Movie LINQs

Collections are great for storing multiple instances of a data type, but it’s often the case that we need to filter, manipulate, or analyze this data in bulk. C# includes LINQ (language integrated query) which allows us to query our data much like a database.

For this assignment, you will be supplied with a DLL containing movie data to be analyzed. Using this provided DLL, you will write LINQ statements to answer a series of questions.

# Requirements:

* Create a new project (I recommend a console project, but you may use Win Forms or WPF as you wish).
* Download and integrate the provided DLL.
* Using the classes and built-in data from the DLL, write LINQ statements to answer the queries listed below.
  + For each query, copy/paste your full LINQ statement(s) (and any associated code as needed) as well as the answer result for the query into this document. Paste your answer beneath the appropriate query, please.

# Queries

1. What are the years of the oldest and newest movies in the list?

var NewMovie = from Movie in MovieLoader.AllMovies

orderby Movie.Year descending

select Movie;

Console.WriteLine("Newest Movie: ");

foreach (var Movie in NewMovie.Take(1))

{

Console.WriteLine($"{Movie.Title,-20} : {Movie.Year,10}");

}

var OldMovie = from Movie in MovieLoader.AllMovies

orderby Movie.Year ascending

select Movie;

Console.WriteLine("Oldest Movie: ");

foreach (var Movie in OldMovie.Take(1))

{

Console.WriteLine($"{Movie.Title,-20} : {Movie.Year,10}");

}

Newest Movie: Lord of the Rings: The Fellowship of the Ring: 2001

Oldest Movie: Intolerance: 1916

1. Split the total year range in half. Movies on the lower half are considered “classic” while those on the upper half are considered “new”. How many classic movies are there? How many new movies?

int avg = 0;

int totYear = 0;

foreach (var Movie in MovieLoader.AllMovies)

{

totYear += Movie.Year;

}

avg = totYear / MovieLoader.AllMovies.Count();

var Classic = from Movie in MovieLoader.AllMovies

orderby Movie.Year descending

where Movie.Year < avg

select Movie;

Console.WriteLine("All the Classics: ");

foreach (var Movie in Classic)

{

Console.WriteLine($"{Movie.Title,-20} : {Movie.Year,10}");

}

Console.WriteLine($"Total Classics: {Classic.Count()}");

var Newbies = from Movie in MovieLoader.AllMovies

orderby Movie.Year descending

where Movie.Year >= avg

select Movie;

Console.WriteLine("All the Newbies: ");

foreach (var Movie in Newbies)

{

Console.WriteLine($"{Movie.Title,-20} : {Movie.Year,10}");

}

Console.WriteLine($"Total Newbies: {Newbies.Count()}");

47 Classics, 53 Newbies

1. Are there more odd years or even years in the list? Show the quantities of both, please.

var Evens = from Movie in MovieLoader.AllMovies

where Movie.Year % 2 == 0

select Movie;

var Odds = from Movie in MovieLoader.AllMovies

where Movie.Year % 2 != 0

select Movie;

Console.WriteLine($"Total Number of Even-Yeared Movies: {Evens.Count()}");

Console.WriteLine($"Total Number of Odd-Yeared Movies: {Odds.Count()}");

51 Even, 49 Odd

1. Which year contains the most titles? What is the quantity?

var PopYear = from Movie in MovieLoader.AllMovies

group Movie by Movie.Year into g

orderby g.Count() descending

select g;

Console.WriteLine("Most Movies: ");

foreach (var Movie in PopYear.Take(1))

{

Console.WriteLine($"Year: {Movie.Key}, Number of Movies: {Movie.Count()}");

}

4 movies in 1982

1. Which rating shows up the most? What is the quantity?

var PopRating = from Movie in MovieLoader.AllMovies

group Movie by Movie.Rating into g

orderby g.Count() descending

select g;

Console.WriteLine("Most Movies per Rating: ");

foreach (var Movie in PopRating.Take(1))

{

Console.WriteLine($"Rating: {Movie.Key}, Number of Movies: {Movie.Count()}");

}

NotRated, shows up 32 Times

1. Which title is the shortest (in character length)? Which is the longest? Your answer should be the titles themselves (but no other data).

var ShortTitle = from Movie in MovieLoader.AllMovies

orderby Movie.Title.Length ascending

select Movie;

foreach (var Movie in ShortTitle.Take(1))

{

Console.WriteLine($"Shortest Title: {Movie.Title}");

