2020

CAB301 Project Report

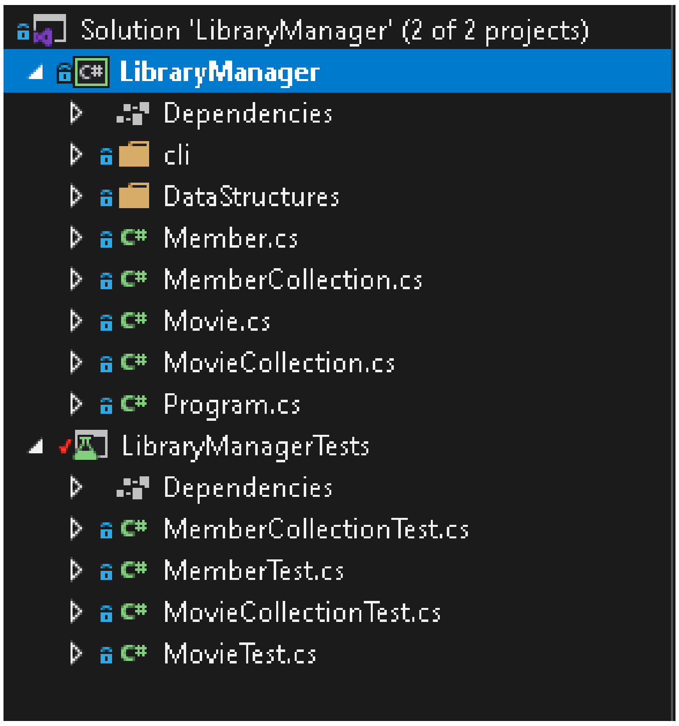
COMMUITY DVD LIBRARY MANAGER DEVELOPED IN C#

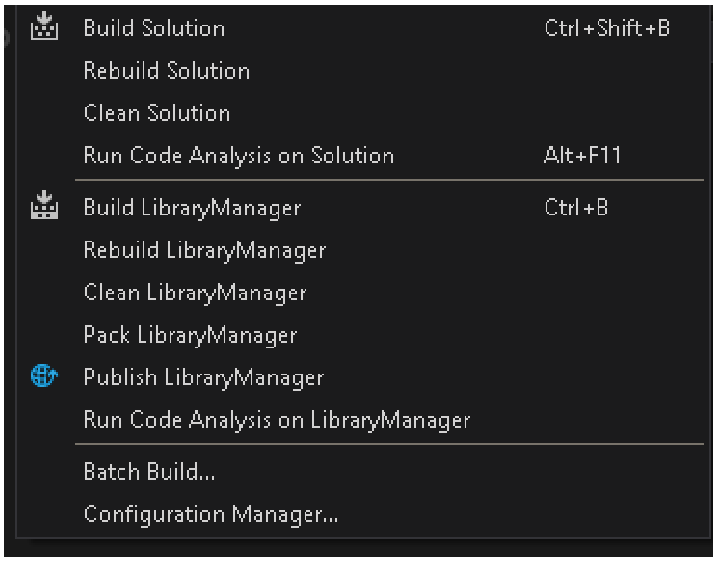
NICHOLAS KRESS, N9467688

CONTENTS

NOTE ON BUILDING PROJECT

The NUnit testing framework was used to test program functionality (See Unit Testing pg 10). The Program can still be run without NUnit installed. To ensure the project Builds Properly without NUnit testing framework installed, select LibraryManager in solution explorer and the run Build by selecting *Build -> Build LibraryManager* or (Ctrl+B).





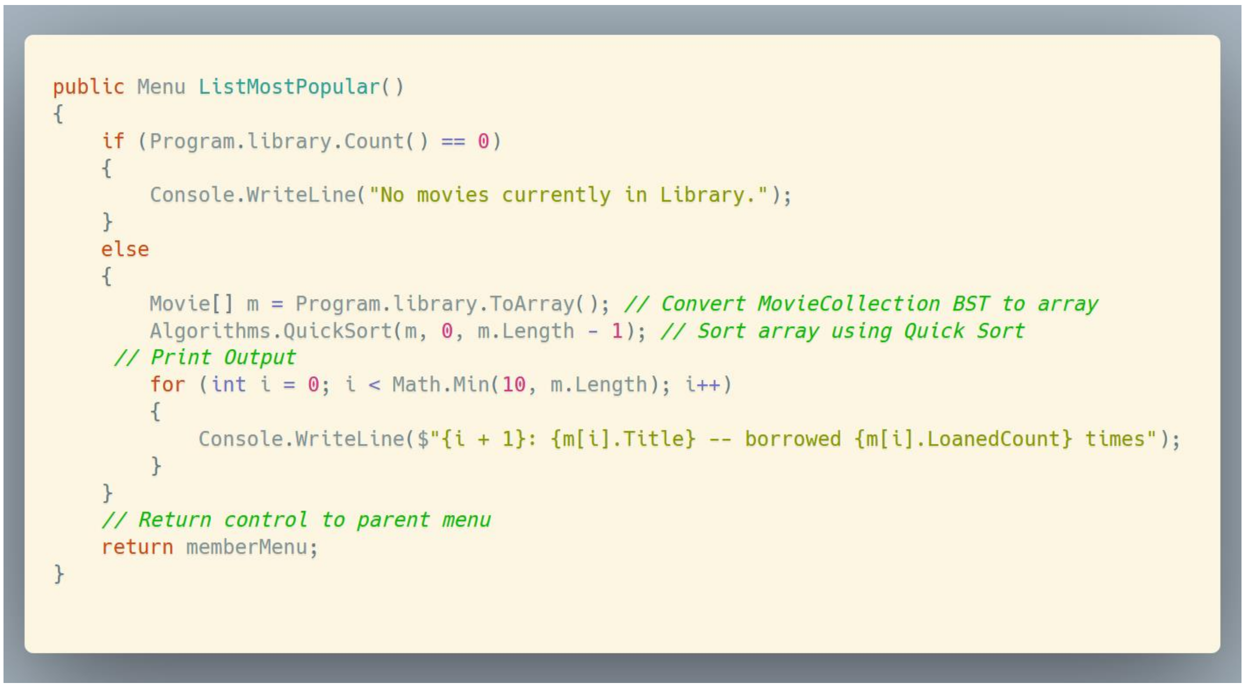
Or, if running from the console, move to the directory *./Library*Manager/ in the project directory before running

```
$ dotnet run
```

