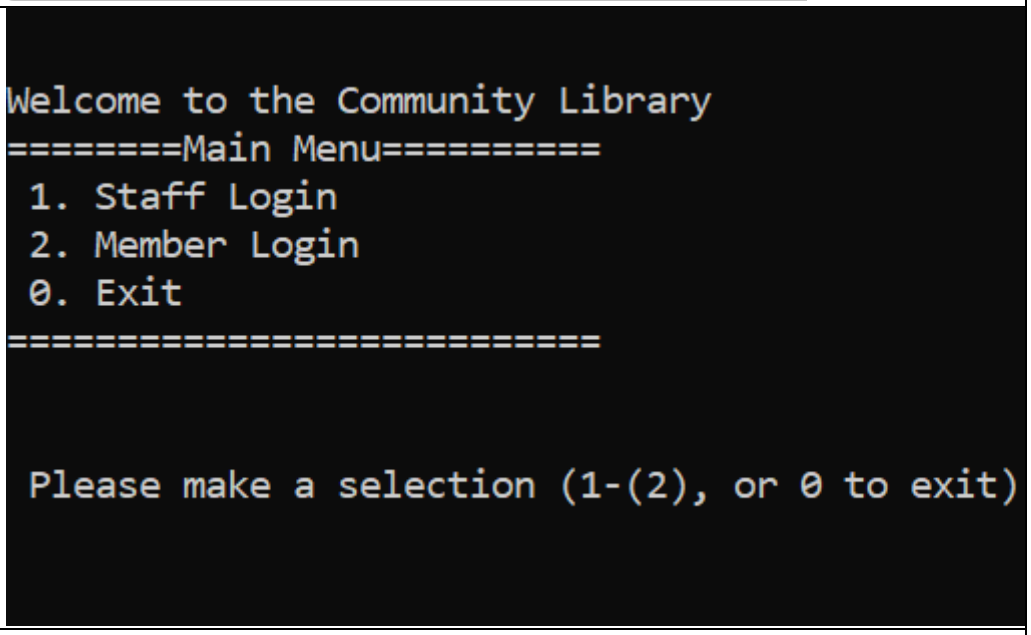
Test report

Test summary

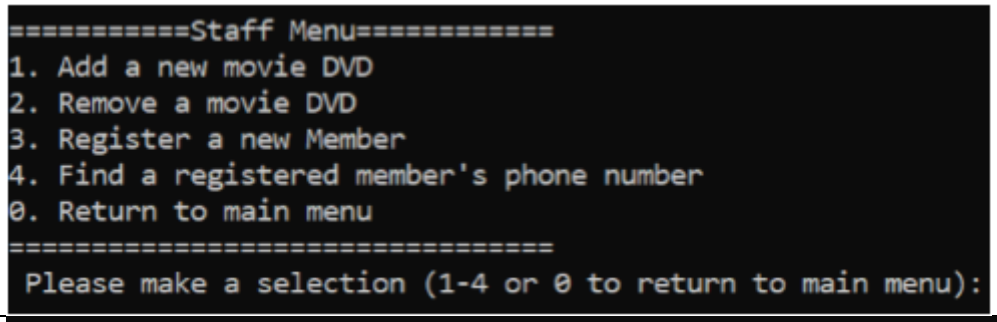
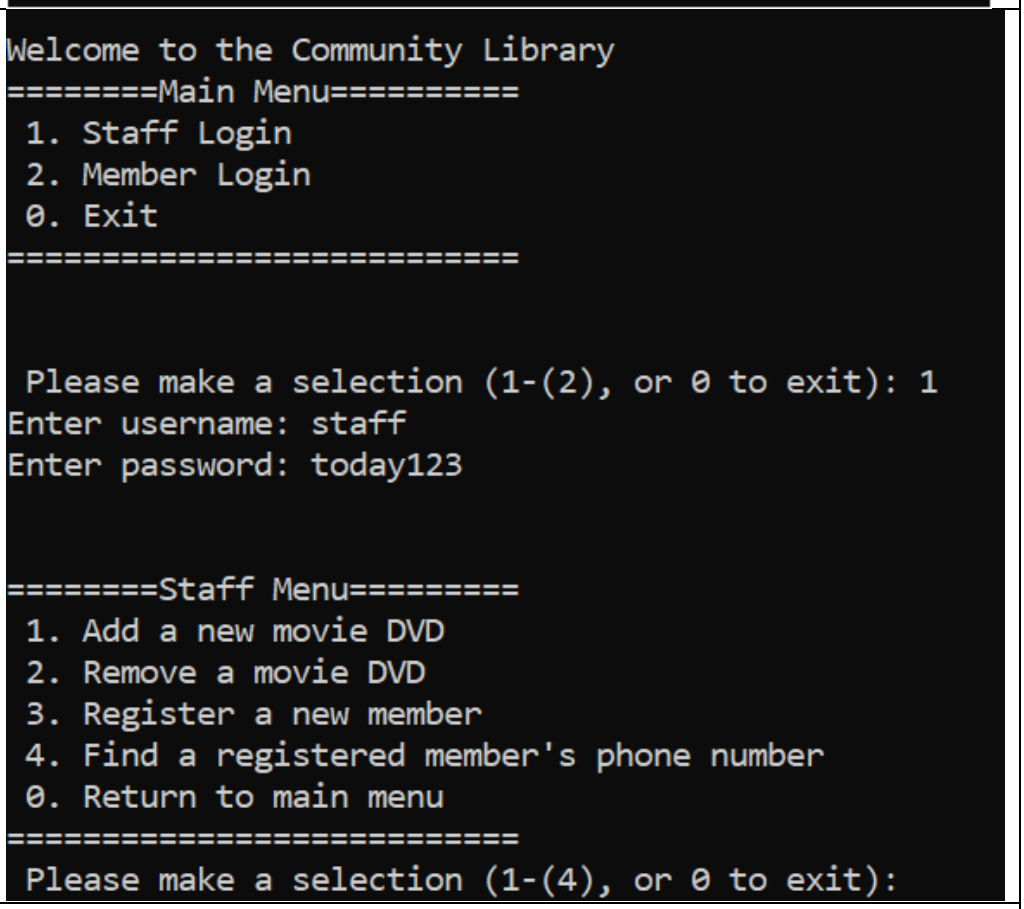
Staff's username and password are hardcoded as "staff" and "today123".

There are three custom testing sets of members. The testing sets are included in the Appendices.

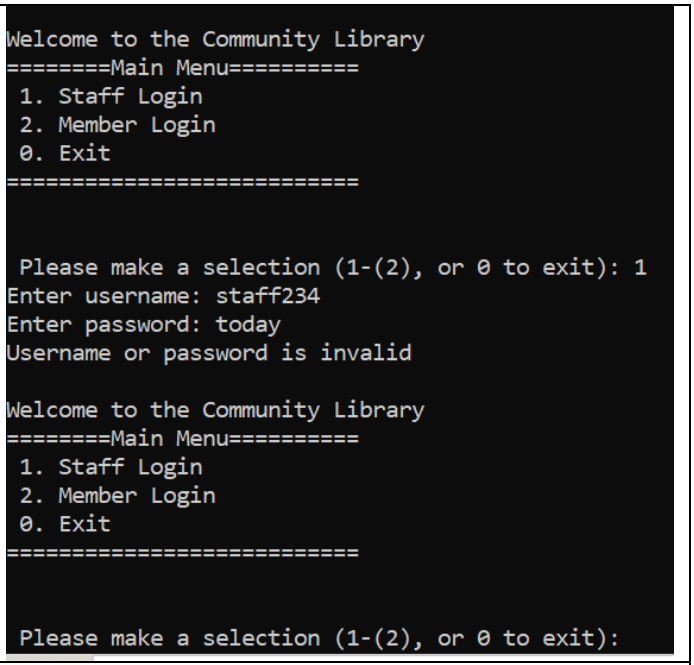
Main menu functionalities

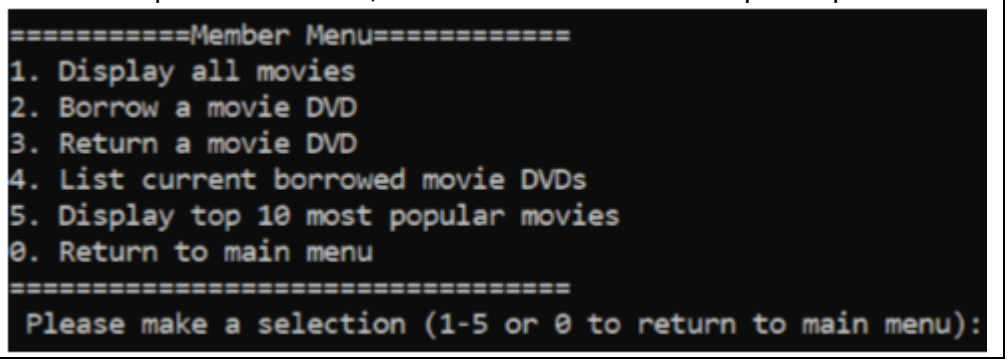
Category	Welcome screen
Input	Start the application
Expected Output	
Actual Output	
Comment	The actual output and expected output matches.

