

```
1  /*-----
2  * Name:      Dan Cassidy
3  * Date:      2015-06-09
4  * Assignment: cView-P2
5  * Source File: Program.cs
6  * Course:    CSCI-C 490, C# Programming, MoWe 08:00
7  * Project:   The overall goal of this project is to capitalize on the fact that government, from
8  *            local to national, has made some of its data open by developing a way to explore
9  *            this data and present it to a user in a meaningful fashion. This phase of the
10 *            project is meant to explore a subset of fields in a Public Facility dataset as
11 *            represented in the dataset at https://data.southbend.in.gov/d/jeef-dsq9.
12 * Purpose:   Small wrapper program for demonstrating the CViewDataInteractive class.
13 -----*/
14
15 using System;
16 using System.Collections.Generic;
17 using System.Linq;
18 using System.Text;
19 using System.Threading.Tasks;
20 using CView;
21
22 namespace cView_P2_DanCassidy_Console
23 {
24     class Program
25     {
26         /*-----
27         * Method:   Main
28         * Purpose:  Serves as the entry point to the program.
29         * Input:    String array object representing any command line arguments. Ignored.
30         * Output:   Nothing.
31         -----*/
32         static void Main(string[] args)
33         {
34             //Declare a new CViewDataInteractive object.
35             var data = new CViewDataInteractive();
36
37             //Interactively manipulate said object.
38             data.InteractiveManipulation();
39         }
40     }
41 }
42
```

```

1  /*-----
2  * Name:      Dan Cassidy
3  * Date:      2015-06-09
4  * Assignment: cView-P2
5  * Source File: CViewDataInteractive.cs
6  * Course:    CSCI-C 490, C# Programming, MoWe 08:00
7  * Purpose:   Provides interactive management of a CViewDataSet object.
8  *-----*/
9
10 using System;
11 using System.Collections.Generic;
12 using System.Linq;
13 using System.Text;
14 using System.Threading.Tasks;
15
16 namespace CView
17 {
18     class CViewDataInteractive
19     {
20         //Helper constants for menu validation.
21         private const mainMenu MAINMENU_MIN = mainMenu.ADD;
22         private const mainMenu MAINMENU_MAX = mainMenu.EXIT;
23         private const fieldMenu FIELDMENU_MIN = fieldMenu.NAME;
24         private const fieldMenu FIELDMENU_MAX = fieldMenu.BACK;
25
26         //Primary class field/instance variable.
27         private CViewDataSet data = new CViewDataSet();
28
29         //Enum for the main menu. Basic code idea from Stack Overflow.
30         //http://stackoverflow.com/a/15752719
31         private enum mainMenu
32         {
33             ADD = 1,
34             MODIFY,
35             SEARCH,
36             DELETE,
37             DISPLAY_ALL,
38             EXIT
39         }
40
41         //Enum for the modify menu. Basic code idea from Stack Overflow.
42         //http://stackoverflow.com/a/15752719
43         private enum fieldMenu
44         {
45             NAME = CViewData.FIELDS_MIN,
46             FACILITY,
47             ADDRESS,
48             CITY,
49             PHONE,
50             BACK
51         }
52
53         /*-----
54         * Method: InteractiveManipulation
55         * Purpose: Entry point for interactive manipulation of CViewDataSet object.
56         * Input:   Nothing.
57         * Output:  Nothing.
58         *-----*/
59         public void InteractiveManipulation()
60         {
61             //Loop the main menu until the user decides to exit.
62             while (MainMenuAction(MainMenuDisplay()) != mainMenu.EXIT) ;
63         }
64

```

```

65      /*-----
66      * Method:   DataAdd
67      * Purpose: Interactively add an item based on the user's input.
68      * Input:    Nothing.
69      * Output:   Nothing.
70      -----*/
71  private void DataAdd()
72  {
73      //New CViewData object that will be added to the dataset.
74      CViewData dataToAdd = new CViewData();
75
76      //Prompt the user to input information about the new item.