TOP 10 FUNCTION

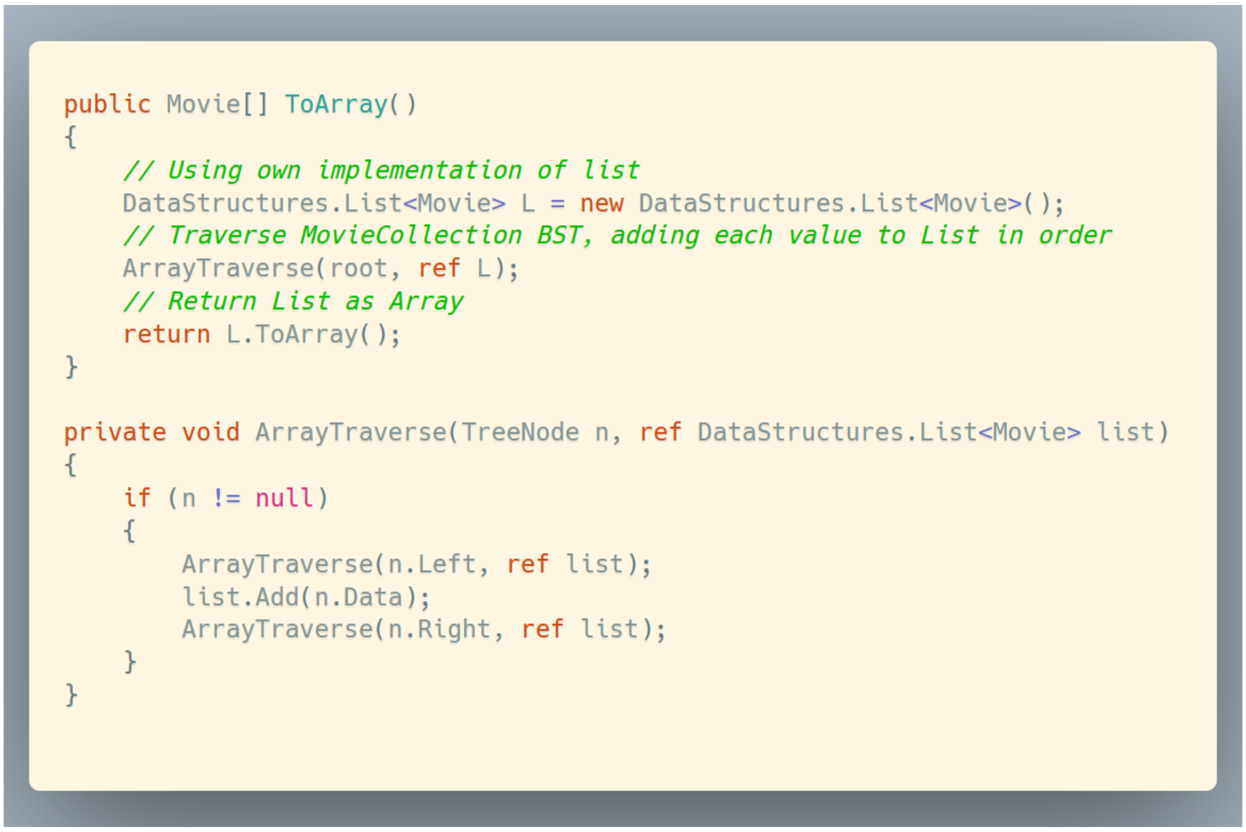
The code for the function ListMostPopular and all functions encapsulated in this function are displayed below. Each heading describes the functions folder/file location within the project.

CLI.MENU.LISTMOSTPOPULAR: PRINT OUT 10 MOST POPULAR MOVIES WITH COUNTS TO THE CONSOLE



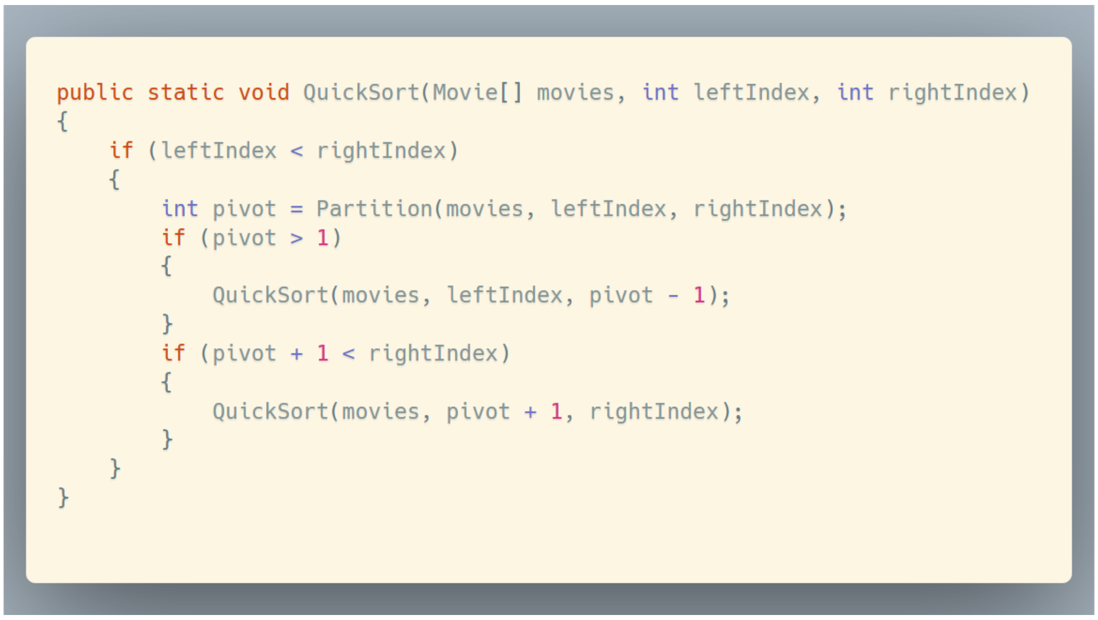
```
public Menu ListMostPopular()
{
    if (Program.library.Count() == 0)
    {
        Console.WriteLine("No movies currently in Library.");
    }
    else
    {
        Movie[] m = Program.library.ToArray(); // Convert MovieCollection BST to array
        Algorithms.QuickSort(m, 0, m.Length - 1); // Sort array using Quick Sort
     // Print Output
        for (int i = 0; i < Math.Min(10, m.Length); i++)
        {
            Console.WriteLine($"{i + 1}: {m[i].Title} -- borrowed {m[i].LoanedCount} times");
        }
    }
    // Return control to parent menu
    return memberMenu;
}
```

