Program Cover Sheet

|  |
| --- |
| Name: Austin Rippee |
| Assignment: Assignment 5 |
| List any parts of the assignment that do not work/were not completed:  Two things didn’t work in my project, Getting the average cost for a specific category and getting more than one title to display on the cheapest/priciest and least/most quantity shown (in the last part). The first one was a bit confusing on why it didn’t work. I attempted to try the same idea as in the notes using the aggregate function  Dim objFictionAverage = Aggregate books In myBooks Where books.strCategory = "F" Into Average(books.sngPrice)  but was getting multiple errors reading “Overload resolution failed because no accessible ‘Average’ can be called without a narrowing conversion” in which I attempted to fix using  Dim objFictionAverage = Aggregate books In myBooks Where books.strCategory = "F" Into Average(Of books.sngPrice, Single)  But no luck. The second problem I ran into was at the end where if the user had multiple titles where the price was at its lowest or highest and/or the quantity was at its highest/lowest then display all titles. I was only able to figure out how to get one. I attempted to try and use a way to get all by trying:  objCheapestPriceBooks = From books In myBooks  Where sngPrice = objCheapestPrice  Order By books.sngPrice  Select books.strTitle  But this was also a fail since I wasn’t able to get that object converted correctly into a single from objCheapestPrice. This goes for the rest of them. |

|  |
| --- |
| Instructor’s Comments: |
| Grade: |

Program Submission Requirements: (1) all files, zipped and uploaded to Canvas and (2) a completed cover sheet, program execution screenshots and source code printed, **stapled** and turned in during class. Failure to follow the submission requirements will result in points lost on that particular assignment.

Imports System.IO

'------------------------------------------------------------

'- File Name : Books\_R\_Us.vb -

'- Part of Project: Main -

'------------------------------------------------------------

'- Written By: Austin Rippee -

'- Written On: February 27th, 2022 -

'------------------------------------------------------------

'- File Purpose: -

'- This file contains the main application form where the -

'- user will input a path file in the format and it will

'- encode the file for various types of statistics

'------------------------------------------------------------

'- Program Purpose: -

'- -

'- This program allows for the user to enter a file path

'- where encodes the file line by line, character by character

'- and then displays many different statistics about the file

'------------------------------------------------------------

'- Global Variable Dictionary (alphabetically): -

'- intCounter – Keeps track of the number of items read in. –

'- sngGrandTotal – Total amount of sales dollars read in. –

'------------------------------------------------------------

Public Class clsBook

Public Property strCategory As String

Public Property intQuantity As Integer

Public Property sngPrice As Single

Public Property strTitle As String

Public Property sngInventoryTotal As Single

'------------------------------------------------------------

'- Subprogram Name: New (category, quantity, price, title, inventoryTotal) -

'------------------------------------------------------------

'- Written By: Austin Rippee -

'- Written On: February 27th, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- -

'- This subroutine is called whenever the user creates a new

'– instance of the object

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- category - strCategory as a string

'- quantity - intQuantity as an integer

'- price - sngPrice as a single

'- title - strTitle as a string

'- inventoryTotal - sngInventoryTotal as a single

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- (None) -

'------------------------------------------------------------

Public Sub New(ByVal category As String, ByVal quantity As Integer, ByVal price As Single, ByVal title As String, ByVal inventoryTotal As Single)

Me.strCategory = category

Me.intQuantity = quantity

Me.sngPrice = price

Me.strTitle = title

Me.sngInventoryTotal = inventoryTotal

End Sub

'------------------------------------------------------------

'- Function Name: ToString() -

'------------------------------------------------------------

'- Written By: Austin Rippee -

'- Written On: February 27th, 2022 -

'------------------------------------------------------------

'- Function Purpose: -

'- -

'- This function is called whenever the user displays the

'– object through an ienumerator and is provided using the

'- format/

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- (none)

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- (None) -

'------------------------------------------------------------

'- Returns: -

'- String.Format() – the book list with specific format -

'------------------------------------------------------------

Public Overrides Function ToString() As String

Return String.Format("{5} {3,-27} {0} {1,13} {2,15} {4,15}", strCategory, intQuantity, sngPrice.ToString("F2"), strTitle, sngInventoryTotal.ToString("F2"), " ")