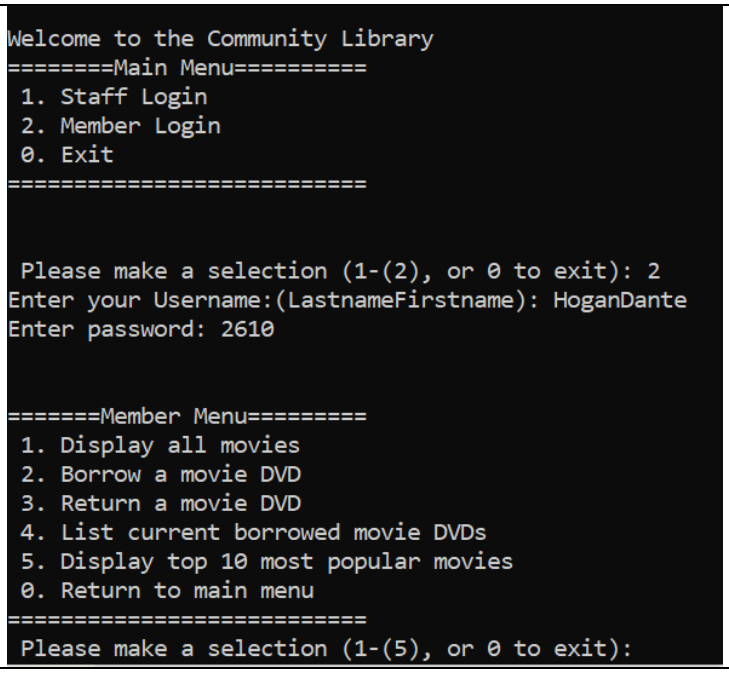
Category	Log into staff menu (Successful)
Input	Option: 1, username: staff, password: today123;
Expected Output	Once the inputs are entered, the staff menu shows up as expected.

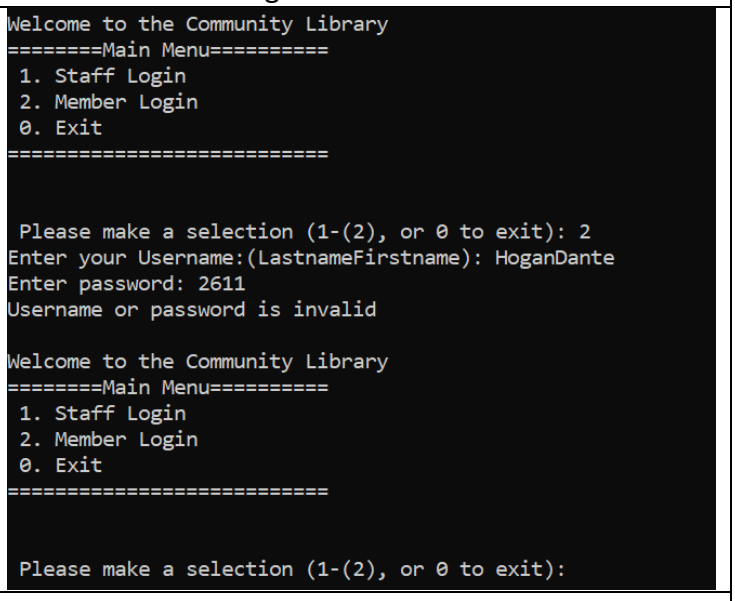
	
Actual Output	
Comment	The actual output and expected output matches.

Category	Log into staff menu (Unsuccessful)
Input	Option: 1; username: staff234; password: today;
Expected Output	A warning message shows up and tells the user that the username and password combination is not correct. The screen goes back to the welcome menu.

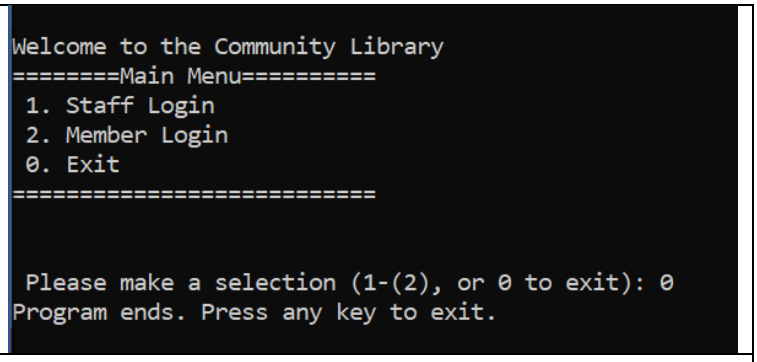
Actual Output	
Comment	The actual output and expected output matches.

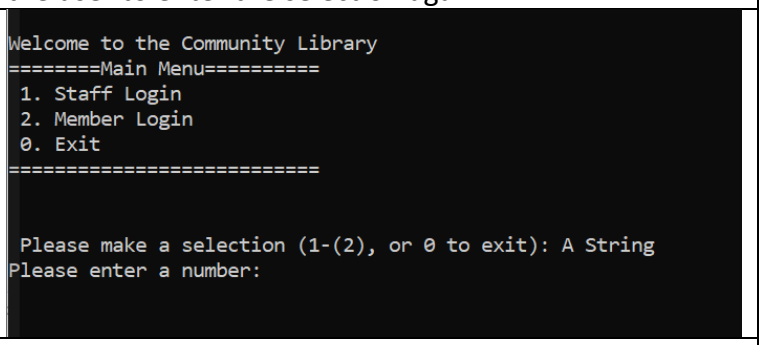
Category	Log into member menu (Successful)
Input	Option: 2; username: HoganDante; password: 2610; (Taken from testing set 1)
Expected Output	Once the inputs are entered, the member menu shows up as expected. 

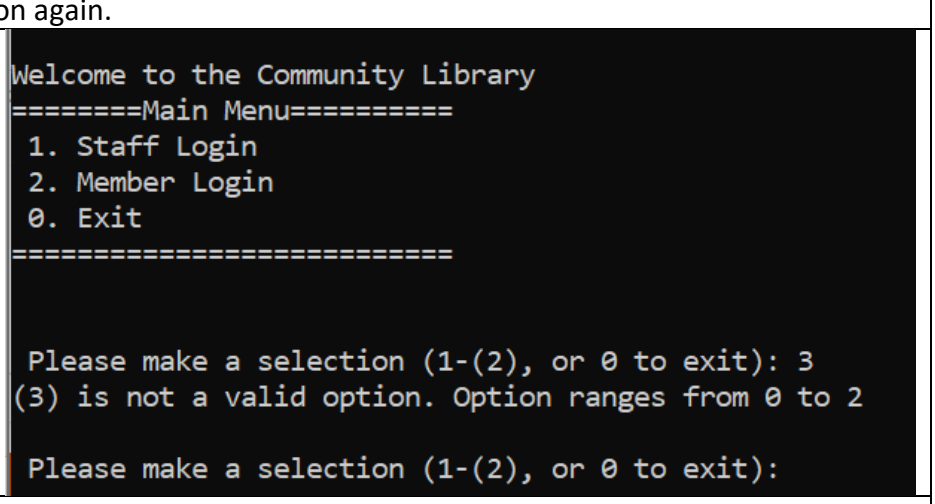
Actual Output	
Comment	The actual output and expected output matches.

Category	Log into member menu (Unsuccessful)
Input	Option: 2; username: HoganDante; password: 2611; (Taken from testing set 1)
Expected Output	A warning message shows up and tells the user that the username and password combination is not correct. The screen goes back to the welcome menu.
Actual Output	
Comment	The actual output and expected output matches.

Category	Exit
Input	Option: 0;
Expected Output	The program ends.

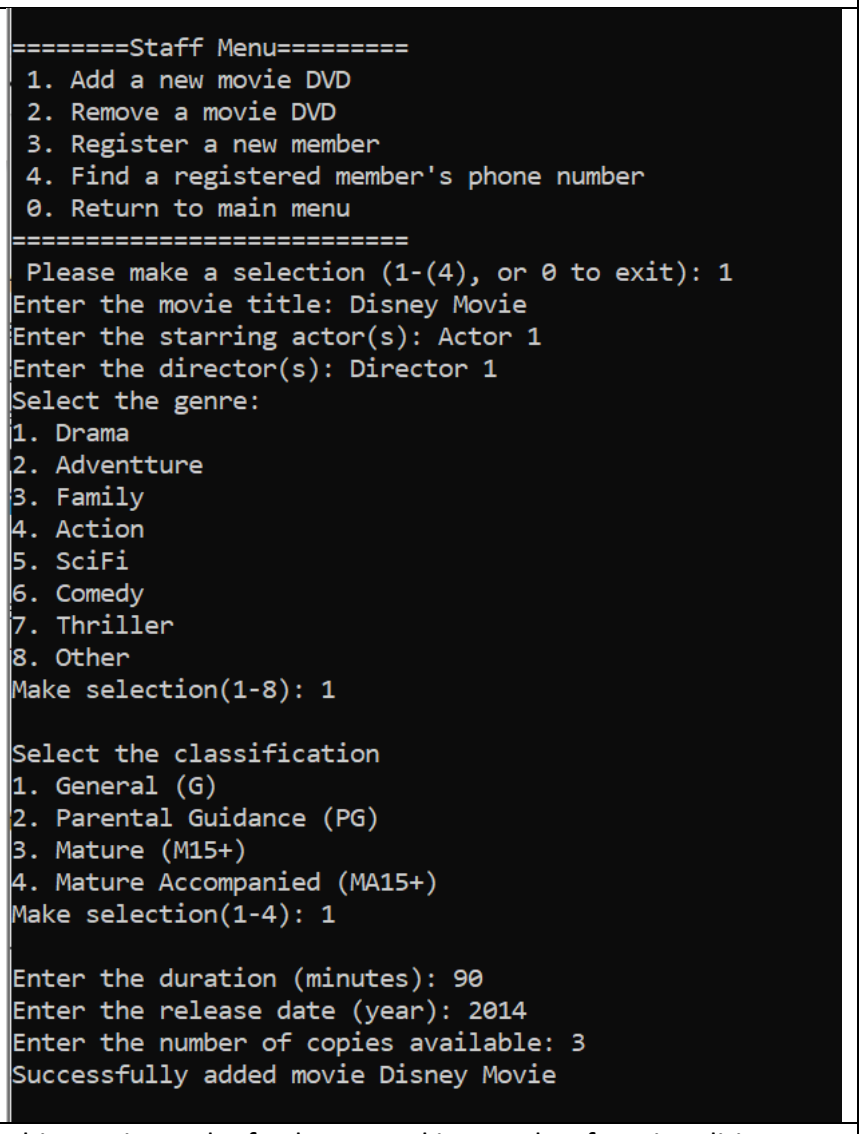
Actual Output	
Comment	The actual output and expected output matches.