MOVIECOLLECTION.TOARRAY: RETURN AN ARRAY OF MOVIES FROM THE CURRENT MOVIECOLLECTION OBJECT



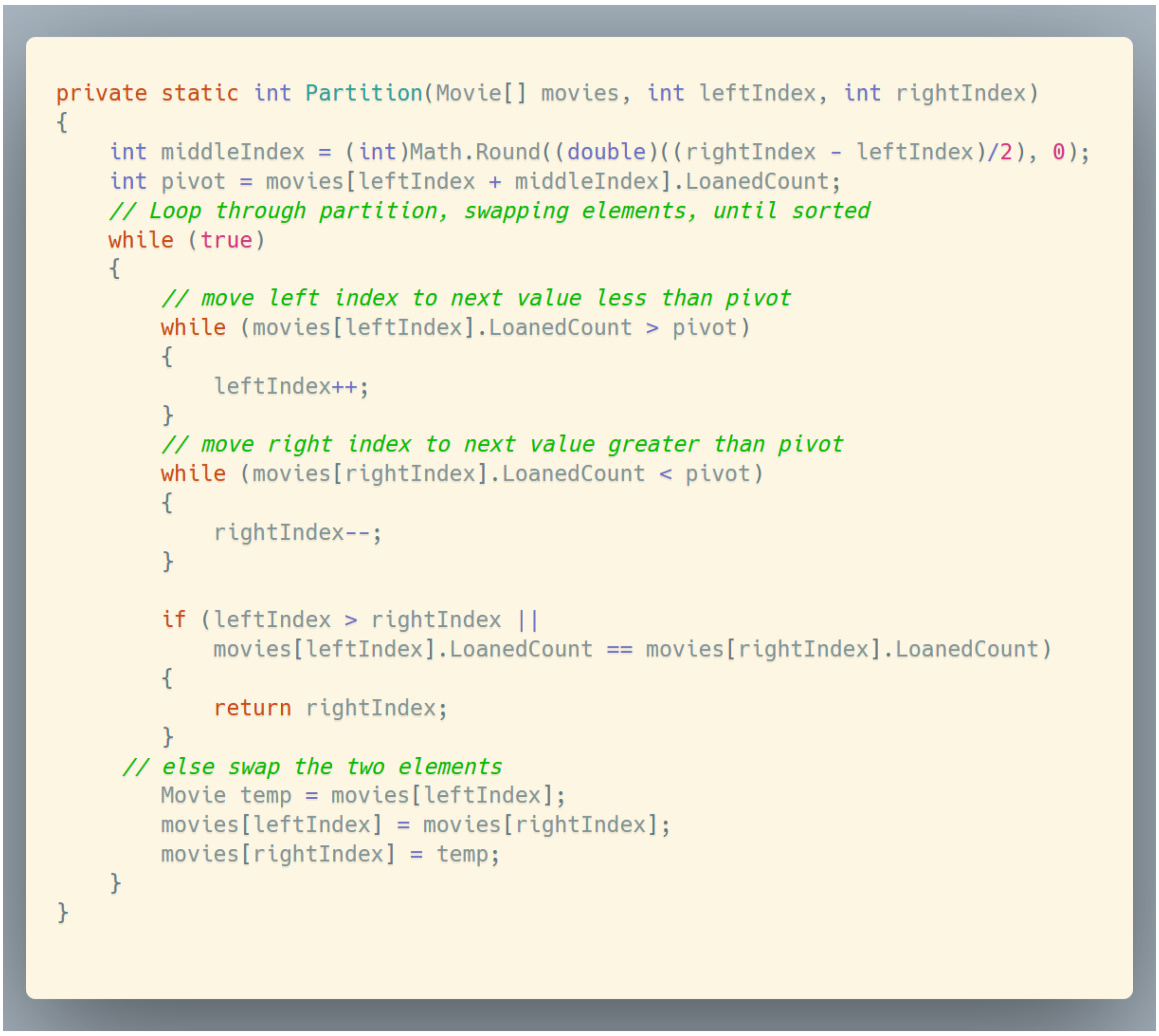
```
public Movie[] ToArray()
{
    // Using own implementation of list
    DataStructures.List<Movie> L = new DataStructures.List<Movie>();
    // Traverse MovieCollection BST, adding each value to List in order
    ArrayTraverse(root, ref L);
    // Return List as Array
    return L.ToArray();
}

private void ArrayTraverse(TreeNode n, ref DataStructures.List<Movie> list)
{
    if (n != null)
    {
        ArrayTraverse(n.Left, ref list);
        list.Add(n.Data);
        ArrayTraverse(n.Right, ref list);
    }
}
```

DATASTRUCTURES.ALGORITHMS.QUICKSORT: SORT AN ARRAY OF MOVIES BY LOANCOUNT IN DESCENDING ORDER



```
public static void QuickSort(Movie[] movies, int leftIndex, int rightIndex)
{
    if (leftIndex < rightIndex)
    {
        int pivot = Partition(movies, leftIndex, rightIndex);
        if (pivot > 1)
        {
            QuickSort(movies, leftIndex, pivot - 1);
        }
        if (pivot + 1 < rightIndex)
        {
            QuickSort(movies, pivot + 1, rightIndex);
        }
    }
}
```

DATASTRUCTURES.ALGORITHMS.PARTITION: SORT THE ARRAY BETWEEN THE CURRENT INDEXES AND RETURN THE PIVOT INDEX



```
private static int Partition(Movie[] movies, int leftIndex, int rightIndex)
{
    int middleIndex = (int)Math.Round((double)((rightIndex - leftIndex)/2), 0);
    int pivot = movies[leftIndex + middleIndex].LoanedCount;
    // Loop through partition, swapping elements, until sorted
    while (true)
    {
        // move left index to next value less than pivot
        while (movies[leftIndex].LoanedCount > pivot)
        {
            leftIndex++;
        }
        // move right index to next value greater than pivot
        while (movies[rightIndex].LoanedCount < pivot)
        {
            rightIndex--;
        }

        if (leftIndex > rightIndex ||
            movies[leftIndex].LoanedCount == movies[rightIndex].LoanedCount)
        {
            return rightIndex;
        }
     // else swap the two elements
        Movie temp = movies[leftIndex];
        movies[leftIndex] = movies[rightIndex];
        movies[rightIndex] = temp;
    }
}
```

ALGORITHM ANALYSIS

The *ListTopTen* functionality can be split into 3 sections: Flattened the array, Sort this flattened array, and print the sorted & flattened array.

FLATTEN TIME COMPLEXITY

The function for flattening the BST to an array simply traverses the BST and adds each element to a new array. Thus the time complexity of the function *MovieCollection.ToArray* is always $C(n) \in \Theta(n)$.

QUICKSORT TIME COMPLEXITY

The sorting algorithm used to sort the array is a modified version of a divide and conquer algorithm: Quicksort provided by *Levitin, A. (2012) Chapter 5.2: Quicksort. Introduction to The Design and Analysis of Algorithms. (3rd Edition, pp. 176-178).* The Algorithm has been modified to sort the array in *descending* order rather than the ascending order in which the original algorithm sorts the array. The pseudocode of the resulting algorithm is written below, with the basic operation highlighted.

```
QuickSort(A[1..n), LeftIndex, RightIndex)
     if LeftIndex < RightIndex then
          split ← null
          pivot ← A[LeftIndex + (RightIndex - LeftIndex)/2]
          i ← LeftIndex
          j ← RightIndex
               while split = null do
                    repeat i ← i + 1 until A[i] <= pivot
                    repeat j ← j + 1 until A[j] >= pivot
                    if i > j or A[i] == A[j]
                         split ← j
                    else
                         Temp ← A[i]
                         A[i] ← A[j]
                         A[j] ← Temp
          if split > 1
               QuickSort(A, LeftIndex, split - 1)
          if split + 1 < RightIndex
               QuickSort(A, split + 1, RightIndex)
```

According to *Tang. M (2020). CAB301 Algorithms and Complexity: Advanced Sorting Algorithms [Lecture 5 slides]*, the time complexity of Quicksort is:

- $C_{worst}(n) \in \Theta(n^2)$
- $C_{best}(n) \in \Theta(n \log n)$
- $C_{avg}(n) \in \Theta(n \log n)$