End Function

End Class

'------------------------------------------------------------

'- Class Name: SortedList(Of T) -

'------------------------------------------------------------

'- Written By: Austin Rippee -

'- Written On: February 27th, 2022 -

'------------------------------------------------------------

'- Class Purpose: -

'- -

'- This class is designed create a new list whenever the user

'– wants to create a new list object. In this case, a new list

'- object of a book is called whenever the user wants to add

'- to this list object

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- T – The different datatype that is being passed through

' – for the different data type

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- (None) -

'------------------------------------------------------------

Public Class SortedList(Of T)

Implements IEnumerable

Private listItems As New List(Of T)

'------------------------------------------------------------

'- Subprogram Name: AddItem(AnItem) -

'------------------------------------------------------------

'- Written By: Austin Rippee -

'- Written On: February 27th, 2022 -

'------------------------------------------------------------

'- Subprogram Purpose: -

'- -

'- This subroutine is called whenever the user adds an item

'– to the list

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- AnItem – the value that will be added to the list –

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- (None) -

'------------------------------------------------------------

Public Sub AddItem(ByVal AnItem As T)

listItems.Add(AnItem)

End Sub

'Readonly property that allos for the user to get the count of the listitems

Public ReadOnly Property Count() As Integer

Get

'Returns the count

Return listItems.Count

End Get

End Property

'------------------------------------------------------------

'- Function Name: GetEnumerator() -

'------------------------------------------------------------

'- Written By: Austin Rippee -

'- Written On: February 27, 2022 -

'------------------------------------------------------------

'- Function Purpose: -

'- -

'- This function handles getting each instance in a for each

'– construct

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- sender – Identifies which particular control raised the –

'- click event -

'- e – Holds the EventArgs object sent to the routine -

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- (None) -

'------------------------------------------------------------

'- Returns: -

'- GetEnumerator() – gets the enumerator -

'------------------------------------------------------------

Public Function GetEnumerator() As IEnumerator Implements IEnumerable.GetEnumerator

'Returns the iterator to the underlying List class

Return listItems.GetEnumerator()

End Function

End Class

Module Books\_R\_Us

'------------------------------------------------------------

'- Module Name: Books\_R\_Us -

'------------------------------------------------------------

'- Written By: Austin Rippee -

'- Written On: February 27th -

'------------------------------------------------------------

'- Module Purpose: -

'- -

'- This subroutine is the main routine of the program in which

'– the user performs the normal functions of the program.

'------------------------------------------------------------

'- Parameter Dictionary (in parameter order): -

'- args – value of a string that is passing through –

'------------------------------------------------------------

'- Local Variable Dictionary (alphabetically): -

'- intColNum - the placeholder to keep track of where you are in the line

'- line - reader to read through the file line

'- myBooks - An object of clsBook

'- objCheapestPrice - Creates an object that stores the LINQ to find the cheapest price in the list

'- objCheapestPriceBooks - Creates an object that stores the LINQ to find the cheapest priced titles in the list

'- objFictionBooksCount - Creates an object that stores the LINQ to find the count of fiction books in the list

'- objFictionBooksMax - Creates an object that stores the LINQ to find the highest price fiction book in the list

'- objFictionBooksMin - Creates an object that stores the LINQ to find the lowest price fiction book in the list

'- objFrom50to100 - Creates an object that loops through all books where the price is above $50 and less than or equal to $100

'- objFrom100to150 - Creates an object that loops through all books where the price is above $100 and less than or equal to $150

'- objLeastQuantityBook - Creates an object that stores the LINQ to find the amount of times the least book the quantity has

'- objLeastQuantityBookTitles - Creates an object that stores the LINQ to find the titles of the least showed up book

'- objLessThan50 - 'Creates an object that loops through all books where the price is above $0 and less than or equal to $50

'- objMoreThan150 - Creates an object that loops through all books where the price is above $150

'- objMostQuantityBook - Creates an object that stores the LINQ to find the amount of times the most times the book the quantity has