77      Console.WriteLine("-----");
78      Console.WriteLine("| Add New Item |");
79      Console.WriteLine("-----");
80      Console.Write("Facility Name: ");
81      dataToAdd.Name = Console.ReadLine();
82      Console.Write("Facility Type: ");
83      dataToAdd.FacilityType = Console.ReadLine();
84      Console.Write("Address: ");
85      dataToAdd.Address = Console.ReadLine();
86      Console.Write("City: ");
87      dataToAdd.City = Console.ReadLine();
88      Console.Write("Phone Number: ");
89      dataToAdd.PhoneNumber = Console.ReadLine();
90
91      //Extra line for formatting.
92      Console.WriteLine();
93
94      //Add the new item to the main data set.
95      data.Add(dataToAdd);
96
97      //Sort the data set.
98      data.SortByName();
99  }
100
101  /*-----
102  * Method:   DataDelete
103  * Purpose: Interactively deletes an object based upon user input.
104  * Input:    Nothing.
105  * Output:   Nothing.
106  -----*/
107  private void DataDelete()
108  {
109      //Default value of 0 in case the user doesn't enter a choice and just hits 'enter'.
110      int indexToDelete = 0;
111
112      //Display the user's choice.
113      Console.WriteLine("-----");
114      Console.WriteLine("| Delete Item -- Existing Items |");
115      Console.WriteLine("-----");
116
117      //Display a numbered list of all the objects in the data set.
118      DataDisplayAllNumbered();
119
120      //Get the user's choice of which object to delete.
121      Console.Write("\nSelect item (0 to cancel): ");
122      int.TryParse(Console.ReadLine(), out indexToDelete);
123      indexToDelete--;
124
125      //Extra line for formatting.
126      Console.WriteLine();
127
128      //Display the results.

```

```

129         Console.WriteLine("-----");
130         Console.WriteLine(" | Delete Item -- Results |");
131         Console.WriteLine("-----");
132
133         //Validate the user's choice.
134         if (indexToDelete == -1)
135         {
136             //The user changed their mind.
137             Console.WriteLine("Cancelled.\n");
138             return;
139         }
140         else if (indexToDelete < 0 || indexToDelete >= data.Count)
141         {
142             //The user input an invalid object index.
143             Console.WriteLine("Invalid item.\n");
144             return;
145         }
146
147         //Delete the object and display confirmation of its deletion.
148         data.Delete(indexToDelete);
149         Console.WriteLine("Item {0} has been deleted.\n", indexToDelete + 1);
150
151         //Display the still existing items.
152         DataDisplayAll(false);
153     }
154
155     /*-----
156     * Method:   DataDisplayAll
157     * Purpose:  Displays the header and the serialized dataset object.
158     * Input:    bool displayTitle, determines whether the method should print a title showing
159     *           that this method was the one that was called.
160     * Output:   Nothing.
161     -----*/
162     private void DataDisplayAll(bool displayTitle = true)
163     {
164         //Choose whether to display the title.
165         if (displayTitle)
166         {
167             //Display the user's choice.
168             Console.WriteLine("-----");
169             Console.WriteLine(" | Display All Items |");
170             Console.WriteLine("-----");
171         }
172
173         //Display all the objects.
174         Console.WriteLine("{0}\n{1}", data.Header, data.Count != 0 ?
175             data.ToString() : "No items currently stored.\n");
176     }
177
178     /*-----
179     * Method:   DataDisplayAllNumbered
180     * Purpose:  Display a header and a numbered list of objects.
181     * Input:    Nothing.
182     * Output:   Nothing.
183     -----*/
184     private void DataDisplayAllNumbered()
185     {
186         //Display the header.
187         Console.WriteLine("Item {0}", data.Header);
188
189         //If the dataset is not empty.
190         if (data.Count != 0)
191             //Display the numbered objects, starting at 1.
192             for (int objectNum = 0; objectNum < data.Count; objectNum++)

```

```

193         Console.WriteLine("{0,4} {1}", objectNum + 1, data[objectNum]);
194     else
195         //Display a message saying that dataset is empty.
196         Console.WriteLine("No items currently stored.");
197 }
198
199 /*-----
200  * Method:   DataModify
201  * Purpose:  Interactively modifies an object based on the user's input.
202  * Input:    Nothing.
203  * Output:   Nothing.
204  -----*/
205 private void DataModify()
206 {
207     //Default value of 0 in case the user doesn't enter a choice and just hits 'enter'.