PRINT TIME COMPLEXITY

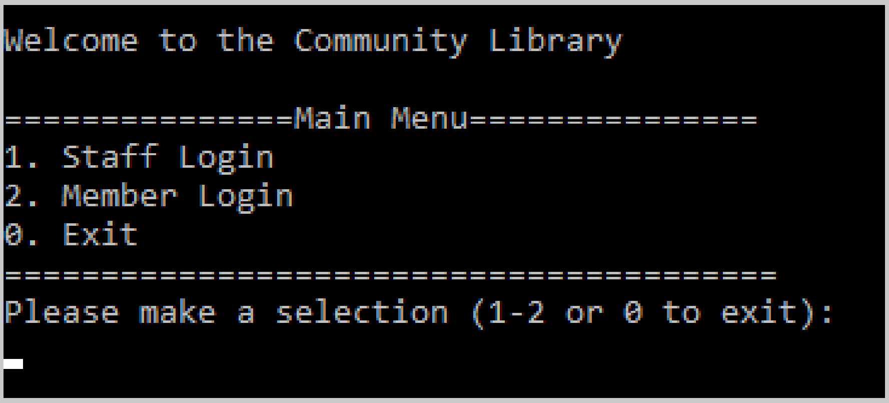
The print function simply loops through the now sorted array and prints out each value, meaning it has a time complexity of $C(n) \in \Theta(n)$. It would be more efficient to print out each value while sorting the array, however Quicksort does not operate in a way in which we could achieve this.

The total time complexity of the function is $C(n) \in \Theta(n \log n + 2n) \in \Theta(n \log n)$.

TEST CASES

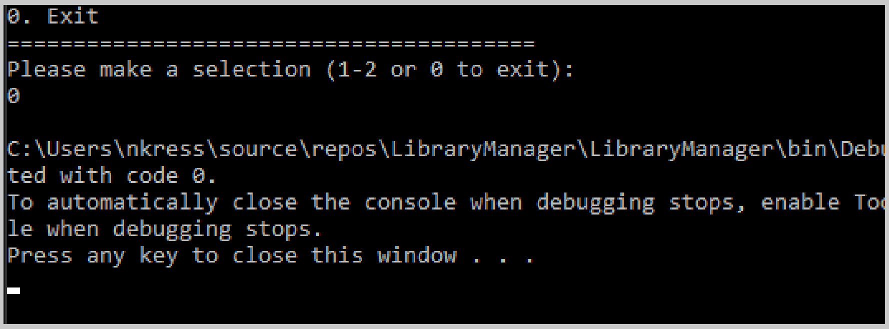
MAIN MENU

On application open, Main menu is displayed



```
Welcome to the Community Library

===============Main Menu===============
1. Staff Login
2. Member Login
0. Exit
=======================================
Please make a selection (1-2 or 0 to exit):
```

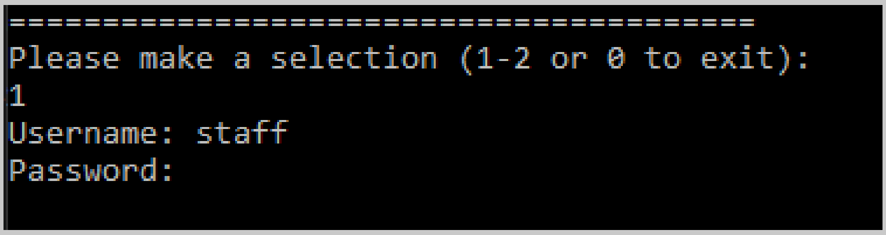
Pressing 0 at main menu exits program



```
0. Exit
=======================================
Please make a selection (1-2 or 0 to exit):
0

C:\Users\nkress\source\repos\LibraryManager\LibraryManager\bin\Debu
ted with code 0.
To automatically close the console when debugging stops, enable Too
le when debugging stops.
Press any key to close this window . . .
```

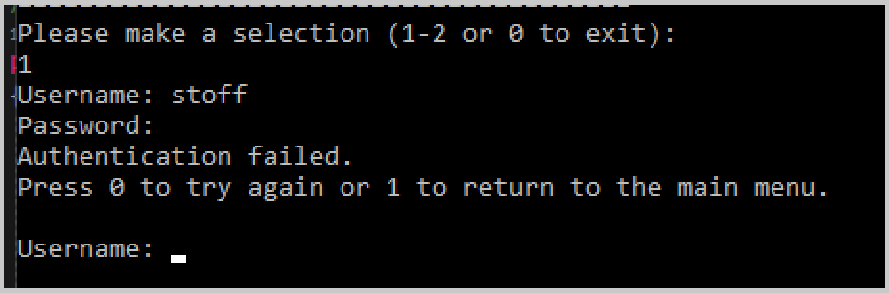
STAFF MENU

Staff can login with username “staff” and password “today123”



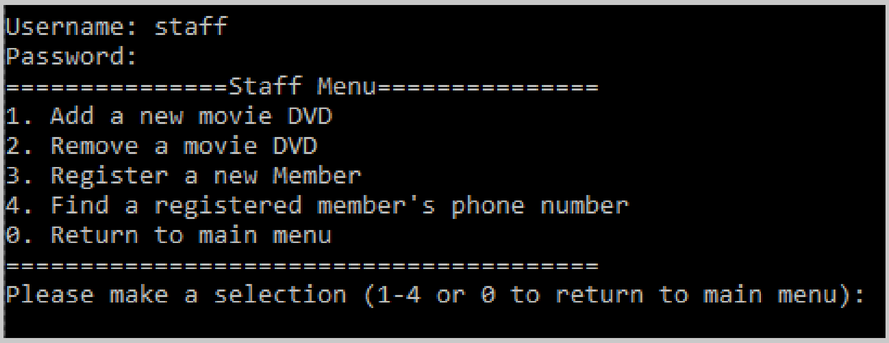
```
=======================================
Please make a selection (1-2 or 0 to exit):
1
Username: staff
Password:
```

Using another username or password does not allow access



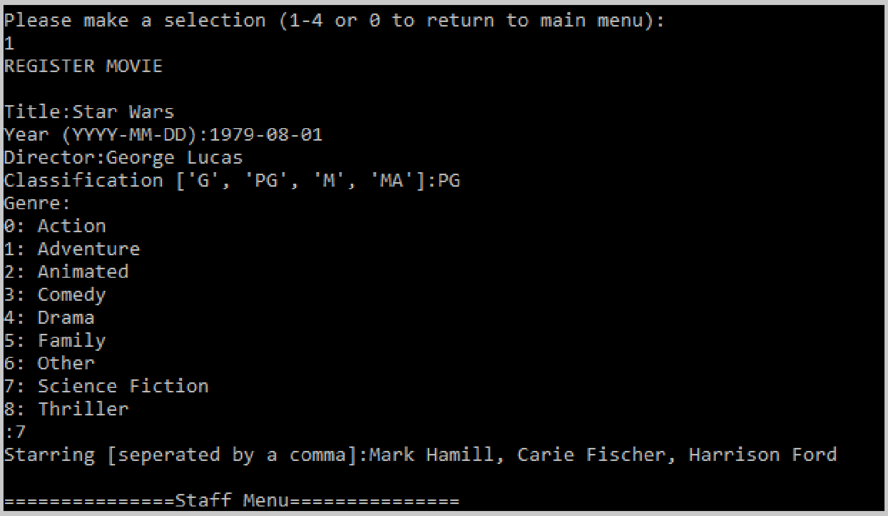
```
=======================================
Please make a selection (1-2 or 0 to exit):
1
Username: stoff
Password:
Authentication failed.
Press 0 to try again or 1 to return to the main menu.

Username:
```