'- objMostQuantityBookTitles - Creates an object that stores the LINQ to find the titles of the most showed up book

'- objNonFictionBooksCount - Creates an object that stores the LINQ to find the count of non fiction books in the list

'- objNonFictionBooksMax - Creates an object that stores the LINQ to find the highest price non fiction book in the list

'- objNonFictionBooksMin - Creates an object that stores the LINQ to find the lowest price non fiction book in the list

'- objPriciestBook - Creates an object that stores the LINQ to find the priciest price in the list

'- objPriciestBookTitles - Creates an object that stores the LINQ to find the priciest priced titles in the list

'- objSciFictionBooksCount - Creates an object that stores the LINQ to find the count of science fiction books in the list

'- objSciFictionBooksMax - Creates an object that stores the LINQ to find the highest price science fiction book in the list

'- objSciFictionBooksMin - Creates an object that stores the LINQ to find the lowest price science fiction book in the list

'- objSortedBooks - Creates an object that sotres the LINQ to sort the books in the list from the file

'- sngExtendedCost - Variable to perform the quantity \* price extendedcost

'- strCategory - holds the category from the characters saved using the mid method

'- strCategoryStatsTitle - Displays the subtitles for the category statistics

'- strChr - holds the character using the mid method

'- strFictionBooksCountString - creates the string to display the quantity of science fiction books in the list

'- strFictionBooksMaxString - Creates a string to display the max price of any of the fiction books in the list

'- strFictionBooksMinString - Creates a string to display the min price of any of the fiction books in the list

'- strFictionBooksString - String to print the line of fiction # of books, min, avg, max

'- strFrom50to100String - String to print the books that range from 50 to 100 dollars

'- strFrom100to150String - String to print the books that range from 100 to 150 dollars

'- strLess50String - String to print the books that range less than 50 dollars

'- strMoreThan150String - String to print the books that range from more than 150 dollars

'- strNonFictionBooksCountString - creates the string to display the quantity of non fiction books in the list

'- strNonFictionBooksMaxString - Creates a string to display the max price of any of the non fiction books in the list

'- strNonFictionBooksMinString - Creates a string to display the min price of any of the non fiction books in the list

'- strNonFictionBooksString - String to print the line of nonfiction # of books, min, avg, max

'- strPrice - holds the price from the characters saved using the mid method

'- strReportSeparators - string to print the report separators

'- strReportTitles - string to print the repor title

'- strSciFictionBooksCountString - creates the string to display the quantity of fiction books in the list

'- strSciFictionBooksMaxString - Creates a string to display the max price of any of the science fiction books in the list

'- strSciFictionBooksMinString - Creates a string to display the min price of any of the science fiction books in the list

'- strSciFictionBooksString - String to print the line of science fiction # of books, min, avg, max

'- strQuantity - holds the quantity from the characters saved using the mid method

'- strSourcePath - gets the source path from a user input

'- strTitle - holds the title of the books from the characters saved in that line

'- strTxtFileName - holds the file name of the source path

'------------------------------------------------------------

Sub Main(args As String())

Console.Title = "Books 'R' Us" 'Changes program title

Console.Clear() 'Clears the console

Console.WriteLine("Please enter the path and name of the file to process: ") 'Intial line asking for the file

'Sets the source path as what the user enters

Dim strSourcePath As String = Console.ReadLine()

'Gets the file name of the source path the user entered

Dim strTxtFileName As String = System.IO.Path.GetFileName(strSourcePath)

If System.IO.File.Exists(strSourcePath) Then

' Store the line in this String.

Dim line As String

'Creates a new list of clasBook that was created above

Dim myBooks As New SortedList(Of clsBook)

' Create new StreamReader instance with Using block.