Category	The input for selection must be a number
Input	Option: A String;
Expected Output	A warning message shows up and tells the user that the password needs to be a number. The program prompts the user to enter the selection again.
Actual Output	
Comment	The actual output and expected output matches.

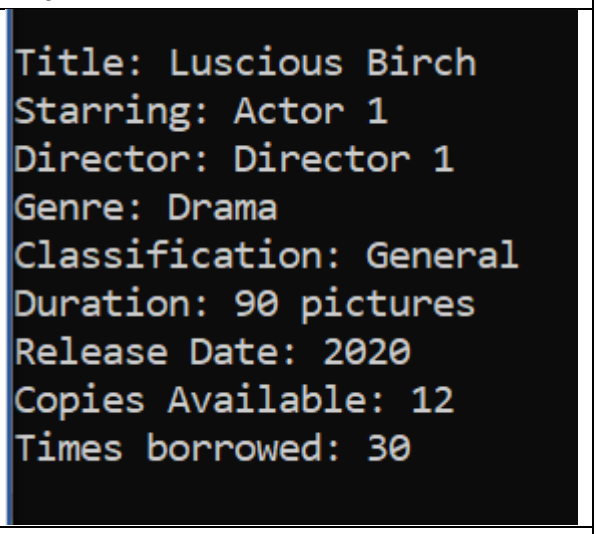
Category	The input for selection must be appropriate
Input	Option: 6
Expected Output	A warning message shows up and tells the user that the password needs to be within the appropriate range. The program prompts the user to enter the selection again.
Actual Output	
Comment	The actual output and expected output matches.

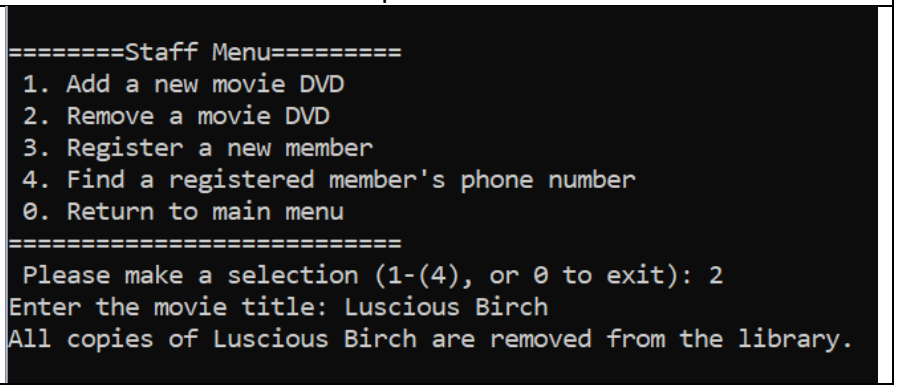
Staff menu functionalities

Setup from the Welcome menu: Option: 1; username: staff; password: today123;

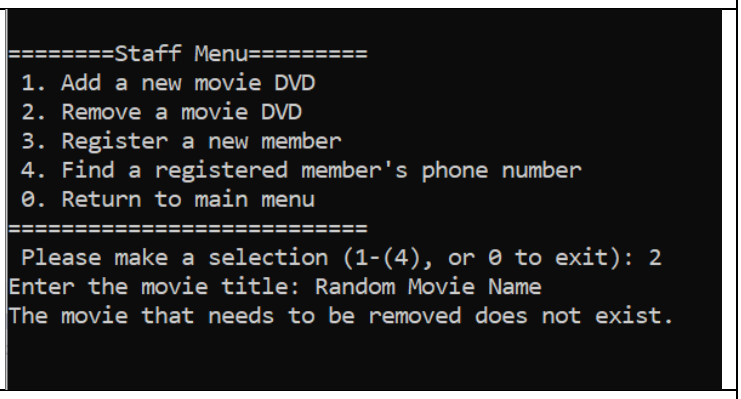
Category	Add a new movie DVD (This movie does not exist in the movie library.)
Input	Use testing set 1 for a list of pre-added movies. This testing set does not have the movie “Disney Movie” Option: 1; Movie title: Disney Movie; Actor: Actor 1; Director: Director 1; Genre: 1 (Drama); Classification: 1(General); Duration: 90; Year: 2014; Copies: 3;
Expected Output	The movie is added because it is not added before.
Actual Output	
Comment	This movie can be further tested in member functionalities test regarding further functionalities.

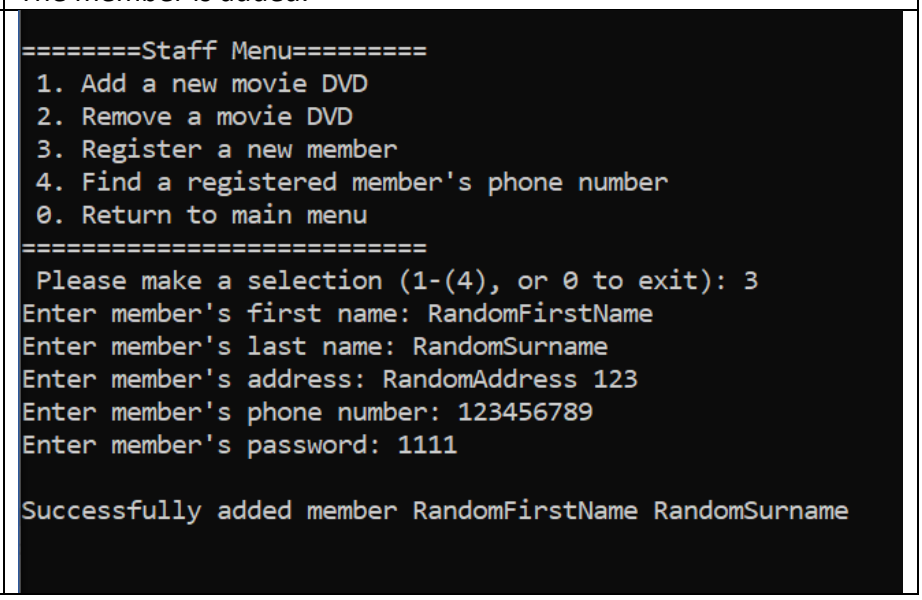
Category	Add a new movie DVD (This movie already exists in the movie library.)

Input	Use testing set 1 for a list of pre-added movies. This testing set already has the movie “Luscious Birch” with 28 copies. Option: 1; Movie title: Luscious Birch; Option: 2
Expected Output	Luscious Birch have 30 (28 + 2 recently added copies). We log in the member menu and use the display all movies function to check the number of copies “Luscious Birch”.
Actual Output	
Comment	The actual output and expected output matches.

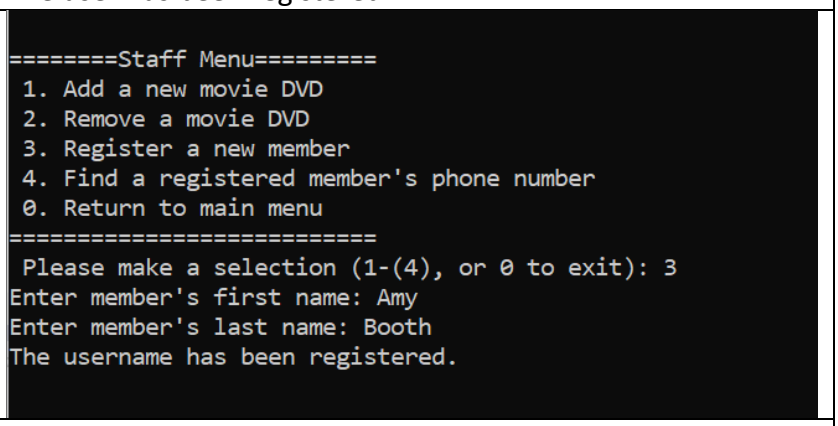
Category	Remove a movie DVD (Successful)
Input	Use testing set 1 for a list of pre-added movies. This testing set already has the movie “Luscious Birch” Option: 2; Movie title: Luscious Birch;
Expected Output	All copies of “Luscious Birch” are removed from the library. When a member uses the display all movies from the library or the top 10, the movies no longer show up. However, if a member has it, “Luscious Birch” still shows up in their list of borrowed movies.
Actual Output	

Comment	The actual output and expected output matches. This movie will be further tested in member functionalities test regarding further functionalities.

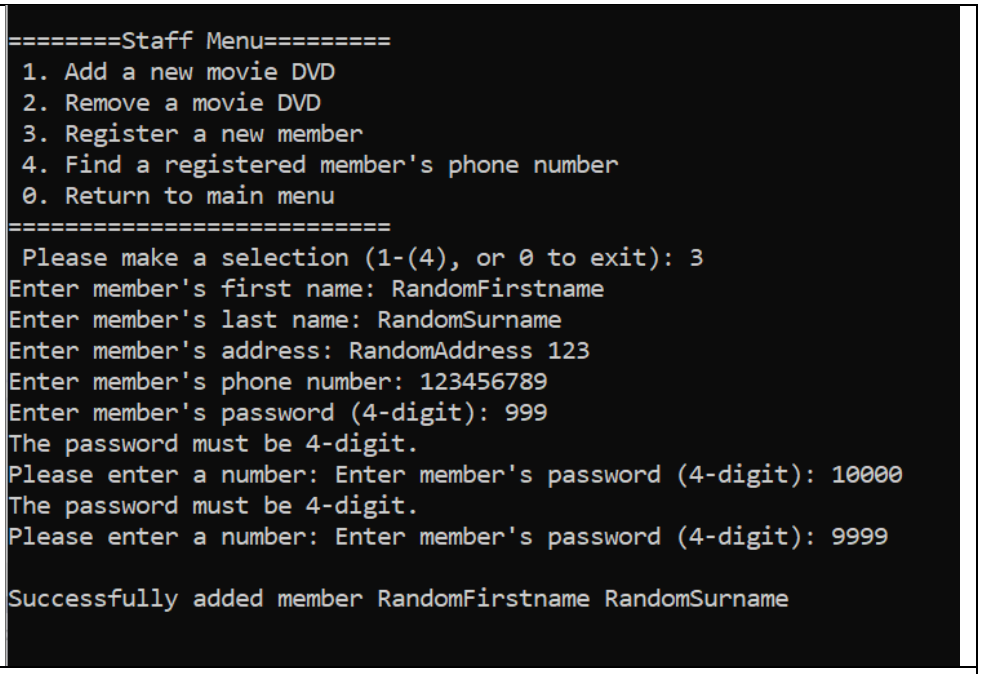
Category	Remove a movie DVD (Unsuccessful)
Input	Use testing set 1 for a list of pre-added movies. This testing set does not have the movie “Random Movie Name” Option: 2; Movie title: Random Movie Name;
Expected Output	The program gives a warning that the movie does not exist.
Actual Output	
Comment	The actual output and expected output matches.