Staff Menu displays on access granted



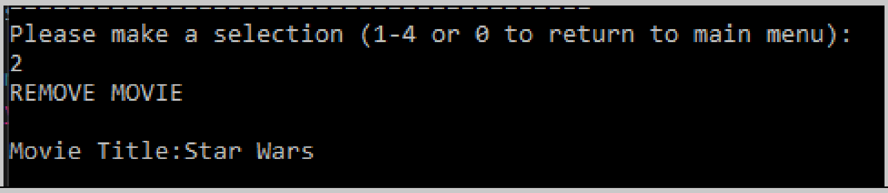
```
Username: staff
Password:
==============Staff Menu==============
1. Add a new movie DVD
2. Remove a movie DVD
3. Register a new Member
4. Find a registered member's phone number
0. Return to main menu
======================================
Please make a selection (1-4 or 0 to return to main menu):
```

Register a new movie



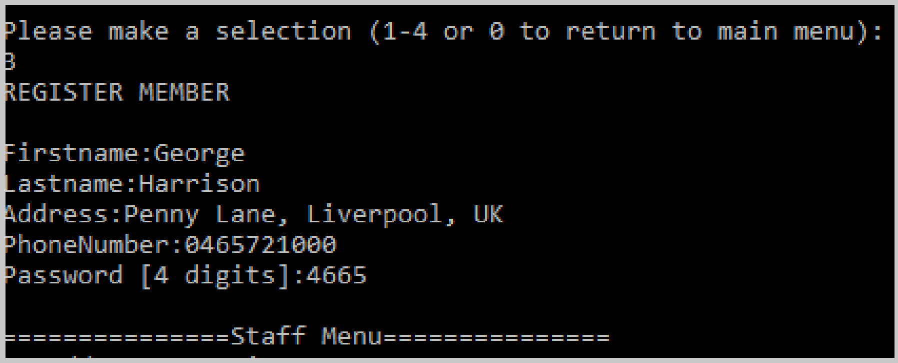
```
Please make a selection (1-4 or 0 to return to main menu):
1
REGISTER MOVIE

Title:Star Wars
Year (YYYY-MM-DD):1979-08-01
Director:George Lucas
Classification ['G', 'PG', 'M', 'MA']:PG
Genre:
0: Action
1: Adventure
2: Animated
3: Comedy
4: Drama
5: Family
6: Other
7: Science Fiction
8: Thriller
:7
Starring [seperated by a comma]:Mark Hamill, Carie Fischer, Harrison Ford

==============Staff Menu==============
```

Remove a movie



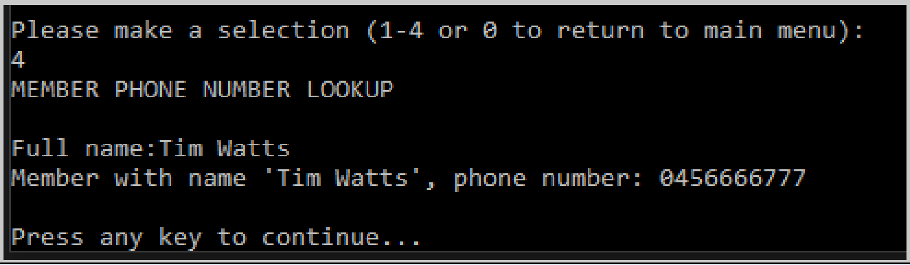
```
----------------------------------------
Please make a selection (1-4 or 0 to return to main menu):
2
REMOVE MOVIE

Movie Title:Star Wars
```

Register a new member



```
Please make a selection (1-4 or 0 to return to main menu):
3
REGISTER MEMBER

Firstname:George
Lastname:Harrison
Address:Penny Lane, Liverpool, UK
PhoneNumber:0465721000
Password [4 digits]:4665

==============Staff Menu==============
```