Using reader As New StreamReader(strSourcePath)

Do Until reader.EndOfStream

line = reader.ReadLine()

'Initiates variables

Dim strCategory As String = ""

Dim strQuantity As String = ""

Dim strPrice As String = ""

Dim strTitle As String = ""

Dim strChr As String = ""

Dim intColNum As Integer

'Finds Category

For intColNum = 1 To line.Length

'Performs a Mid method to take the next character and uses it as the category

strChr = Mid$(line, intColNum, 1)

'Checks if the character has reached a space

If strChr = (" ") Then

Exit For

Else

'Adds the current letter to the category word

strCategory = strCategory & strChr

End If

Next

'Finds quantity

For intColNum = intColNum + 1 To line.Length

'Performs a Mid method to take the next characters until it hits a space and uses it as the quantity

strChr = Mid$(line, intColNum, 1)

'Checks if the character has reached a space

If strChr = (" ") Then

Exit For

Else

'Adds the current letter to the quantity word

strQuantity = strQuantity & strChr

End If

Next

'Finds Unit Price

For intColNum = intColNum + 1 To line.Length

'Performs a Mid method to take the next characters until it hits a space and uses it as the price

strChr = Mid$(line, intColNum, 1)

'Checks if the character has reached a space

If strChr = (" ") Then

Exit For

Else

'Adds the current letter to the price word

strPrice = strPrice & strChr

End If

Next

'Finds Title

strTitle = Mid$(line, intColNum + 1, line.Length) 'Takes the rest of the line and simply adds it as the title

'Variable to perform the quantity \* price extendedcost

Dim sngExtendedCost As Single = CSng(strQuantity) \* CSng(strPrice)

'Adds the individual book looping through each line in the text file

myBooks.AddItem(New clsBook(strCategory, CInt(strQuantity), CSng(strPrice), strTitle, Math.Round(sngExtendedCost, 2)))

Loop

End Using

'Prints out the title report

Console.WriteLine()

Console.WriteLine(vbTab & vbTab & vbTab & vbTab & "Books 'R' Us")

Console.WriteLine(vbTab & vbTab & vbTab & " \*\*\* Inventory Report \*\*\*")

Console.WriteLine(vbTab & vbTab & vbTab & "-----------------------------")

Console.WriteLine()

'Titles and separators for the report formatted

Dim strReportTitles As String = String.Format("{0,11} {1,24} {2,12} {3,12} {4,15}", "Title", "Category", "Quantity", "Unit Cost", "Extended Cost", " ")

Dim strReportSeparators As String = String.Format("{0,4} {1,11} {2,12} {3,12} {4,15}", "------------------------", "--------", "--------", "---------", "-------------", " ")

Console.WriteLine(strReportTitles)

Console.WriteLine(strReportSeparators)

'Sorts the list in alphabetical order

Dim objSortedBooks As Object

objSortedBooks = From books In myBooks

Order By books.strTitle

'Loops through every book in the list and displays them

For Each book In objSortedBooks

Console.WriteLine(book)

Next

'Displays the titles for the total inventory value

Console.WriteLine()

Console.WriteLine(StrDup(73, "-"))

Console.WriteLine(StrDup(11, " ") & "Total Inventory Value (Quantity \* Unit Price) Statistics")

Console.WriteLine(StrDup(73, "-"))

'Title for 0-50 range

Console.WriteLine("Those books in the range of 0.00 - 50.00 are:")

'Creates a LINQ that loops through all books where the price is above $0 and less than or equal to $50

Dim objLessThan50 As Object

objLessThan50 = From books In myBooks

Where books.sngInventoryTotal > 0 And books.sngInventoryTotal <= 50

Order By books.sngInventoryTotal

Select books

'Loops through all books in the query and then prints them out formatted

For Each book In objLessThan50

Dim strLess50String As String = String.Format("{2} {0,-30} Price: {1,0}", book.strTitle, Format(book.sngInventoryTotal, "Currency"), " ")

Console.WriteLine(strLess50String)

Next

Console.WriteLine()

'Title for 50-100 range

Console.WriteLine("Those books in the range of 50.00 - 100.00 are:")

'Creates a LINQ that loops through all books where the price is above $50 and less than or equal to $100

Dim objFrom50to100 As Object

objFrom50to100 = From books In myBooks

Where books.sngInventoryTotal > 50 And books.sngInventoryTotal <= 100

Order By books.sngInventoryTotal

Select books

'Displays the string format for the books within 50 to 100 and then displays them