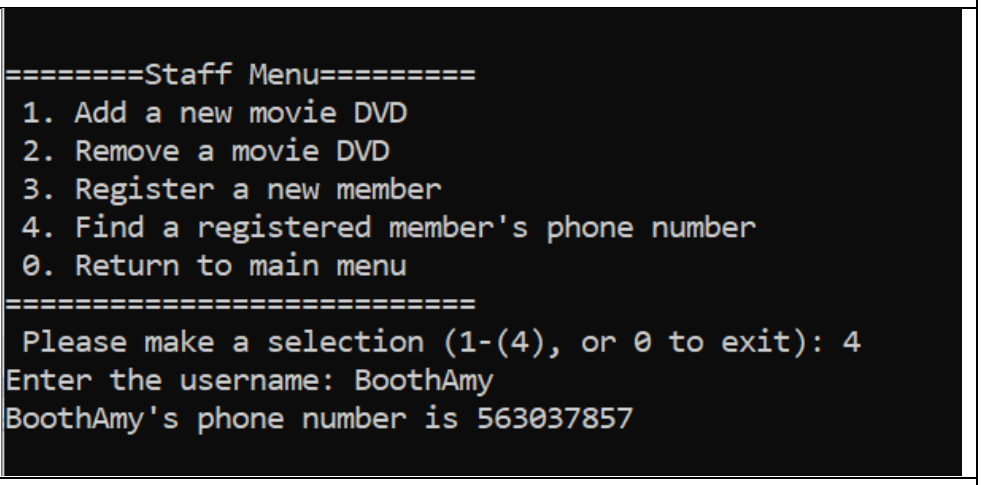
Category	Register a new member (Successful)
Input	Use testing set 1 for a list of pre-added members. This testing set does not have the member "RandomFirstname" ”RandomSurname” Option: 3; First name: RandomFirstname; Last name: RandomSurname; Address: RandomAddress 123; Phone number: 123456789; password: 1111;
Expected Output	The member is added.
Actual Output	

Comment	The actual output and expected output matches. This member will be further tested in test member functionalities test regarding further functionalities.

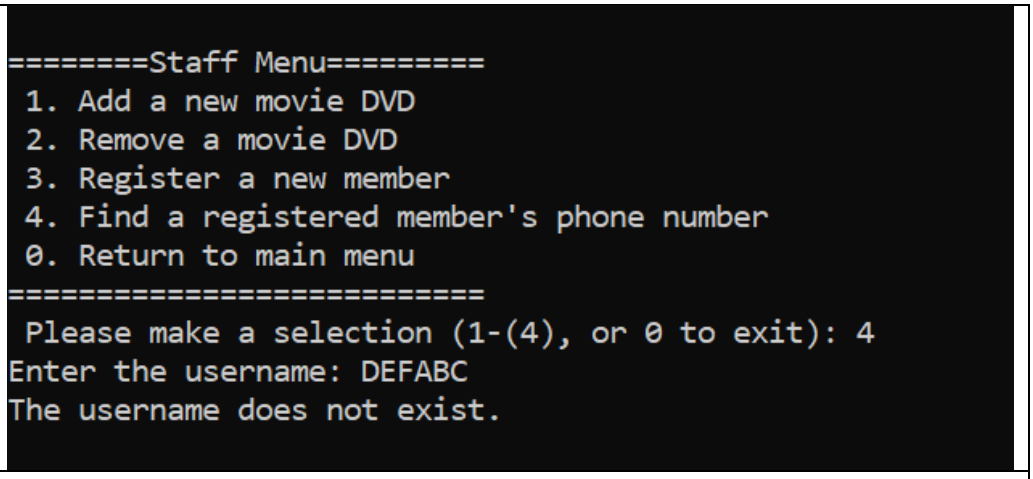
Category	Register a new member (Unsuccessful)
Input	Use testing set 1 for a list of pre-added members. This testing set has the member "Amy" ”Booth” Option: 3; First name: Amy; Last name: Booth;
Expected Output	The user has been registered.
Actual Output	
Comment	The actual output and expected output matches.

Category	Register a new member (Unsuccessful)
Input	Use testing set 1 for a list of pre-added members. This testing set does not have the member "RandomFirstname" ”RandomSurname” Option: 3; First name: RandomFirstname; Last name: RandomSurname; Address: RandomAddress 123; Phone number: 123456789; password: 999; password:10000;
Expected Output	The program prompts the user to re-enter the password (4-digit). password: 9999 is used

Actual Output	
Comment	The actual output and expected output matches.

Category	Find a registered member's phone number (Successful)
Input	Use testing set 1 for a list of pre-added members. The testing set has member "Amy" "Booth". Her phone number is 563037857. Option: 4; Username: BoothAmy;
Expected Output	The expected phone number is 563037857.
Actual Output	
Comment	The actual output and expected output matches.

Category	Find a registered member's phone number (Unsuccessful)
Input	Use testing set 1 for a list of pre-added members. The testing set does not have member "ABC" "DEF". Her phone number is 563037857. Option: 4; Username: DEFABC;
Expected Output	The username does not exist, therefore there's no corresponding phone number.

Actual Output	
Comment	The actual output and expected output matches.

Member menu functionalities

New member can be registered or member from the provided testing set can be used. The following member is from Testing set 1. Some categories might be tested additionally with Testing set 2 and Testing set 3. Regarding the "Display top 10 most popular movies" tests, the Member class's "borrowedTimes" attributes is initialized with a value for the purpose of testing easier and faster. When a movie is added manually, the "borrowedTimes" attribute of that movie is zero.

Testing set 1

Setup from the Welcome menu: Option: 2; username: MorenoAdam; password: 7425; (or any registered user)

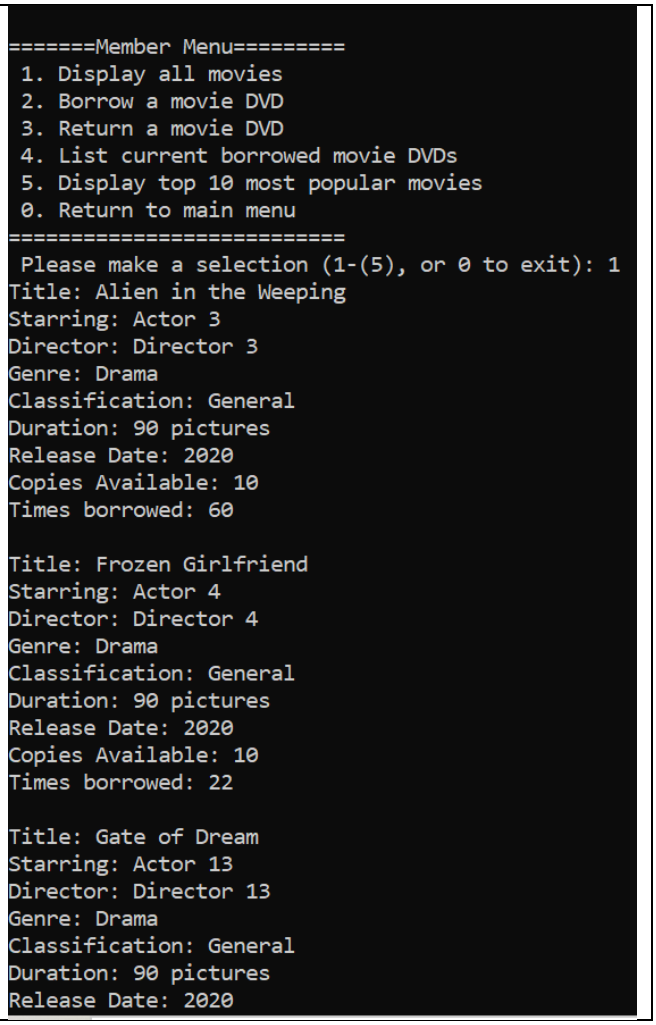
Testing set 2

Setup from the Welcome menu: Option: 2; username: BeckTahmina; password: 1452; (or any registered user)