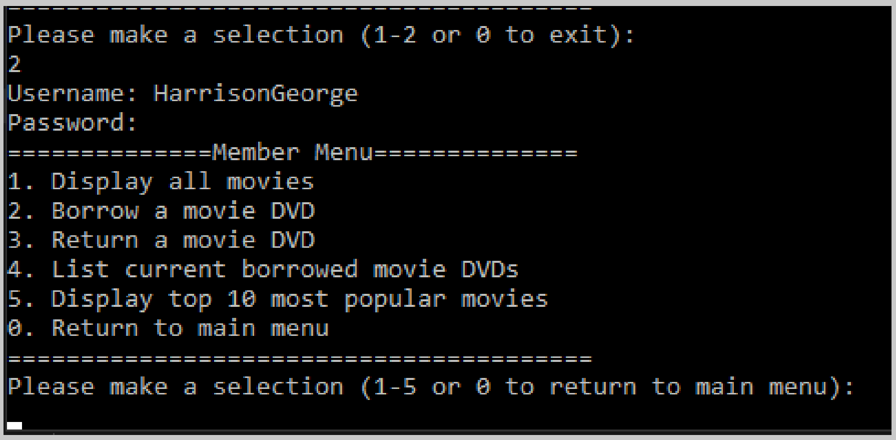
Find members phone number by full name



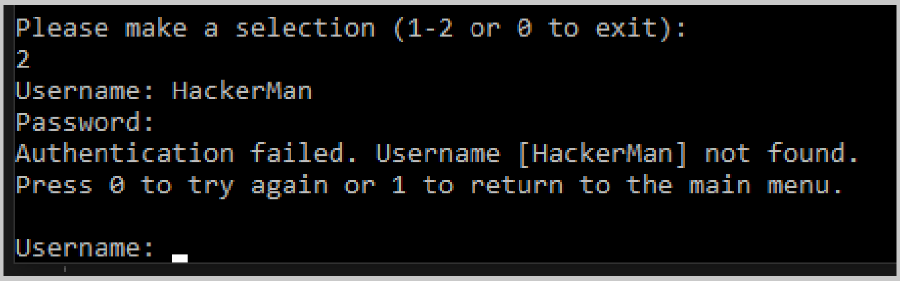
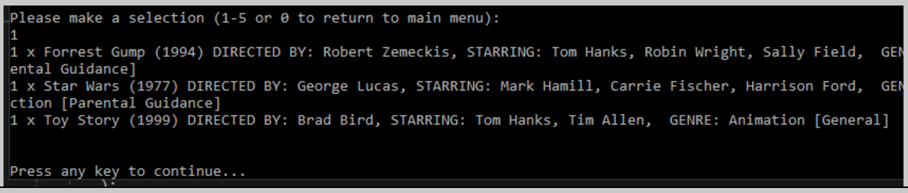
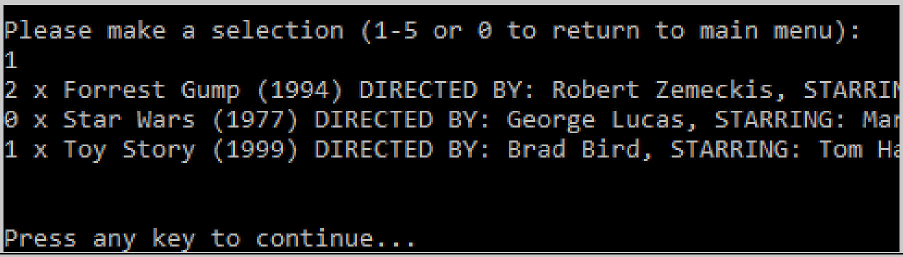
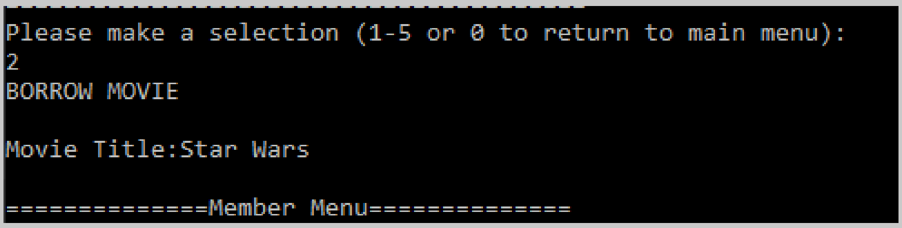
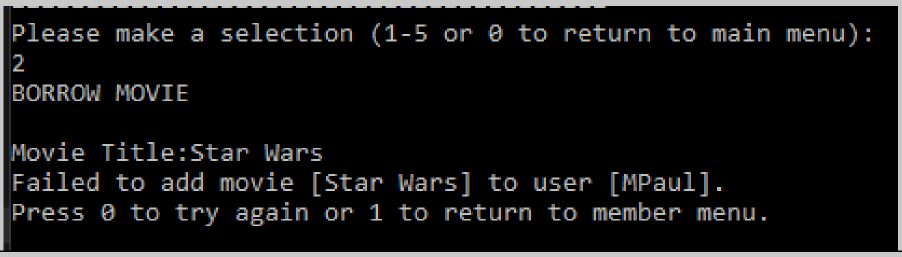
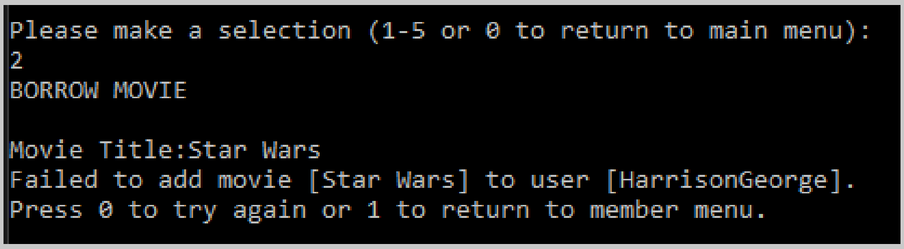
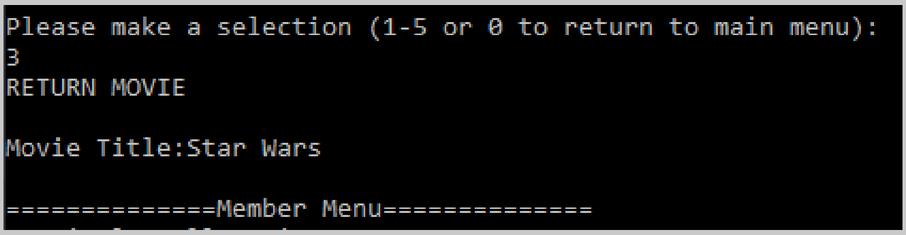
```
Please make a selection (1-4 or 0 to return to main menu):
4
MEMBER PHONE NUMBER LOOKUP

Full name:Tim Watts
Member with name 'Tim Watts', phone number: 0456666777

Press any key to continue...
```

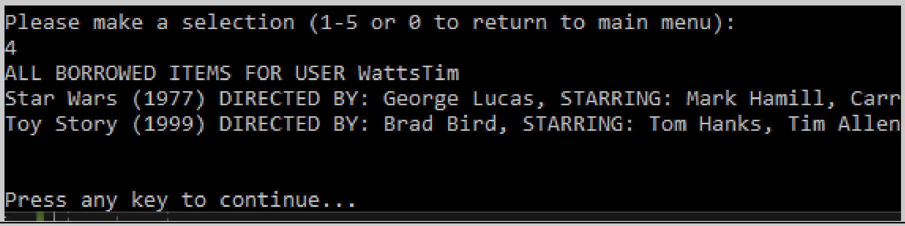
MEMBER MENU

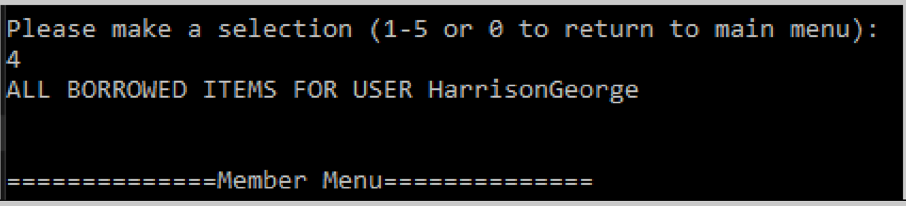
Member can login with username and password entered by staff

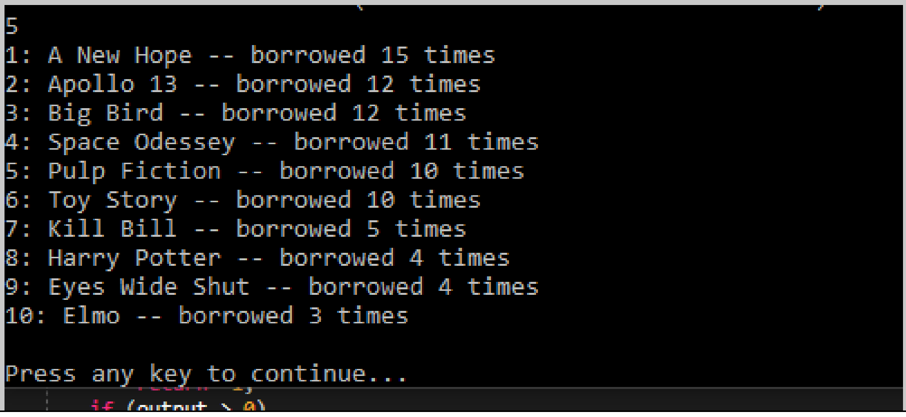


```
----------------------------------------
Please make a selection (1-2 or 0 to exit):
2
Username: HarrisonGeorge
Password:
==============Member Menu==============
1. Display all movies
2. Borrow a movie DVD
3. Return a movie DVD
4. List current borrowed movie DVDs
5. Display top 10 most popular movies
0. Return to main menu
========================================
Please make a selection (1-5 or 0 to return to main menu):
```