208     int indexToModify = 0;
209
210     //Display the user's choice.
211     Console.WriteLine("-----");
212     Console.WriteLine("| Modify Item -- Existing Items |");
213     Console.WriteLine("-----");
214
215     //Display a numbered list of all the objects in the data set.
216     DataDisplayAllNumbered();
217
218     //Get the user's choice of which object to delete.
219     Console.Write("\nSelect item (0 to cancel): ");
220     int.TryParse(Console.ReadLine(), out indexToModify);
221     indexToModify--;
222
223     //Extra line for formatting.
224     Console.WriteLine();
225
226     //Validate the user's choice.
227     if (indexToModify == -1)
228     {
229         //The user changed their mind.
230         Console.WriteLine("Cancelled.\n");
231         return;
232     }
233     else if (indexToModify < 0 || indexToModify >= data.Count)
234     {
235         //The user input an invalid object index.
236         Console.WriteLine("Invalid item.\n");
237         return;
238     }
239
240     do
241     {
242         //Display the chosen object.
243         Console.WriteLine("-----");
244         Console.WriteLine("| Modify Item -- Chosen Item |");
245         Console.WriteLine("-----");
246         Console.WriteLine("{0}\n{1}\n", data.Header, data[indexToModify]);
247
248         //Loop while the use has not chosen to go back.
249     } while (DataModifyMenuAction(FieldMenuDisplay(), indexToModify) != fieldMenu.BACK);
250 }
251
252 /*-----
253  * Method:   DataModifyMenuAction
254  * Purpose:  Acts on the user's choice made at the Modify Menu.
255  * Input:    fieldMenu choice, represents the action specified.
256  * Output:   fieldMenu, represents the action specified.

```

```
257 -----*/
258 private fieldMenu DataModifyMenuAction(fieldMenu choice, int indexToModify)
259 {
260     //Decide what to do based on the user's choice.
261     switch (choice)
262     {
263         case fieldMenu.NAME:
264             //Change the name of the item.
265             Console.WriteLine("Current Facility Name: {0}", data[indexToModify].Name);
266             Console.Write("New Facility Name: ");
267             data[indexToModify].Name = Console.ReadLine();
268
269             //Sort the data set after changing the name since name is the sort criteria.
270             data.SortByName();
271
272             break;
273
274         case fieldMenu.FACILITY:
275             //Change the facility type of the item.
276             Console.WriteLine("Current Facility Type: {0}",
277                             data[indexToModify].FacilityType);
278             Console.Write("New Facility Type: ");
279             data[indexToModify].FacilityType = Console.ReadLine();
280
281             break;
282
283         case fieldMenu.ADDRESS:
284             //Change the address of the item.
285             Console.WriteLine("Current Address: {0}", data[indexToModify].Address);
286             Console.Write("New Address: ");
287             data[indexToModify].Address = Console.ReadLine();
288
289             break;
290
291         case fieldMenu.CITY:
292             //Change the city of the item.
293             Console.WriteLine("Current City: {0}", data[indexToModify].City);
294             Console.Write("New City: ");
295             data[indexToModify].City = Console.ReadLine();
296
297             break;
298
299         case fieldMenu.PHONE:
300             //Change the phone number of the item.
301             Console.WriteLine("Current Phone Number: {0}", data[indexToModify].PhoneNumber);
302             Console.Write("New Phone Number: ");
303             data[indexToModify].PhoneNumber = Console.ReadLine();
304
305             break;
306
307         case fieldMenu.BACK:
308             //Nothing to do; the user wants to go back.
309         default:
310             //Catch-all.
311             return choice;
312     }
313
314     //Extra line for formatting.
315     Console.WriteLine();
316
317     //Return choice so the calling method knows what the choice was and can act accordingly.
318     return choice;
319 }
320
```

```

321     /*-----
322     * Method:   DataSearch
323     * Purpose:  Interactively searches for objects based upon user input.
324     * Input:    Nothing.
325     * Output:   Nothing.
326     -----*/
327     private void DataSearch()
328     {
329         do
330         {
331             //Display the user's choice.