Testing set 3

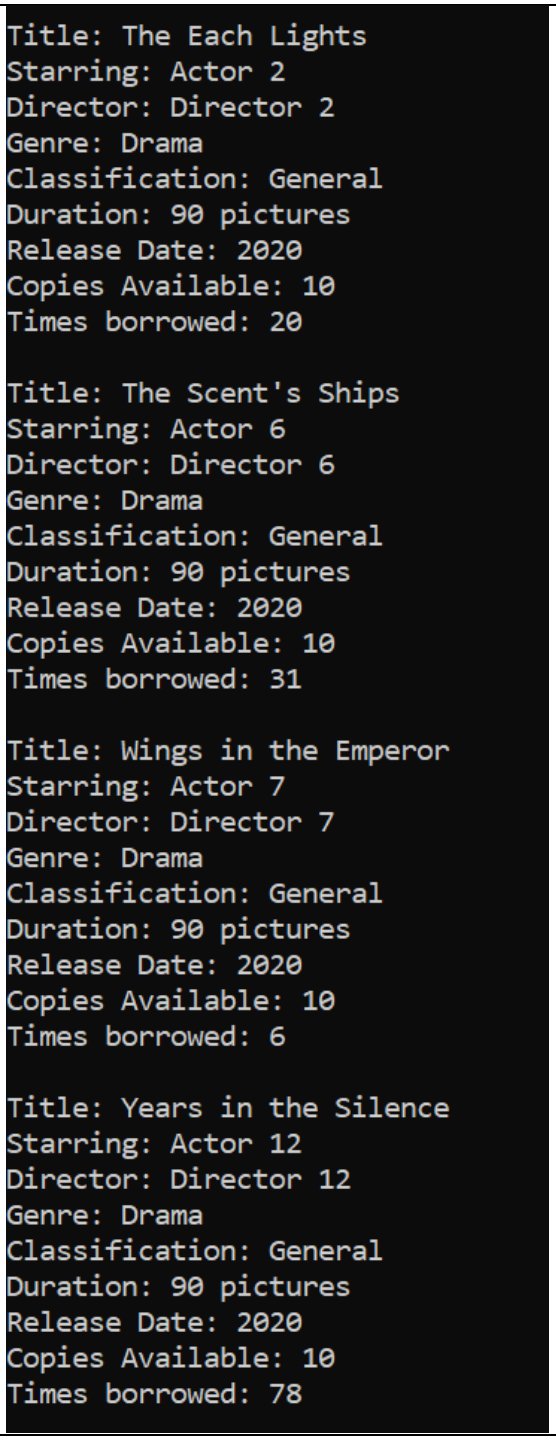
Setup from the Welcome menu: Option: 2; username: SweetKaya; password: 4122; (or any registered user)

Category	Display all movies (Set 1)
Input	Option 1;
Expected Output	The data from all movies in Testing set 1 are accurate and displayed in alphabetical order

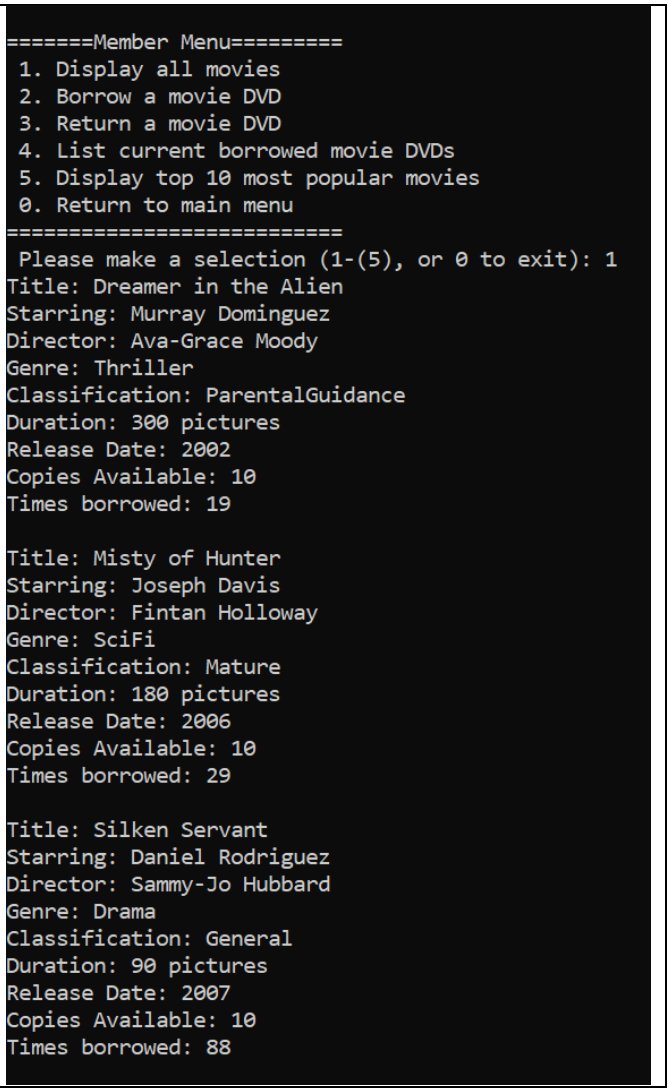
Actual Output	

```
Title: Gate of Dream
Starring: Actor 13
Director: Director 13
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 82

Title: Luscious Birch
Starring: Actor 1
Director: Director 1
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 30

Title: Nobody of Door
Starring: Actor 10
Director: Director 10
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 26

Title: Rose of Snow
Starring: Actor 5
Director: Director 5
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 39
```

	Title: Silk in the Emperor Starring: Actor 8 Director: Director 8 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 2 Title: Sleeping Lord Starring: Actor 9 Director: Director 9 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 99 Title: Tears in the Fire Starring: Actor 11 Director: Director 11 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 40 Title: The Each Lights Starring: Actor 2 Director: Director 2 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 20

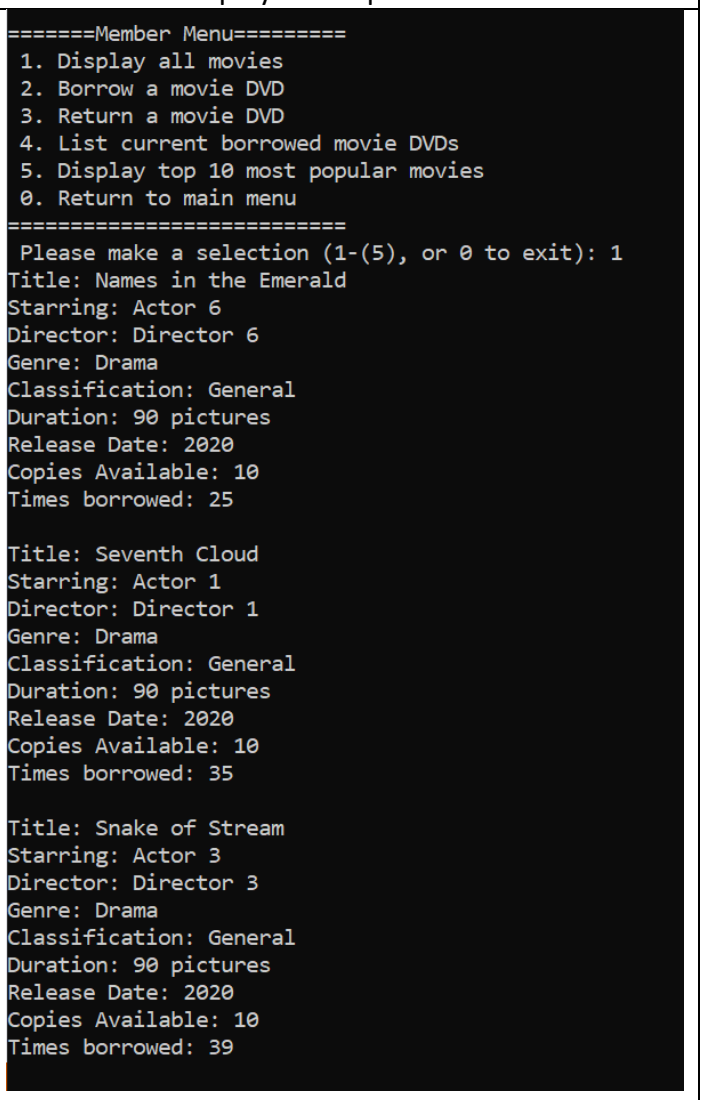
	
Comment	The actual output and expected output matches.

Category	Display all movies (Set 2)
Input	Option 1;
Expected Output	The data from all movies in Testing set 2 are accurate and displayed in alphabetical order

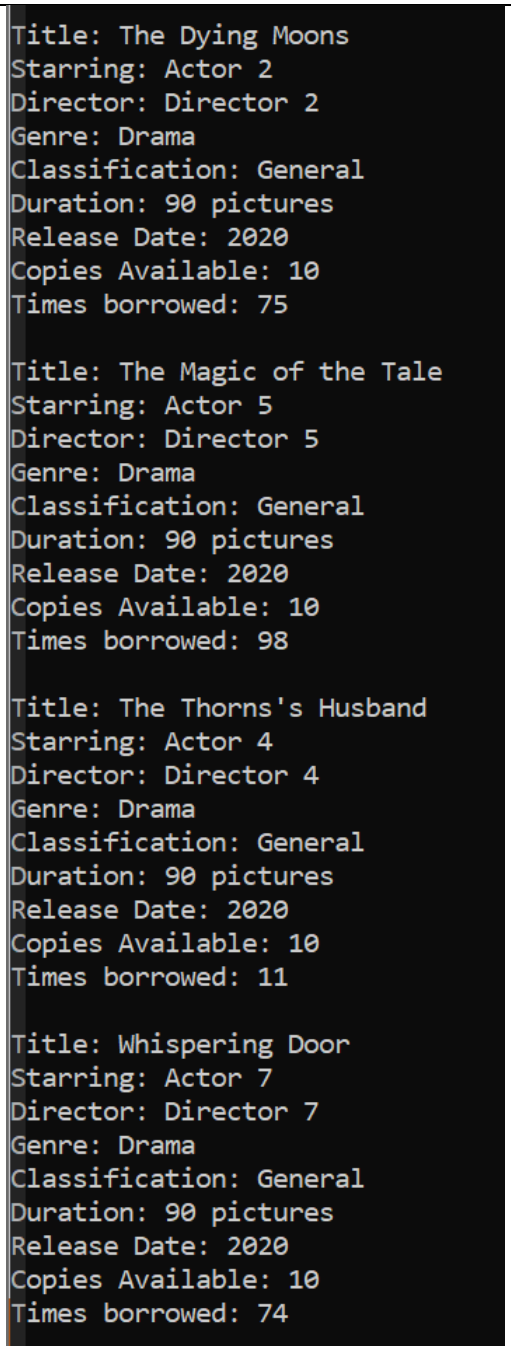
Actual Output	

	Title: Spirits of Sky Starring: Laurence Boone Director: Asmaa Mckay Genre: Action Classification: ParentalGuidance Duration: 180 pictures Release Date: 2009 Copies Available: 10 Times borrowed: 16 Title: Stolen Nothing Starring: Kaleb Williamson Director: Aiden Mclean Genre: Comedy Classification: General Duration: 90 pictures Release Date: 2008 Copies Available: 10 Times borrowed: 14 Title: The Blue Lover Starring: Leoni Whyte Director: Missy Hawes Genre: Family Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 50 Title: The Dream's Wife Starring: Joshua Smith Director: Darcy Riddle Genre: Comedy Classification: MatureAccompanied Duration: 90 pictures Release Date: 2003 Copies Available: 10 Times borrowed: 92