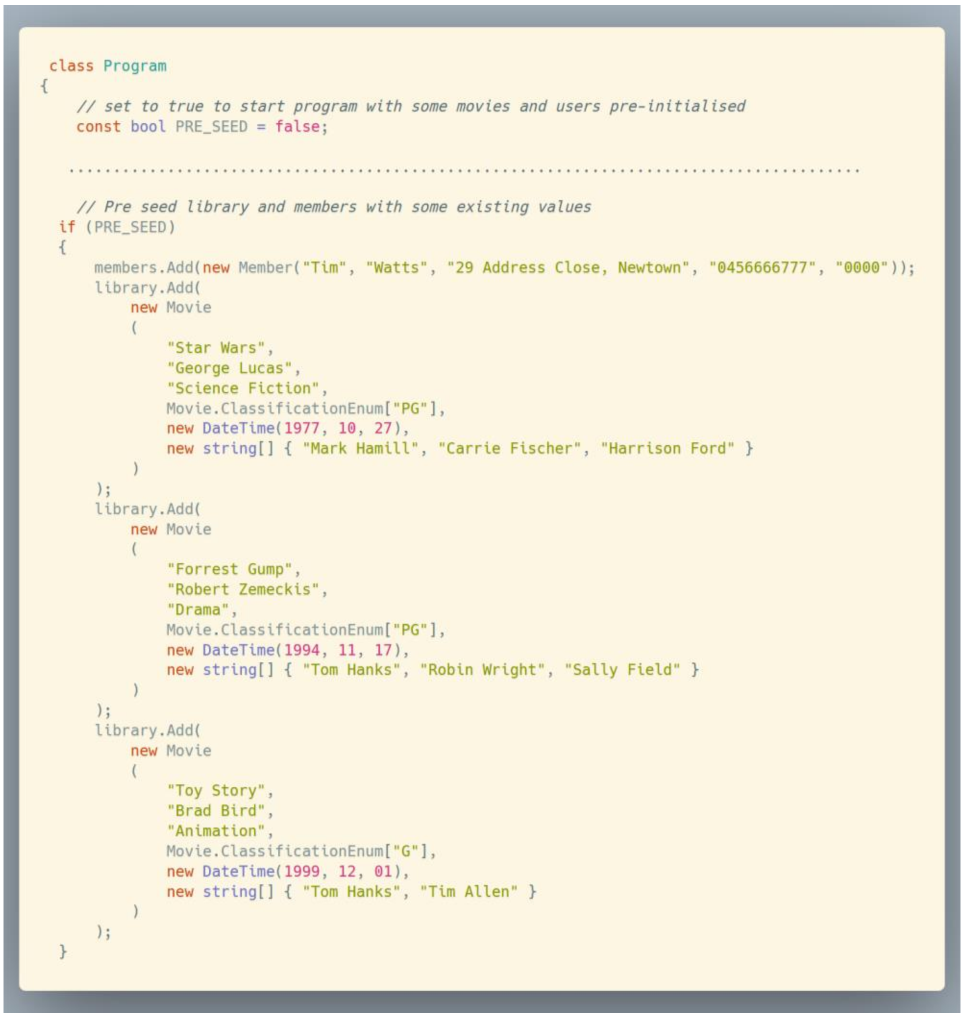
Using bad credentials does not allow login	 Please make a selection (1-2 or 0 to exit): 2 Username: HackerMan Password: Authentication failed. Username [HackerMan] not found. Press 0 to try again or 1 to return to the main menu. Username:
Member can display all movies in *MovieCollection*. Each movie is prefixed by the current copies available.	 Please make a selection (1-5 or 0 to return to main menu): 1 1 x Forrest Gump (1994) DIRECTED BY: Robert Zemeckis, STARRING: Tom Hanks, Robin Wright, Sally Field, GEN ental Guidance] 1 x Star Wars (1977) DIRECTED BY: George Lucas, STARRING: Mark Hamill, Carrie Fischer, Harrison Ford, GEN ction [Parental Guidance] 1 x Toy Story (1999) DIRECTED BY: Brad Bird, STARRING: Tom Hanks, Tim Allen, GENRE: Animation [General] Press any key to continue...
All movies changes when a new copy is added or movie is loaned.	 Please make a selection (1-5 or 0 to return to main menu): 1 2 x Forrest Gump (1994) DIRECTED BY: Robert Zemeckis, STARRI 0 x Star Wars (1977) DIRECTED BY: George Lucas, STARRING: Ma 1 x Toy Story (1999) DIRECTED BY: Brad Bird, STARRING: Tom H Press any key to continue...
Member can borrow a movie	 Please make a selection (1-5 or 0 to return to main menu): 2 BORROW MOVIE Movie Title:Star Wars ==============Member Menu==============
Another member cannot borrow same movie with only single copy	 Please make a selection (1-5 or 0 to return to main menu): 2 BORROW MOVIE Movie Title:Star Wars Failed to add movie [Star Wars] to user [MPaul]. Press 0 to try again or 1 to return to member menu.
Member cannot borrow movie already checked out	 Please make a selection (1-5 or 0 to return to main menu): 2 BORROW MOVIE Movie Title:Star Wars Failed to add movie [Star Wars] to user [HarrisonGeorge]. Press 0 to try again or 1 to return to member menu.
Member can return a movie	 Please make a selection (1-5 or 0 to return to main menu): 3 RETURN MOVIE Movie Title:Star Wars ==============Member Menu==============

Member can display all movies when member has on loan	
	`Please make a selection (1-5 or 0 to return to main menu):`
	`4`
	`ALL BORROWED ITEMS FOR USER WattsTim`
	`Star Wars (1977) DIRECTED BY: George Lucas, STARRING: Mark Hamill, Carr`
	`Toy Story (1999) DIRECTED BY: Brad Bird, STARRING: Tom Hanks, Tim Allen`
	
	`Press any key to continue...`

Display all movies when no DVDs on loan	
	`Please make a selection (1-5 or 0 to return to main menu):`
	`4`
	`ALL BORROWED ITEMS FOR USER HarrisonGeorge`
	
	`==============Member Menu==============`

Display top 10 most borrowed movies, only display 10 movies if more are in collection	
	`5`
	`1: A New Hope -- borrowed 15 times`
	`2: Apollo 13 -- borrowed 12 times`
	`3: Big Bird -- borrowed 12 times`
	`4: Space Odessey -- borrowed 11 times`
	`5: Pulp Fiction -- borrowed 10 times`
	`6: Toy Story -- borrowed 10 times`
	`7: Kill Bill -- borrowed 5 times`
	`8: Harry Potter -- borrowed 4 times`
	`9: Eyes Wide Shut -- borrowed 4 times`
	`10: Elmo -- borrowed 3 times`
	