332             Console.WriteLine("-----");
333             Console.WriteLine("| Search Items |");
334             Console.WriteLine("-----");
335
336             //Loop while the use has not chosen to go back.
337         } while (DataSearchMenuAction(FieldMenuDisplay()) != fieldMenu.BACK);
338     }
339
340     /*-----
341     * Method:   DataSearchMenuAction
342     * Purpose:  Acts on the user's choice made at the Search Menu.
343     * Input:    fieldMenu choice, represents the action specified.
344     * Output:   fieldMenu, represents the action specified.
345     -----*/
346     private fieldMenu DataSearchMenuAction(fieldMenu choice)
347     {
348         //Decide what to display based on the user's choice.
349         switch (choice)
350         {
351             case fieldMenu.NAME:
352                 //Search the name field.
353                 Console.WriteLine("-----");
354                 Console.WriteLine("| Search Items -- Facility Name |");
355                 Console.WriteLine("-----");
356                 break;
357
358             case fieldMenu.FACILITY:
359                 //Search the facility type field.
360                 Console.WriteLine("-----");
361                 Console.WriteLine("| Search Items -- Facility Type |");
362                 Console.WriteLine("-----");
363                 break;
364
365             case fieldMenu.ADDRESS:
366                 //Search the address field.
367                 Console.WriteLine("-----");
368                 Console.WriteLine("| Search Items -- Address |");
369                 Console.WriteLine("-----");
370                 break;
371
372             case fieldMenu.CITY:
373                 //Search the city field.
374                 Console.WriteLine("-----");
375                 Console.WriteLine("| Search Items -- City |");
376                 Console.WriteLine("-----");
377                 break;
378
379             case fieldMenu.PHONE:
380                 //Search the phone number field.
381                 Console.WriteLine("-----");
382                 Console.WriteLine("| Search Items -- Phone Number |");
383                 Console.WriteLine("-----");
384                 break;

```

```

385
386         case fieldMenu.BACK:
387             //Nothing to do; the user wants to go back.
388         default:
389             //Catch-all.
390             return choice;
391     }
392
393     //Ask for the search text.
394     Console.Write("Enter your search text: ");
395
396     //Get the user's search text and pipe that directly into the search method.
397     CViewDataSet foundData = data.Search(Console.ReadLine(), (CViewData.Fields)choice);
398
399     //Show the number of items found.
400     Console.WriteLine("{0} item{1} found.\n", foundData.Count,
401         foundData.Count == 1 ? "" : "s");
402
403     //If any items found, display them.
404     if (foundData.Count != 0)
405         Console.WriteLine("{0}\n{1}", foundData.Header, foundData);
406
407     //Return choice so the calling method knows what the choice was and can act accordingly.
408     return choice;
409 }
410
411 /*-----
412  * Method:   FieldMenuDisplay
413  * Purpose:  Display the field menu and get a choice. Must have valid input to return.
414  * Input:    Nothing.
415  * Output:   fieldMenu, representing the choice that was made.
416  -----*/
417 private fieldMenu FieldMenuDisplay()
418 {
419     fieldMenu menuChoice = 0;
420     bool invalid = true;
421
422     do
423     {
424         //Display the menu.
425         Console.WriteLine("Please select the field you would like to work with:");
426         Console.WriteLine(" 1) Facility Name");
427         Console.WriteLine(" 2) Facility Type");
428         Console.WriteLine(" 3) Street Address");
429         Console.WriteLine(" 4) City");
430         Console.WriteLine(" 5) Phone Number");
431         Console.WriteLine(" 6) Back");
432         Console.Write("Choice: ");
433
434         //Get the user's choice.
435         string input = Console.ReadLine();
436
437         //Extra line for formatting.
438         Console.WriteLine();
439
440         //Validate the user input.
441         invalid = !fieldMenu.TryParse(input, out menuChoice) ||
442             !FieldMenuValidate(menuChoice);
443     } while (invalid);
444
445     //Return the user's choice.
446     return menuChoice;
447 }
448

```



```
449      /*-----
450      * Method: FieldMenuValidate
451      * Purpose: Validates that the choice by the user is within the limits and is logically
452      *           possible.
453      * Input:   mmodifyMenu value, contains the user's choice.
454      * Output:  bool, representing whether the user's choice was valid or not.