}

var LongTitle = from Movie in MovieLoader.AllMovies

orderby Movie.Title.Length descending

select Movie;

foreach (var Movie in LongTitle.Take(1))

{

Console.WriteLine($"Longest Title: {Movie.Title}");

}

Shortest is Jaws, Longest is The Lord of the Rings: The Fellowship of the Ring

1. Which rating has the widest range in years?

int Distance = 0, LowYear = 0, HighYear = 0, RcrdDist = 0;

MPAARating LongBoi = MPAARating.NotRated;

var RatingYears = from Movie in MovieLoader.AllMovies

group Movie by Movie.Rating into g

select g;

foreach (var Item in RatingYears)

{

foreach (var Movie in Item)

{

if (LowYear == 0 && HighYear == 0)

{

LowYear = Movie.Year;

HighYear = Movie.Year;

}

else if (LowYear == 0)

{

LowYear = Movie.Year;

}

else if (HighYear == 0)

{

HighYear = Movie.Year;

}

else if (Movie.Year > HighYear)

{

HighYear = Movie.Year;

}

else if (Movie.Year < LowYear)

{

LowYear = Movie.Year;

}

Distance = HighYear - LowYear;

if (Distance > RcrdDist)

{

RcrdDist = Distance;

LongBoi = Item.Key;

}

}

}

Console.WriteLine($"Rating: {LongBoi}, TimeSpan: {RcrdDist}");

PG13, 85 Years

1. How many films are in each decade of the total year range?

var Decades = from Movie in MovieLoader.AllMovies

group Movie by Movie.Year / 10 into g

orderby g.Key descending

select g;

foreach (var Item in Decades)

{

string MyString = Item.Key.ToString();

Console.WriteLine($"{MyString.Substring(2, 1) + 0}\'s: {Item.Count()}");

}

00’s: 1

90’s: 11

80’s: 8

70’s: 20

60’s: 17

50’s: 16

40’s: 11

30’s: 12

20’s: 3

10’s: 1

1. Which ratings in the MPAARating enum are never used?

bool UsedG = false, UsedNC17 = false, UsedNotRated = false, UsedPG = false, UsedPG13 = false, UsedR = false, UsedUnknown = false, UsedUnrated = false;

var UsedRatings = from Movie in MovieLoader.AllMovies

group Movie by Movie.Rating into g

select g;

foreach (var Item in UsedRatings)

{

//Console.WriteLine(Item.Key);

if (Item.Key == MPAARating.G && UsedG == false)

{

UsedG = true;

}

else if (Item.Key == MPAARating.NC17 && UsedNC17 == false)

{

UsedNC17 = true;

}

else if (Item.Key == MPAARating.NotRated && UsedNotRated == false)

{

UsedNotRated = true;

}

else if (Item.Key == MPAARating.PG && UsedPG == false)

{

UsedPG = true;

}

else if (Item.Key == MPAARating.PG13 && UsedPG13 == false)

{

UsedPG13 = true;

}

else if (Item.Key == MPAARating.R && UsedR == false)

{

UsedR = true;

}

else if (Item.Key == MPAARating.Unknown && UsedUnknown == false)

{

UsedUnknown = true;

}

else if (Item.Key == MPAARating.Unrated && UsedUnrated == false)

{

UsedUnrated = true;

}

}

if (UsedG != true)

{

Console.WriteLine("The G Rating is never used.");

}

if (UsedNC17 != true)

{

Console.WriteLine("The NC17 Rating is never used.");

}

if (UsedNotRated != true)

{

Console.WriteLine("The Not Rated Rating is never used.");

}

if (UsedPG != true)

{

Console.WriteLine("The PG Rating is never used.");

}

if (UsedPG13 != true)

{

Console.WriteLine("The PG13 Rating is never used.");

}

if (UsedR != true)

{

Console.WriteLine("The R Rating is never used.");

}

if (UsedUnknown != true)

{

Console.WriteLine("The Unknown Rating is never used.");

}

if (UsedUnrated != true)

{

Console.WriteLine("The Unrated Rating is never used.");

}

NC17, unknown, and Unrated are never used

1. Display all films, first by rating (lowest to highest enum value), then by title (alphabetically ascending).

var RatingList = from Movie in MovieLoader.AllMovies

orderby Movie.Rating ascending

select Movie;

var TitleList = from Movie in MovieLoader.AllMovies

orderby Movie.Title descending

select Movie;