For Each book In objFrom50to100

Dim strFrom50to100String As String = String.Format("{2} {0,-30} Price: {1,0}", book.strTitle, Format(book.sngInventoryTotal, "Currency"), " ")

Console.WriteLine(strFrom50to100String)

Next

Console.WriteLine()

'Title for 100-150 range

Console.WriteLine("Those books in the range of 100.00 - 150.00 are:")

'Creates a LINQ that loops through all books where the price is above $100 and less than or equal to $150

Dim objFrom100to150 As Object

objFrom100to150 = From books In myBooks

Where books.sngInventoryTotal > 100 And books.sngInventoryTotal <= 150

Order By books.sngInventoryTotal

Select books

'Displays the string format for the books within 100 to 150 and then displays them

For Each book In objFrom100to150

Dim strFrom100to150String As String = String.Format("{2} {0,-30} Price: {1,0}", book.strTitle, Format(book.sngInventoryTotal, "Currency"), " ")

Console.WriteLine(strFrom100to150String)

Next

Console.WriteLine()

'Title for 150 and above range

Console.WriteLine("Those books in the range of 150.00 and above are:")

'Creates a LINQ that loops through all books where the price is above $150

Dim objMoreThan150 As Object

objMoreThan150 = From books In myBooks

Where books.sngInventoryTotal > 150

Order By books.sngInventoryTotal

Select books

'Displays the string format for the books more than 150 and then displays them

For Each book In objMoreThan150

Dim strMoreThan150String As String = String.Format("{2} {0,-30} Price: {1,0}", book.strTitle, Format(book.sngInventoryTotal, "Currency"), " ")

Console.WriteLine(strMoreThan150String)

Next

Console.WriteLine()

Console.WriteLine()

'Displays the titles for the unit price range by category statistics

Console.WriteLine(StrDup(73, "-"))

Console.WriteLine(StrDup(11, " ") & "Unit Price Range by Category Statistics")

Console.WriteLine(StrDup(73, "-"))

'Displays the subtitles for the category statistics

Dim strCategoryStatsTitle As String = String.Format("{0} {1,15} {2,15} {3,15} {4,15}", "Category", "# of Titles", "Low", "Ave", "High")

Console.WriteLine(strCategoryStatsTitle)

'LINQ query to find the count for the total fiction books

Dim objFictionBooksCount As Object

objFictionBooksCount = From books In myBooks

Where books.strCategory = "F"

Group By books.strCategory Into Count

'LINQ querey to find the minimum price for a fiction book

Dim objFictionBooksMin As Object

objFictionBooksMin = From books In myBooks

Where books.strCategory = "F"

Order By books.sngPrice

Select books.sngPrice Take 1

'================================================================================================================================================================

'- Not sure why this code doesn't work. I looked at the notes and was able to get this but kept getting an

'- overload error and couldn't figure out what a good fix was.

'-

'-

'- Dim objFictionAverage = Aggregate books In myBooks Where books.strCategory = "F" Into Average(books.sngPrice)

'-

'-

'================================================================================================================================================================

'LINQ querey to find the maximum price for a fiction book

Dim objFictionBooksMax As Object

objFictionBooksMax = From books In myBooks

Where books.strCategory = "F"

Order By books.sngPrice Descending

Select books.sngPrice Take 1

Dim strFictionBooksCountString As String = ""

'For loop to loop through all books in the fictionbookscount to display them

For Each books In objFictionBooksCount

strFictionBooksCountString = String.Format(" {0,12}", books.count)

Next

Dim strFictionBooksMinString As String = ""

'For loop to loop through all books in fictionbooksmin to display them

For Each books In objFictionBooksMin

strFictionBooksMinString = String.Format(" {0,12}", books)

Next

Dim strFictionBooksMaxString As String = ""

'For loop to loop through all books in fictionbooksmax to display them

For Each books In objFictionBooksMax

strFictionBooksMaxString = String.Format(" {0,12}", books)

Next

'String format to print out the entire line for fiction books and its corresponding statistics

Dim strFictionBooksString As String = String.Format("{0,4} {1,15} {2,19} {3,15} {4,15}", "F", strFictionBooksCountString, Format(strFictionBooksMinString, "Currency"), "Avg?", Format(strFictionBooksMaxString, "Currency"))