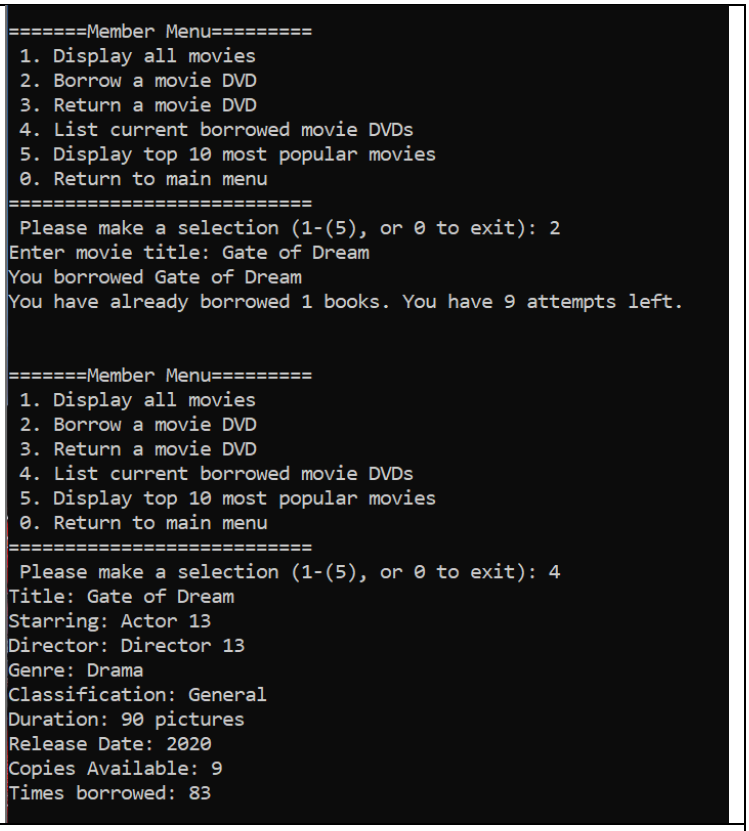
	Title: The Servants of the Lights Starring: Ajay Molloy Director: Taslima May Genre: Drama Classification: MatureAccompanied Duration: 90 pictures Release Date: 2001 Copies Available: 10 Times borrowed: 90 Title: The Splintered Gift Starring: Harry Miller Director: Tahmina Beck Genre: Family Classification: ParentalGuidance Duration: 120 pictures Release Date: 2010 Copies Available: 10 Times borrowed: 40 Title: The Thief's Visions Starring: Stephan Mcghee Director: Alayna Welch Genre: SciFi Classification: Mature Duration: 270 pictures Release Date: 2011 Copies Available: 10 Times borrowed: 51 Title: The Valley of the Ashes Starring: Jack Garcia Director: Jamie-Leigh Genre: Thriller Classification: MatureAccompanied Duration: 270 pictures Release Date: 2012 Copies Available: 10 Times borrowed: 63 Title: Wings in the Emperor Starring: Hamza Cardenas Director: Ruqayyah Soto Genre: Comedy Classification: ParentalGuidance Duration: 300 pictures Release Date: 2015 Copies Available: 10 Times borrowed: 70 Title: Wings in the Waves Starring: Cora Benton Director: Genevieve Patton Genre: Other Classification: Mature Duration: 240 pictures Release Date: 2013 Copies Available: 10 Times borrowed: 5
Comment	The actual output and expected output matches.

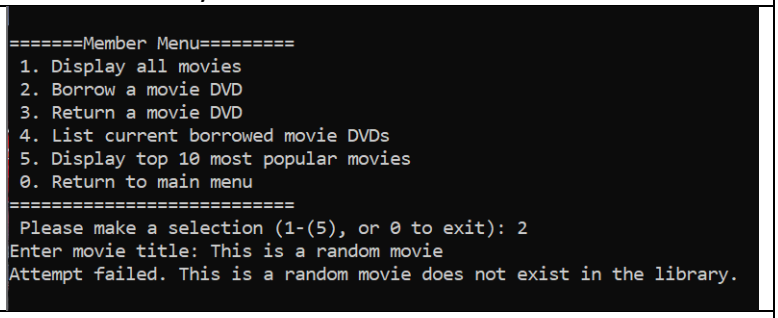
Category	Display all movies (Set 3)
Input	Option 1;
Expected Output	The data from all movies in Testing set 3 are accurate and displayed in alphabetical order
Actual Output	

```
=======Member Menu=========
 1. Display all movies
 2. Borrow a movie DVD
 3. Return a movie DVD
 4. List current borrowed movie DVDs
 5. Display top 10 most popular movies
 0. Return to main menu
==========================
 Please make a selection (1-(5), or 0 to exit): 1
Title: Names in the Emerald
Starring: Actor 6
Director: Director 6
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 25

Title: Seventh Cloud
Starring: Actor 1
Director: Director 1
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 35

Title: Snake of Stream
Starring: Actor 3
Director: Director 3
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 39
```

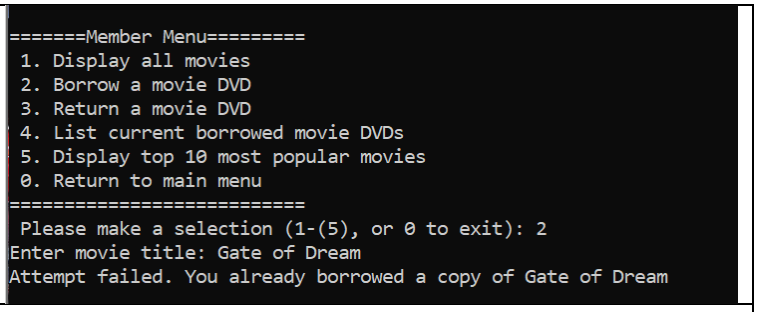
	
Comment	The actual output and expected output matches.

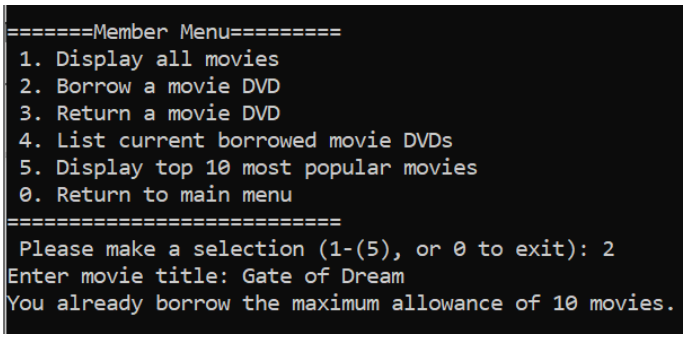
Category	Borrow a movie DVD (Successful)
Input	Option 2; Movie Title: “Gate of Dream” (or any movie from the testing set)
Expected Output	The movie is successfully borrowed and shows up on the member’s borrowed list.

Actual Output	
Comment	The actual output and expected output matches.

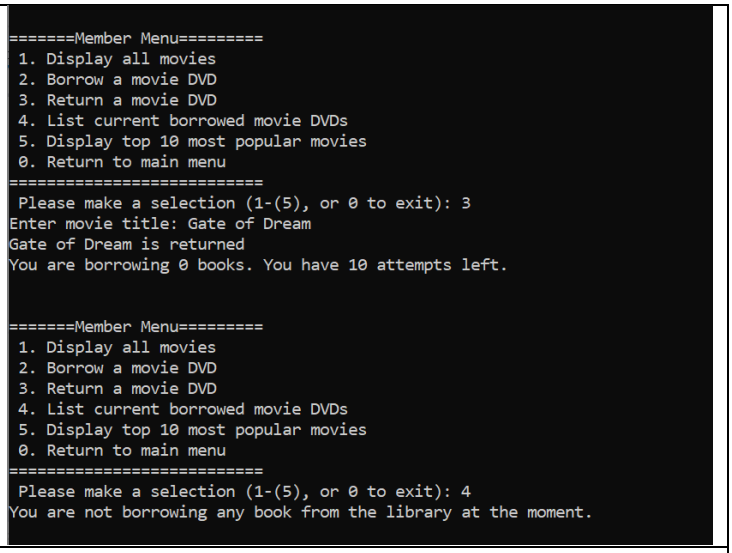
Category	Borrow a movie DVD (Unsuccessful)
Input	Option 2; Movie Title: “This is a random movie” (or any movie not from the testing set)
Expected Output	The program gives a warning that this borrow is unsuccessful because “This is a random movie” is not from the library.
Actual Output	
Comment	The actual output and expected output matches.

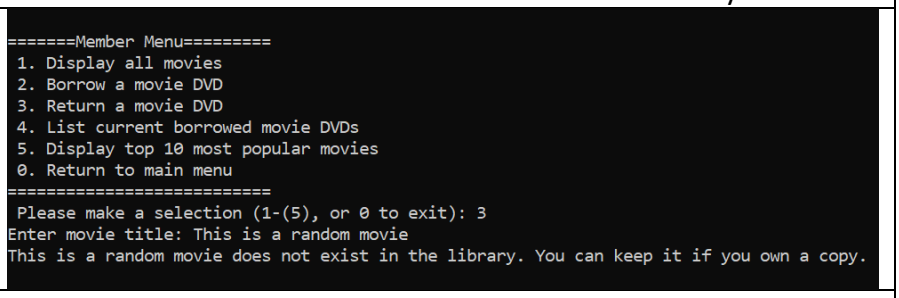
Category	Borrow a movie DVD (Unsuccessful)
Input	Pre-condition: Borrow: “Gate of Dream” (or any movie from the testing set) Option 2; Movie Title: ““Gate of Dream”
Expected Output	The program gives a warning that this borrow is unsuccessful because the member already owns a copy of ““Gate of Dream”.