Console.WriteLine("Movies Sorted According to Ratings, Descending");

foreach (var Movie in RatingList)

{

Console.WriteLine($"{Movie.Title}, {Movie.Rating}");

}

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("Movies Sorted According to Titles, Ascending");

foreach (var Movie in TitleList)

{

Console.WriteLine($"{Movie.Title}, {Movie.Rating}");

}

1. Group the movies by number of words in the title. List the results by group (meaning the number of words in the title) and the number of films in that group. Clearly label your results, please.

var TitleLength = from Movie in MovieLoader.AllMovies

group Movie by Movie.Title.Length into g

select g;

foreach (var ItemList in TitleLength)

{

Console.WriteLine($"Group: {ItemList.Key}, Films: {ItemList.Count()}");

}

Group: 12, Films: 8

Group: 13, Films: 5

Group: 10, Films: 6

Group: 11, Films: 6

Group: 19, Films: 4

Group: 18, Films: 8

Group: 16, Films: 6

Group: 7, Films: 9

Group: 9, Films: 8

Group: 6, Films: 1

Group: 21, Films: 7

Group: 17, Films: 3

Group: 26, Films: 1

Group: 28, Films: 2

Group: 14, Films: 2

Group: 31, Films: 3

Group: 27, Films: 1

Group: 32, Films: 1

Group: 15, Films: 6

Group: 22, Films: 1

Group: 5, Films: 2

Group: 24, Films: 4

Group: 49, Films: 1

Group: 4, Films: 1

Group: 23, Films: 2

Group: 34, Films: 1

Group: 20, Films: 1

1. My teenager is 15 and may watch any film that is rated G, PG, or PG-13. How many films from the master list could they watch? Which films are they?

var KidFriendly = from Movie in MovieLoader.AllMovies

where Movie.Rating == MPAARating.G || Movie.Rating == MPAARating.PG || Movie.Rating == MPAARating.PG13

orderby Movie.Rating ascending

select Movie;

Console.WriteLine($"You could watch {KidFriendly.Count()} Movies. They are: ");

foreach(var Movie in KidFriendly)

{

Console.WriteLine($"{Movie.Title}, Rating: {Movie.Rating}");

}

38 Movies Total,

SINGIN' IN THE RAIN, Rating: G

GONE WITH THE WIND, Rating: G

CITY LIGHTS, Rating: G

2001: A SPACE ODYSSEY, Rating: G

SNOW WHITE AND THE SEVEN DWARFS, Rating: G

THE SOUND OF MUSIC, Rating: G

MODERN TIMES, Rating: G

TOY STORY, Rating: G

BEN-HUR, Rating: G

CITIZEN KANE, Rating: PG

CASABLANCA, Rating: PG

LAWRENCE OF ARABIA, Rating: PG

VERTIGO, Rating: PG

THE WIZARD OF OZ, Rating: PG

STAR WARS, Rating: PG

THE GRADUATE, Rating: PG

IT'S A WONDERFUL LIFE, Rating: PG

E.T. THE EXTRA-TERRESTRIAL, Rating: PG

HIGH NOON, Rating: PG

ANNIE HALL, Rating: PG

THE BRIDGE ON THE RIVER KWAI, Rating: PG

DR. STRANGELOVE, Rating: PG

A STREETCAR NAMED DESIRE, Rating: PG

REAR WINDOW, Rating: PG

JAWS, Rating: PG

ROCKY, Rating: PG

AMERICAN GRAFFITI, Rating: PG

CABARET, Rating: PG

THE AFRICAN QUEEN, Rating: PG

RAIDERS OF THE LOST ARK, Rating: PG

TOOTSIE, Rating: PG

BUTCH CASSIDY AND THE SUNDANCE KID, Rating: PG

ALL THE PRESIDENT'S MEN, Rating: PG

THE LORD OF THE RINGS: THE FELLOWSHIP OF THE RING, Rating: PG13

FORREST GUMP, Rating: PG13

SPARTACUS, Rating: PG13

TITANIC, Rating: PG13

THE SIXTH SENSE, Rating: PG13

# What you’ll need

* LINQ
* Referencing external DLLs

# You should check out

* MSDN documentation on the Enumerable class

# Rubric

**Automatic 0** – You fail to use LINQ in each query, your answers are not readable, or your doc is not able to be graded for any reason.

You will be awarded 8.33 points per query you answer correctly.