'Displays it

Console.WriteLine(strFictionBooksString)

'LINQ query to find the count for the total nonfiction books

Dim objNonFictionBooksCount As Object

objNonFictionBooksCount = From books In myBooks

Where books.strCategory = "N"

Group By books.strCategory Into Count

'LINQ querey to find the minimum price for a nonfiction book

Dim objNonFictionBooksMin As Object

objNonFictionBooksMin = From books In myBooks

Where books.strCategory = "N"

Order By books.sngPrice

Select books.sngPrice Take 1

'================================================================================================================================================================

'- Not sure why this code doesn't work. I looked at the notes and was able to get this but kept getting an

'- overload error and couldn't figure out what a good fix was.

'-

'-

'- Dim objNonFictionAverage = Aggregate books In myBooks Where books.strCategory = "N" Into Average(books.sngPrice)

'-

'-

'================================================================================================================================================================

'LINQ querey to find the maximum price for a nonfiction book

Dim objNonFictionBooksMax As Object

objNonFictionBooksMax = From books In myBooks

Where books.strCategory = "N"

Order By books.sngPrice Descending

Select books.sngPrice Take 1

Dim strNonFictionBooksCountString As String = ""

'For loop to loop through all books in the nonfictionbookscount to display them

For Each books In objNonFictionBooksCount

strNonFictionBooksCountString = String.Format(" {0,12}", books.count)

Next

Dim strNonFictionBooksMinString As String = ""

'For loop to loop through all books in nonfictionbooksmin to display them

For Each books In objNonFictionBooksMin

strNonFictionBooksMinString = String.Format(" {0,12}", books)

Next

Dim strNonFictionBooksMaxString As String = ""

'For loop to loop through all books in nonfictionbooksmax to display them

For Each books In objNonFictionBooksMax

strNonFictionBooksMaxString = String.Format(" {0,12}", books)

Next

'String format to print out the entire line for nonfiction books and its corresponding statistics

Dim strNonFictionBooksString As String = String.Format("{0,4} {1,15} {2,19} {3,15} {4,15}", "N", strNonFictionBooksCountString, Format(strNonFictionBooksMinString, "Currency"), "Avg?", Format(strNonFictionBooksMaxString, "Currency"))

'Displays it

Console.WriteLine(strNonFictionBooksString)

'LINQ querey to find the count for a science fiction book

Dim objSciFictionBooksCount As Object

objSciFictionBooksCount = From books In myBooks

Where books.strCategory = "S"

Group By books.strCategory Into Count

'LINQ querey to find the minimum price for a science fiction book

Dim objSciFictionBooksMin As Object

objSciFictionBooksMin = From books In myBooks

Where books.strCategory = "S"

Order By books.sngPrice

Select books.sngPrice Take 1

'================================================================================================================================================================

'- Not sure why this code doesn't work. I looked at the notes and was able to get this but kept getting an

'- overload error and couldn't figure out what a good fix was.

'-

'-

'- Dim objSciFictionAverage = Aggregate books In myBooks Where books.strCategory = "S" Into Average(books.sngPrice)

'-

'-

'================================================================================================================================================================

'LINQ querey to find the maximum price for a science fiction book

Dim objSciFictionBooksMax As Object

objSciFictionBooksMax = From books In myBooks

Where books.strCategory = "S"

Order By books.sngPrice Descending

Select books.sngPrice Take 1

Dim strSciFictionBooksCountString As String = ""

'For loop to loop through all books in scifictionbookscount to display them

For Each books In objSciFictionBooksCount

strSciFictionBooksCountString = String.Format(" {0,12}", books.count)

Next

Dim strSciFictionBooksMinString As String = ""

'For loop to loop through all books in scifictionbooksmin to display them

For Each books In objSciFictionBooksMin

strSciFictionBooksMinString = String.Format(" {0,12}", books)

Next

Dim strSciFictionBooksMaxString As String = ""

'For loop to loop through all books in scifictionbooksmax to display them

For Each books In objSciFictionBooksMax

strSciFictionBooksMaxString = String.Format(" {0,12}", books)