455      -----*/
456      private bool FieldMenuValidate(fieldMenu value)
457      {
458          //Check to make sure that the user input is within valid limits.
459          if (value < FIELDMENU_MIN || value > FIELDMENU_MAX)
460              return false;
461
462          //Otherwise, input is good.
463          return true;
464      }
465
466      /*-----
467      * Method: MainMenuAction
468      * Purpose: Acts on the user's choice made at the Main Menu.
469      * Input:   mainMenu choice, represents the action specified.
470      * Output:  mainMenu, represents the action specified.
471      -----*/
472      private mainMenu MainMenuAction(mainMenu choice)
473      {
474          //Decide what to do based on the user's choice.
475          switch (choice)
476          {
477              case mainMenu.ADD:
478                  //Add a new item.
479                  DataAdd();
480                  break;
481
482              case mainMenu.MODIFY:
483                  //Modify an existing item.
484                  DataModify();
485                  break;
486
487              case mainMenu.SEARCH:
488                  //Search items.
489                  DataSearch();
490                  break;
491
492              case mainMenu.DELETE:
493                  //Delete an item.
494                  DataDelete();
495                  break;
496
497              case mainMenu.DISPLAY_ALL:
498                  //Display all the items.
499                  DataDisplayAll();
500                  break;
501
502              case mainMenu.EXIT:
503                  //Do nothing, exiting the method.
504              default:
505                  //Catch-all.
506                  break;
507          }
508
509          //Return choice so the calling method knows what the choice was and can act accordingly.
510          return choice;
511      }
512
```

```

513     /*-----
514     * Method:   MainMenuDisplay
515     * Purpose:  Display the main menu and get a choice. Must have valid input to return.
516     * Input:    Nothing.
517     * Output:   mainMenu, representing the choice that was made.
518     -----*/
519     private mainMenu MainMenuDisplay()
520     {
521         mainMenu menuChoice = 0;
522         bool invalid = true;
523
524         do
525         {
526             //Display the menu.
527             Console.WriteLine("-----");
528             Console.WriteLine("| Main Interactive Menu |");
529             Console.WriteLine("-----");
530             Console.WriteLine("Please select an option:");
531             Console.WriteLine(" 1) Add New Item");
532             Console.WriteLine(" 2) Modify Item");
533             Console.WriteLine(" 3) Search Items");
534             Console.WriteLine(" 4) Delete Item");
535             Console.WriteLine(" 5) Display All Items");
536             Console.WriteLine(" 6) Exit");
537             Console.Write("Choice: ");
538
539             //Get the user's choice.
540             string input = Console.ReadLine();
541
542             //Extra line for formatting.
543             Console.WriteLine();
544
545             //Validate the user input.
546             invalid = !mainMenu.TryParse(input, out menuChoice) ||
547                     !MainMenuValidate(menuChoice);
548         } while (invalid);
549
550         //Return the user's choice.
551         return menuChoice;
552     }
553
554     /*-----
555     * Method:   MainMenuValidate
556     * Purpose:  Validates that the choice by the user is within the limits and is logically
557     *           possible.
558     * Input:    mainMenu value, contains the user's choice.
559     * Output:   bool, representing whether the user's choice was valid or not.
560     -----*/
561     private bool MainMenuValidate(mainMenu value)
562     {
563         //Check to make sure that the user input is within valid limits.
564         if (value < MAINMENU_MIN || value > MAINMENU_MAX)
565             return false;
566
567         //If the data set is empty, limit user to adding an entry or exiting.
568         if (data.Count == 0 && (value != mainMenu.ADD && value != mainMenu.EXIT))
569         {
570             Console.WriteLine("No data is present. Please choose a different option.\n");
571             return false;
572         }
573
574         //Otherwise, input is good.
575         return true;
576     }

```

```
577     }  
578 }  
579
```

```

1  /*-----
2  * Name:      Dan Cassidy
3  * Date:      2015-06-09
4  * Assignment: cView-P2
5  * Source File: CViewDataSet.cs
6  * Course:    CSCI-C 490, C# Programming, MoWe 08:00
7  * Purpose:   Encapsulates a List-based collection of CViewData objects and contains related
8  *            methods and properties.