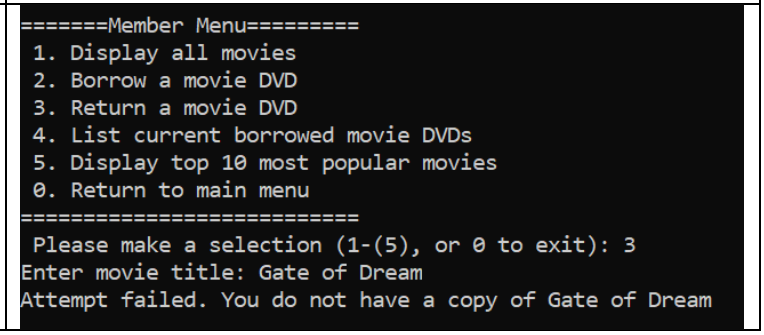
Actual Output	
Comment	The actual output and expected output matches.

Category	Borrow a movie DVD (Unsuccessful)
Input	Pre-condition: Borrow ten unique movies from the testing set except “Gate of Dream” Option 2; Movie Title: ““Gate of Dream”
Expected Output	The program gives a warning that this borrow is unsuccessful because the member already borrows 10 unique movies.
Actual Output	
Comment	The actual output and expected output matches.

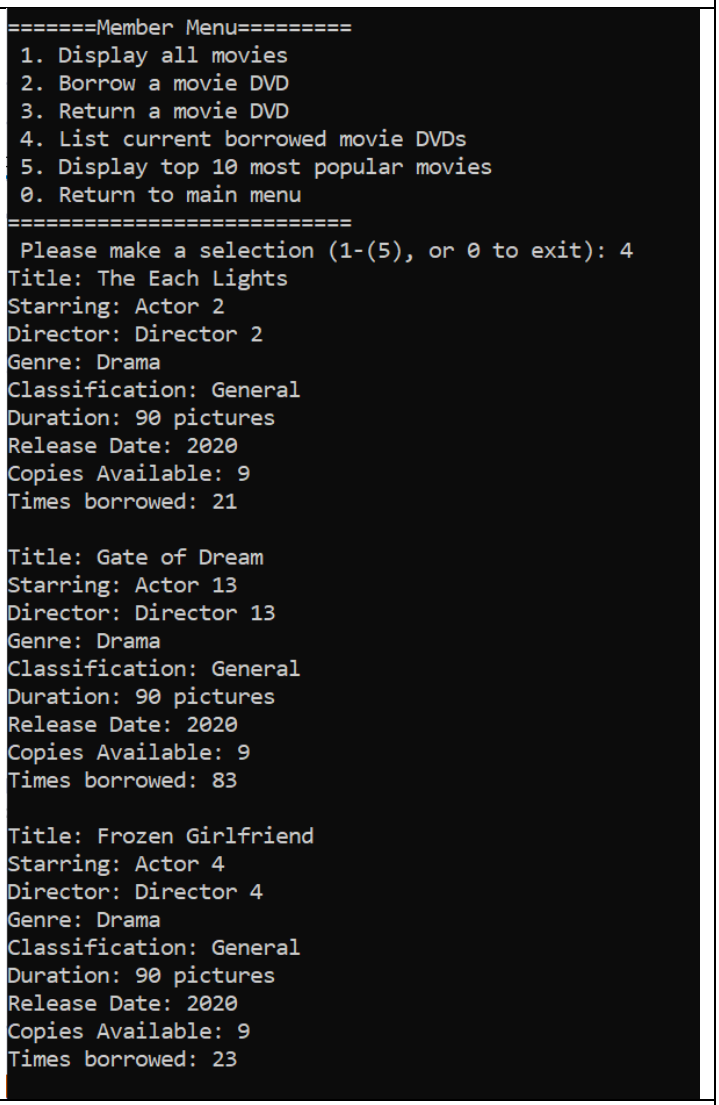
Category	Return a movie DVD (Successful)
Input	Pre-condition: Borrow: “Gate of Dream” (or any movie from the testing set) Option: 3; Movie: “Gate of Dream”
Expected Output	The movie is returned successfully. The movie is no longer in the member’s current borrowed list.

Actual Output	
Comment	The actual output and expected output matches.

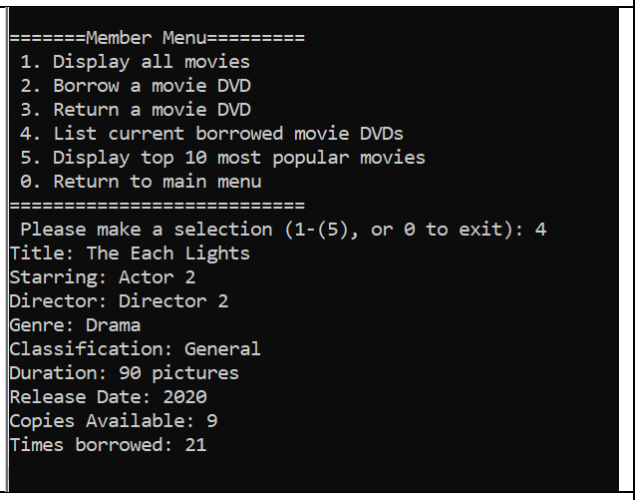
Category	Return a movie DVD (Unsuccessful)
Input	Option 3; Movie Title: “This is a random movie” (or any movie not from the testing set)
Expected Output	The program gives a warning that this return is unsuccessful because “This is a random movie” is not from the library.
Actual Output	
Comment	

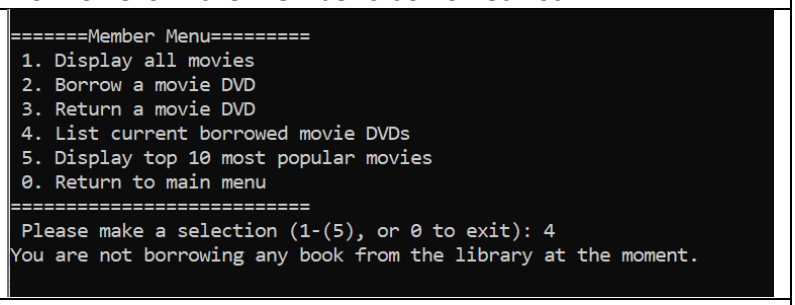
Category	Return a movie DVD (Unsuccessful)
Input	Pre-condition: The member do not already have “Gate of Dream” (or any movie from the testing set) Option 3; Movie Title: ““Gate of Dream”
Expected Output	The program gives a warning that this borrow is unsuccessful because the member does not own a copy of ““Gate of Dream”.
Actual Output	

Comment	The actual output and expected output matches.

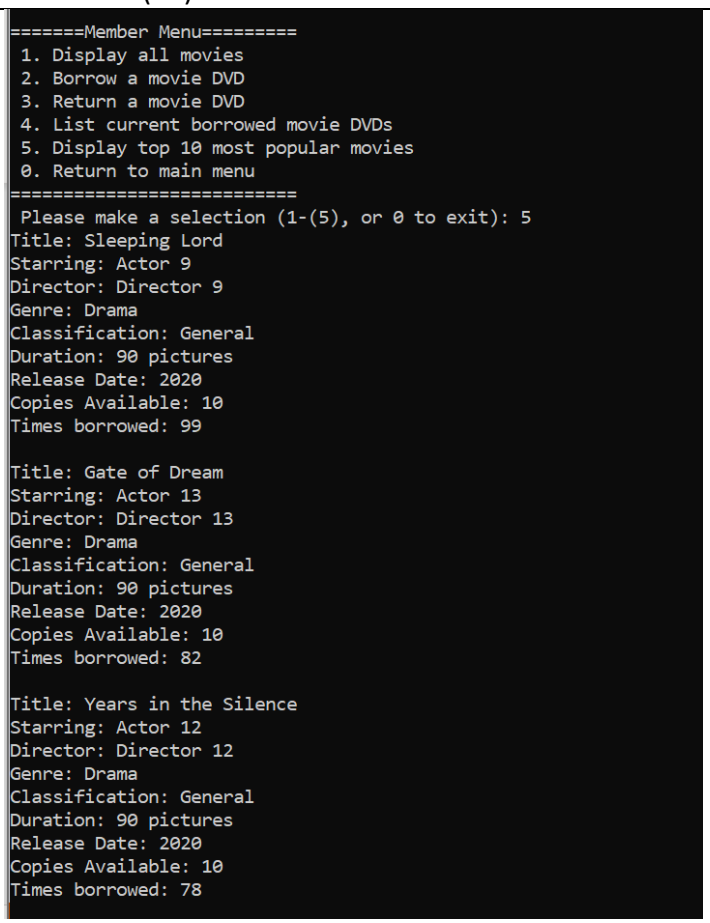
Category	List current borrowed movie DVDs
Input	Pre-condition: Borrow: “Gate of Dream”, “The Each Lights”, “Frozen Girlfriend” (or any movies from the testing set) Option 4;
Expected Output	The movies are in the member’s borrowed list.
Actual Output	
Comment	The actual output and expected output matches.