Next

'String format to print out the entire line for science fiction books and its corresponding statistics

Dim strSciFictionBooksString As String = String.Format("{0,4} {1,15} {2,19} {3,15} {4,15}", "S", strSciFictionBooksCountString, Format(strSciFictionBooksMinString, "Currency"), "Avg?", Format(strSciFictionBooksMaxString, "Currency"))

'Displays it

Console.WriteLine(strSciFictionBooksString)

'Displays the titles for the overall book statistics

Console.WriteLine()

Console.WriteLine(StrDup(73, "-"))

Console.WriteLine(StrDup(16, " ") & "Overall Books Statistics")

Console.WriteLine(StrDup(73, "-"))

'LINQ query to find the cheapest price of any book

Dim objCheapestPrice As Object

objCheapestPrice = From books In myBooks

Order By books.sngPrice

Select books.sngPrice Take 1

'Loops through all books in the cheapest price and displays them

For Each books In objCheapestPrice

Console.WriteLine("The cheapest book title(s) at a unit price of " & Format(CStr(books), "Currency") & " are: ")

Next

'LINQ query to find the titles of the cheapest price of any book

Dim objCheapestPriceBooks As Object

objCheapestPriceBooks = From books In myBooks

Order By books.sngPrice

Select books.strTitle Take 1

'Loops through all books of the cheapest price books and displays the titles

For Each books In objCheapestPriceBooks

Console.WriteLine(" {0,6}", books)

Next

Console.WriteLine()

'Loops through all books in the priciest price and displays them

Dim objPriciestBook As Object

objPriciestBook = From books In myBooks

Order By books.sngPrice Descending

Select books.sngPrice Take 1

'Loops through all books in the priciest range and displays them

For Each books In objPriciestBook

Console.WriteLine("The priciest book title(s) at a unit price of " & Format(CStr(books), "Currency") & " are: ")

Next

'LINQ query to find the titles of the priciest price of any book

Dim objPriciestBookTitles As Object

objPriciestBookTitles = From books In myBooks

Order By books.sngPrice Descending

Select books.strTitle Take 1

'Loops through all books of the priciest price books and displays the titles

For Each books In objPriciestBookTitles

Console.WriteLine(" {0,6}", books)

Next

Console.WriteLine()

'LINQ query to find the least quantity of a book that appears in the list

Dim objLeastQuantityBook As Object

objLeastQuantityBook = From books In myBooks

Order By books.intQuantity

Select books.intQuantity Take 1

'Loops through the least quantity books and displays it

For Each books In objLeastQuantityBook

Console.WriteLine("The title with the least quantity on hand at " & CStr(books) & " units are: ")

Next

'LINQ query to find the least quantity titles of a book that appears in the list

Dim objLeastQuantityBookTitles As Object

objLeastQuantityBookTitles = From books In myBooks

Order By books.intQuantity

Select books.strTitle Take 1

'Loops through the least quantity book titles and displays them

For Each books In objLeastQuantityBookTitles

Console.WriteLine(" {0,6}", books)

Next

Console.WriteLine()

'LINQ query to find the most quantity of a book that appears in the list

Dim objMostQuantityBook As Object

objMostQuantityBook = From books In myBooks

Order By books.intQuantity Descending

Select books.intQuantity Take 1

'Loops through the most quantity books and siaplys the titles

For Each books In objMostQuantityBook

Console.WriteLine("The title with the most quantity on hand at " & CStr(books) & " units are: ")

Next

'LINQ query to find the most quantity titles of a book that appears in the list

Dim objMostQuantityBookTitles As Object

objMostQuantityBookTitles = From books In myBooks

Order By books.intQuantity Descending

Select books.strTitle Take 1

'Loops through the most quantity book titles and displays them

For Each books In objMostQuantityBookTitles

Console.WriteLine(" {0,6}", books)

Next

Console.WriteLine()

Else

'Displays if the user does not enter a valid path name

Console.WriteLine("File does not exist. Exit the program and try again.")

End If

Console.ReadLine()

End Sub

End Module

Text

Description automatically generatedText

Description automatically generated