9  -----*/
10
11 using System;
12 using System.Collections.Generic;
13 using System.Linq;
14 using System.Text;
15 using System.Threading.Tasks;
16
17 namespace CView
18 {
19     class CViewDataSet
20     {
21         //Basic field of the class.
22         private List<CViewData> dataSet = new List<CViewData>();
23
24         //Enable read-only access to the Count property.
25         public int Count
26         {
27             get
28             {
29                 return dataSet.Count;
30             }
31         }
32
33         //Enable read-only access to the Header property. Uses the header from the CViewData class
34         //so if needs to be changed, it only needs to be changed in one place.
35         public string Header
36         {
37             get
38             {
39                 return CViewData.HEADER;
40             }
41         }
42
43         /*-----
44         * Method:  this[]
45         * Purpose: Access the objects in this dataset via index number.
46         * Input:   int objectNum, the index of the object that will be accessed.
47         * Output:  CViewData object of the referenced object at the index.
48         -----*/
49         public CViewData this[int objectNum]
50         {
51             get
52             {
53                 //Try to simply return the object at index objectNum.
54                 try
55                 {
56                     return dataSet[objectNum];
57                 }
58                 catch (ArgumentOutOfRangeException)
59                 {
60                     //If this exception is caught, let the user know and return a null.
61                     Console.WriteLine("Index [{0}] is out of range.", objectNum);
62                     return null;
63                 }
64             }
65         }
66     }
67 }

```

```

65         set
66         {
67             //Try to set the object at index objectNum.
68             try
69             {
70                 dataSet[objectNum] = value;
71             }
72             catch (ArgumentOutOfRangeException)
73             {
74                 //If this exception is caught, do nothing further and let the user know.
75                 Console.WriteLine("Index [{0}] is out of range.", objectNum);
76             }
77         }
78     }
79
80     /*-----
81     * Method:   Add
82     * Purpose:  Add a data object to the dataset.
83     * Input:    CViewData toAdd, this is the object that will get added to the dataset.
84     * Output:   Nothing.
85     -----*/
86     public void Add(CViewData toAdd)
87     {
88         //Add object using List Add method.
89         dataSet.Add(toAdd);
90     }
91
92     /*-----
93     * Method:   Delete
94     * Purpose:  Delete an object at the given index from the dataset.
95     * Input:    int indexToRemove, the index of the object to be removed from the dataset.
96     * Output:   Nothing.
97     -----*/
98     public void Delete(int indexToRemove)
99     {
100         //Delete object at specified index by using List RemoveAt method.
101         dataSet.RemoveAt(indexToRemove);
102     }
103
104     /*-----
105     * Method:   Search
106     * Purpose:  Search for a given string in this dataset.
107     * Input:    string toSearchFor, this is the string that will be searched for.
108     * Input:    CViewData.Fields searchField, this is the field that will be searched.
109     * Output:   CViewDataSet object, containing all (if any) objects found.
110     -----*/
111     public CViewDataSet Search(string toSearchFor, CViewData.Fields searchField)
112     {
113         //Shortened form of StringComparison.OrdinalIgnoreCase for code prettiness.
114         var ignoreCase = StringComparison.OrdinalIgnoreCase;
115
116         //Use LINQ to search the objects with case insensitivity. Basic case insensitivity code
117         //idea from Stack Overflow. http://stackoverflow.com/a/444818
118         var foundData =
119             from data in dataSet
120             where
121                 //Search Name property.
122                 (searchField == CViewData.Fields.Name &&
123                  data.Name.IndexOf(toSearchFor, ignoreCase) >= 0) ||
124                 //Search FacilityType property.
125                 (searchField == CViewData.Fields.FacilityType &&
126                  data.FacilityType.IndexOf(toSearchFor, ignoreCase) >= 0) ||
127                 //Search Address property.
128                 (searchField == CViewData.Fields.Address &&

```

```
129         data.Address.IndexOf(toSearchFor, ignoreCase) >= 0) ||
130         //Search City property.
131         (searchField == CViewData.Fields.City &&
132         data.City.IndexOf(toSearchFor, ignoreCase) >= 0) ||
133         //Search PhoneNumber property.