Category	List current borrowed movie DVDs
Input	Pre-condition: Borrow: “Gate of Dream”, “The Each Lights”, “Frozen Girlfriend” (or any

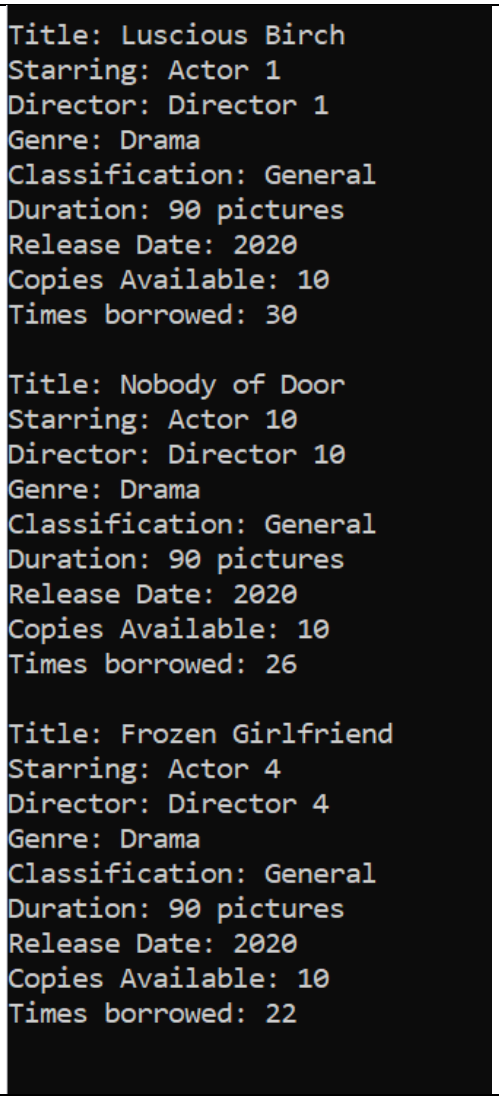
	movies from the testing set). Then Return: “Gate of Dream” and “Frozen Girlfriend”. Option 4;
Expected Output	Only “The Each Lights” is in the member’s borrowed list.
Actual Output	
Comment	The actual output and expected output matches.

Category	List current borrowed movie DVDs
Input	Pre-condition: Borrow: “Gate of Dream”, “The Each Lights”, “Frozen Girlfriend” (or any movies from the testing set). Then return all those movies. Option 4;
Expected Output	No movie is in the member’s borrowed list.
Actual Output	
Comment	The actual output and expected output matches.

Category	Display top 10 most popular movies (Set 1)
Input	Option 5;
Expected Output	Pre-condition: The testing set is tested as it is, without any prior borrowing or returning as the borrowed times have been pre-set.

	The descending popular order should be: "Sleeping Lord"(99), "Gate of Dream"(82), "Years in the Silence"(78), "Alien in the Weeping"(60), “Tears in the Fire”(40),"Rose of Snow"(39), "The Scent's Ships"(31) ,"Luscious Birch" (30), "Nobody of Door”(26), "Frozen Girlfriend"(22)
Actual Output	

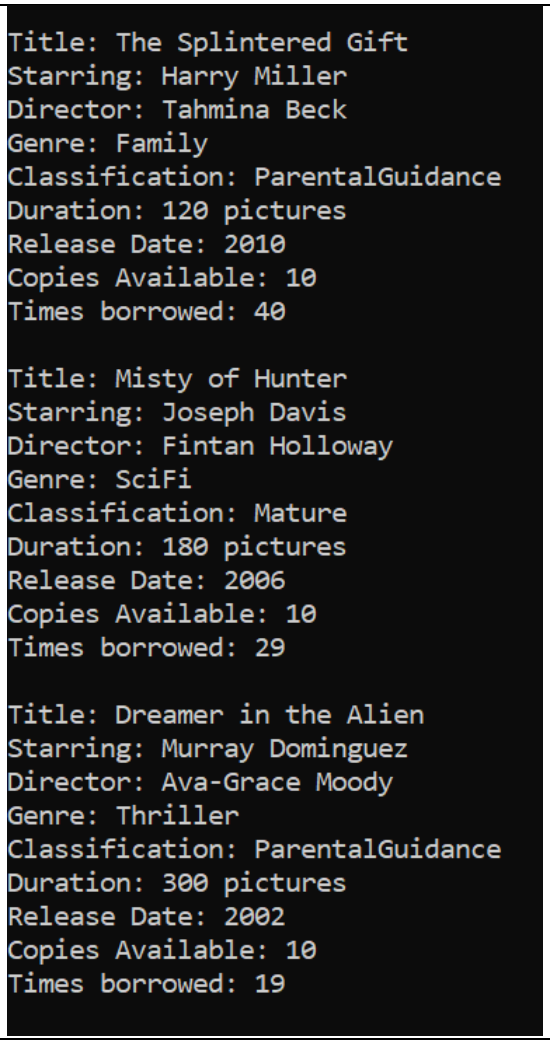
	Title: Alien in the Weeping Starring: Actor 3 Director: Director 3 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 60 Title: Tears in the Fire Starring: Actor 11 Director: Director 11 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 40 Title: Rose of Snow Starring: Actor 5 Director: Director 5 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 39 Title: The Scent's Ships Starring: Actor 6 Director: Director 6 Genre: Drama Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 31

	
Comment	The actual output and expected output matches.

Category	Display top 10 most popular movies (Set 2)
Input	Option: 5;
Expected Output	Pre-condition: The testing set is tested as it is, without any prior borrowing or returning as the borrowed times have been pre-set. The descending popular order should be: "The Dream's Wife" (92), "The Servants of the Lights" (90), "Silken Servant" (88), "Wings in the Emperor"(70), "The Valley of the Ashes" (63), "The Thief's Visions" (51), "The Blue Lover" (50), "The Splintered Gift" (40), "Misty of Hunter"(29), "Dreamer in the Alien" (19)
Actual Output	

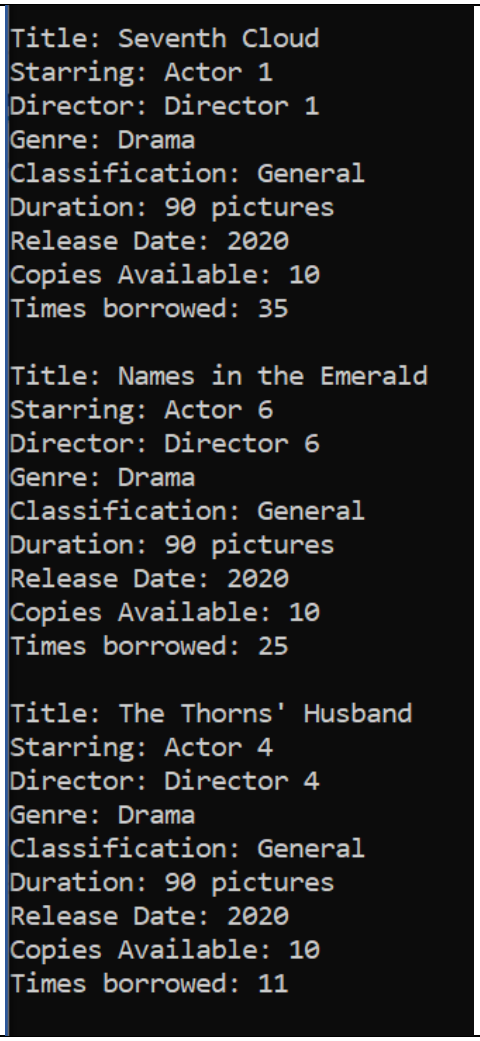
```
=======Member Menu=========
 1. Display all movies
 2. Borrow a movie DVD
 3. Return a movie DVD
 4. List current borrowed movie DVDs
 5. Display top 10 most popular movies
 0. Return to main menu
==========================
 Please make a selection (1-(5), or 0 to exit): 5
Title: The Dream's Wife
Starring: Joshua Smith
Director: Darcy Riddle
Genre: Comedy
Classification: MatureAccompanied
Duration: 90 pictures
Release Date: 2003
Copies Available: 10
Times borrowed: 92

Title: The Servants of the Lights
Starring: Ajay Molloy
Director: Taslima May
Genre: Drama
Classification: MatureAccompanied
Duration: 90 pictures
Release Date: 2001
Copies Available: 10
Times borrowed: 90

Title: Silken Servant
Starring: Daniel Rodriguez
Director: Sammy-Jo Hubbard
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2007
Copies Available: 10
Times borrowed: 88
```

	Title: Wings in the Emperor Starring: Hamza Cardenas Director: Ruqayyah Soto Genre: Comedy Classification: ParentalGuidance Duration: 300 pictures Release Date: 2015 Copies Available: 10 Times borrowed: 70 Title: The Valley of the Ashes Starring: Jack Garcia Director: Jamie-Leigh Genre: Thriller Classification: MatureAccompanied Duration: 270 pictures Release Date: 2012 Copies Available: 10 Times borrowed: 63 Title: The Thief's Visions Starring: Stephan Mcghee Director: Alayna Welch Genre: SciFi Classification: Mature Duration: 270 pictures Release Date: 2011 Copies Available: 10 Times borrowed: 51 Title: The Blue Lover Starring: Leoni Whyte Director: Missy Hawes Genre: Family Classification: General Duration: 90 pictures Release Date: 2020 Copies Available: 10 Times borrowed: 50

	
Comment	The actual output and expected output matches.

Category	Display top 10 most popular movies (Set 3)
Input	Option: 5;
Expected Output	Pre-condition: The testing set is tested as it is, without any prior borrowing or returning as the borrowed times have been pre-set. The descending popular order should be: "The Magic of the Tale" (98), "The Dying Moons" (75), "Whispering Door (74), "Snake of Stream" (39), "Seventh Cloud" (35), "Names in the Emerald" (25), "The Thorns' Husband" (11)
Actual Output	

```
==========================
 Please make a selection (1-(5), or 0 to exit): 5
Title: The Magic of the Tale
Starring: Actor 5
Director: Director 5
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 98

Title: The Dying Moons
Starring: Actor 2
Director: Director 2
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 75

Title: Whispering Door
Starring: Actor 7
Director: Director 7
Genre: Drama
Classification: General
Duration: 90 pictures
Release Date: 2020
Copies Available: 10
Times borrowed: 74

Title: Snake of Stream
Starring: Actor 3
Director: Director 3
Genre: Drama
```

	
Comment	The actual output and expected output matches.

References

Levitin, A. (2011). *Introduction to the Design and Analysis of Algorithms.* Retrieved from https://ebookcentral.proquest.com/lib/qut/detail.action?docID=5175770.

Quiles, L. (n.d.). *Merge Sort*. Retrieved from https://cs.fit.edu/~pkc/classes/writing/hw13/luis.pdf