	`Press any key to continue...`

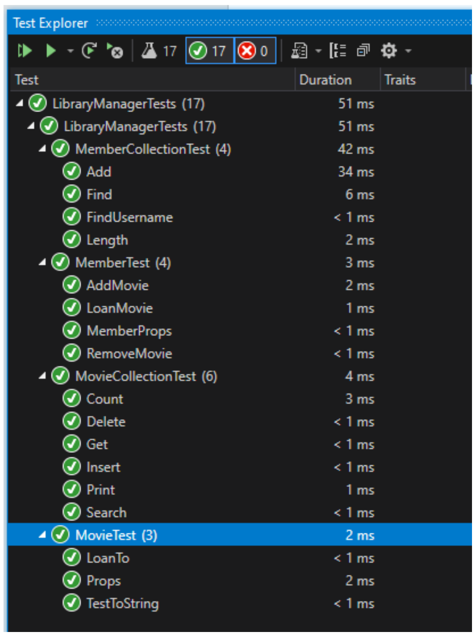
As seen in the excerpt from the Program class below, the program can be run with a user [WattsTim, 0000] and several movies already added to the library. This is disabled by default but can be enabled by changing the PRE_SEED flag to *true* before building the program.



```
class Program
{
    // set to true to start program with some movies and users pre-initialised
    const bool PRE_SEED = false;

  ....................................................................................

    // Pre seed library and members with some existing values
  if (PRE_SEED)
  {
      members.Add(new Member("Tim", "Watts", "29 Address Close, Newtown", "0456666777", "0000"));
      library.Add(
          new Movie
          (
              "Star Wars",
              "George Lucas",
              "Science Fiction",
              Movie.ClassificationEnum["PG"],
              new DateTime(1977, 10, 27),
              new string[] { "Mark Hamill", "Carrie Fischer", "Harrison Ford" }
          )
      );
      library.Add(
          new Movie
          (
              "Forrest Gump",
              "Robert Zemeckis",
              "Drama",
              Movie.ClassificationEnum["PG"],
              new DateTime(1994, 11, 17),
              new string[] { "Tom Hanks", "Robin Wright", "Sally Field" }
          )
      );
      library.Add(
          new Movie
          (
              "Toy Story",
              "Brad Bird",
              "Animation",
              Movie.ClassificationEnum["G"],
              new DateTime(1999, 12, 01),
              new string[] { "Tom Hanks", "Tim Allen" }
          )
      );
  }
```

UNIT TESTING

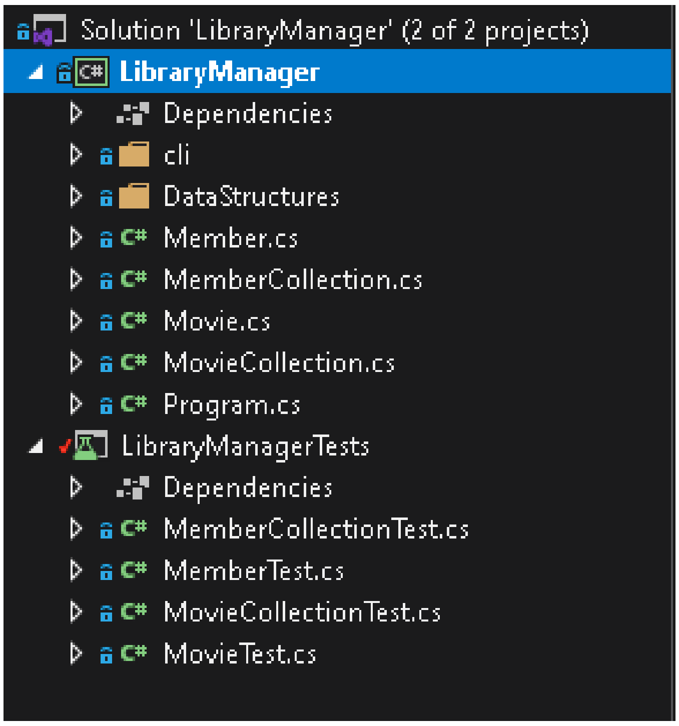
The functionality of the classes *Member, MemberCollection, Movie* and *MovieCollection* have been tested using the NUnit framework in the project *LibraryManagerTests*. The NUnit is required to run these tests. If this framework is not installed, the tests cannot be built/run but the main *LibraryManager* Project can still be built independantly of the Test project.



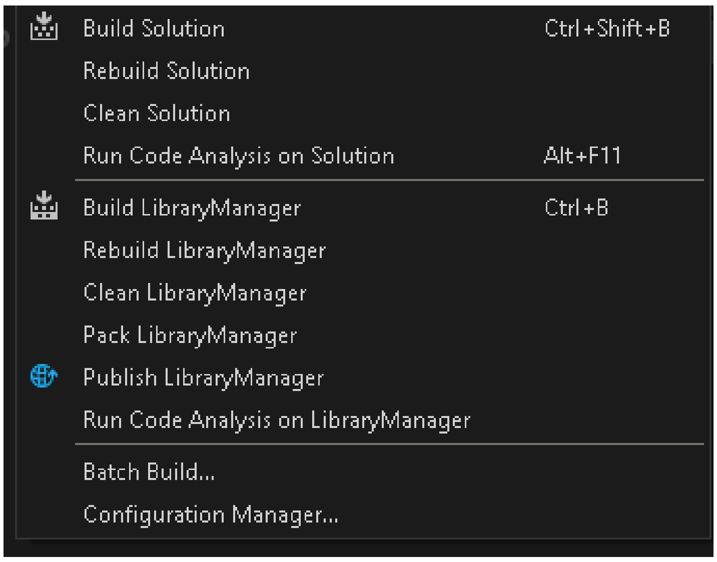
Test Explorer

Test	Duration	Traits
LibraryManagerTests (17)	51 ms	
LibraryManagerTests (17)	51 ms	
MemberCollectionTest (4)	42 ms	
Add	34 ms	
Find	6 ms	
FindUsername	< 1 ms	
Length	2 ms	
MemberTest (4)	3 ms	
AddMovie	2 ms	
LoanMovie	1 ms	
MemberProps	< 1 ms	
RemoveMovie	< 1 ms	
MovieCollectionTest (6)	4 ms	
Count	3 ms	
Delete	< 1 ms	
Get	< 1 ms	
Insert	< 1 ms	
Print	1 ms	
Search	< 1 ms	
MovieTest (3)	2 ms	
LoanTo	< 1 ms	
Props	2 ms	
TestToString	< 1 ms	

To ensure the project Builds Properly without NUnit installed, select LibraryManager in solution explorer and the run Build by selecting *Build -> Build LibraryManager* or (Ctrl+B).



```
Solution 'LibraryManager' (2 of 2 projects)
  LibraryManager
    Dependencies
    cli
    DataStructures
    Member.cs
    MemberCollection.cs
    Movie.cs
    MovieCollection.cs
    Program.cs
  LibraryManagerTests
    Dependencies
    MemberCollectionTest.cs
    MemberTest.cs
    MovieCollectionTest.cs
    MovieTest.cs
```



```
Build Solution                         Ctrl+Shift+B
Rebuild Solution
Clean Solution
Run Code Analysis on Solution          Alt+F11
Build LibraryManager                   Ctrl+B
Rebuild LibraryManager
Clean LibraryManager
Pack LibraryManager
Publish LibraryManager
Run Code Analysis on LibraryManager
Batch Build...
Configuration Manager...
```