134         (searchField == CViewData.Fields.PhoneNumber &&
135         data.PhoneNumber.IndexOf(toSearchFor, ignoreCase) >= 0)
136     select data;
137
138     //Return a new dataset containing the found objects.
139     return new CViewDataSet() { dataSet = foundData.ToList() };
140 }
141
142 /*-----
143  * Method:   SortByName
144  * Purpose:  Sort the dataset by the Name property of the objects.
145  * Input:    Nothing.
146  * Output:   Nothing.
147  -----*/
148 public void SortByName()
149 {
150     //Idea from Stack Overflow: http://stackoverflow.com/a/3309230
151     //Yay lambda expressions!
152     dataSet = dataSet.OrderBy(data => data.Name).ToList();
153 }
154
155 /*-----
156  * Method:   ToString
157  * Purpose:  Override of the ToString() method. Formats the return value so it looks pretty.
158  * Input:    Nothing.
159  * Output:   String object containing serialized collection data.
160  -----*/
161 public override string ToString()
162 {
163     //Declare the string.
164     string toReturn = "";
165
166     //Build the string.
167     foreach (var item in dataSet)
168         toReturn += item.ToString() + "\n";
169
170     //Return the string.
171     return toReturn;
172 }
173 }
174 }
175
```

```
1  /*-----
2  * Name:      Dan Cassidy
3  * Date:      2015-06-09
4  * Assignment: cView-P2
5  * Source File: CViewData.cs
6  * Course:    CSCI-C 490, C# Programming, MoWe 08:00
7  * Purpose:   Contains the basic data class for the cView program, along with some supporting
8  *            methods.
9  *-----*/
10
11 using System;
12 using System.Collections.Generic;
13 using System.Linq;
14 using System.Text;
15 using System.Threading.Tasks;
16
17 namespace CView
18 {
19     class CViewData
20     {
21         //Exposes the min and max fields.
22         public const Fields FIELDS_MIN = Fields.Name;
23         public const Fields FIELDS_MAX = Fields.PhoneNumber;
24
25         //Easily accessible string showing the data order in the ToString() method.
26         public const string HEADER = "Facility Name (Type), Address, City [Phone Number]";
27
28         //Represents the fields in use in this class. In lieu of inheritance and such, this is used
29         //to help facilitate searching (versus using int literals).
30         public enum Fields
31         {
32             Name = 1,
33             FacilityType,
34             Address,
35             City,
36             PhoneNumber
37         }
38
39         //Basic properties of the class.
40         public string Name { get; set; }
41         public string FacilityType { get; set; }
42         public string Address { get; set; }
43         public string City { get; set; }
44         public string PhoneNumber { get; set; }
45
46         /*-----
47         * Method: ToString
48         * Purpose: Override of the ToString() method. Formats the return value so it looks pretty.
49         * Input:   Nothing
50         * Output:  String object containing serialized object data.
51         *-----*/
52         public override string ToString()
53         {
54             return String.Format("{0} ({1}), {2}, {3} [{4}]",
55                 Name, FacilityType, Address, City, PhoneNumber);
56         }
57     }
58 }
59
```

Main menu and adding 3 new items

```
file:///C:/Users/Dan/Box Sync/2014-2015 Summer/CSCI-C 490 (C# Programming)/Project/Phase 2/cView-P2-DanCassidy-Consol...
! Main Interactive Menu !
-----
Please select an option:
1> Add New Item
2> Modify Item
3> Search Items
4> Delete Item
5> Display All Items
6> Exit
Choice: 1

! Add New Item !
-----
Facility Name: South Bend Fire Department
Facility Type: Fire Station
Address: 1222 S Michigan St
City: South Bend
Phone Number: 574-253-9491

! Main Interactive Menu !
-----
Please select an option:
1> Add New Item
2> Modify Item
3> Search Items
4> Delete Item
5> Display All Items
6> Exit
Choice: 1

! Add New Item !
-----
Facility Name: Mishawaka Police Department
Facility Type: Police Station
Address: 200 N Church St
City: Mishawaka
Phone Number: 574-258-1768

! Main Interactive Menu !
-----
Please select an option:
1> Add New Item
2> Modify Item
3> Search Items
4> Delete Item
5> Display All Items
6> Exit
Choice: 1

! Add New Item !
-----
Facility Name: St. Joseph Co. Public Library Lakeville Branch
Facility Type: Library
Address: 120 N Michigan St
City: Lakeville
Phone Number: 574-784-3446

! Main Interactive Menu !
-----
```


Attempting to modify an item: no input, cancelling, and invalid item number

```
file:///C:/Users/Dan/Box Sync/2014-2015 Summer/CSCI-C 490 (C# Programming)/Project/Phase 2/cView-P2-DanCassidy-Consol...
! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 2

! Modify Item -- Existing Items !
Item Facility Name (Type), Address, City [Phone Number]
1 Mishawaka Police Department (Police Station), 200 N Church St, Mishawaka [574-258-1768]
2 South Bend Fire Department (Fire Station), 1222 S Michigan St, South Bend [574-253-9491]
3 St. Joseph Co. Public Library Lakeville Branch (Library), 120 N Michigan St, Lakeville [574-784-3446]
4-3446]
Select item (0 to cancel):
Cancelled.

! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 2

! Modify Item -- Existing Items !
Item Facility Name (Type), Address, City [Phone Number]
1 Mishawaka Police Department (Police Station), 200 N Church St, Mishawaka [574-258-1768]
2 South Bend Fire Department (Fire Station), 1222 S Michigan St, South Bend [574-253-9491]
3 St. Joseph Co. Public Library Lakeville Branch (Library), 120 N Michigan St, Lakeville [574-784-3446]
4-3446]
Select item (0 to cancel): 0
Cancelled.

! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 2

! Modify Item -- Existing Items !
Item Facility Name (Type), Address, City [Phone Number]
1 Mishawaka Police Department (Police Station), 200 N Church St, Mishawaka [574-258-1768]
2 South Bend Fire Department (Fire Station), 1222 S Michigan St, South Bend [574-253-9491]
3 St. Joseph Co. Public Library Lakeville Branch (Library), 120 N Michigan St, Lakeville [574-784-3446]
4-3446]
Select item (0 to cancel): 4
Invalid item.

! Main Interactive Menu !
```

Modifying the name of one of the entries

```
file:///C:/Users/Dan/Box Sync/2014-2015 Summer/CSCI-C 490 (C# Programming)/Project/Phase 2/cView-P2-DanCassidy-Consol...
! Main Interactive Menu !
Please select an option:
1> Add New Item
2> Modify Item
3> Search Items
4> Delete Item
5> Display All Items
6> Exit
Choice: 2

! Modify Item -- Existing Items !
Item  Facility Name <Type>, Address, City [Phone Number]
1  Mishawaka Police Department <Police Station>, 200 N Church St, Mishawaka [574-258-1768]
2  South Bend Fire Department <Fire Station>, 1222 S Michigan St, South Bend [574-253-9491]
3  St. Joseph Co. Public Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]
4-3446]
Select item <0 to cancel>: 3

! Modify Item -- Chosen Item !
Facility Name <Type>, Address, City [Phone Number]
St. Joseph Co. Public Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]
]
Please select the field you would like to work with:
1> Facility Name
2> Facility Type
3> Street Address
4> City
5> Phone Number
6> Back
Choice: 1
Current Facility Name: St. Joseph Co. Public Library Lakeville Branch
New Facility Name: St Jo Co Library Lakeville Branch

! Modify Item -- Chosen Item !
Facility Name <Type>, Address, City [Phone Number]
St Jo Co Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]
]
Please select the field you would like to work with:
1> Facility Name
2> Facility Type
3> Street Address
4> City
5> Phone Number
6> Back
Choice: 6

! Main Interactive Menu !
```

Searching for a city with a search string of "s"

```
file:///C:/Users/Dan/Box Sync/2014-2015 Summer/CSCI-C 490 (C# Programming)/Project/Phase 2/cView-P2-DanCassidy-Consol...
! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 3

! Search Items !
Please select the field you would like to work with:
1) Facility Name
2) Facility Type
3) Street Address
4) City
5) Phone Number
6) Back
Choice: 4

! Search Items -- City !
Enter your search text: s
2 items found.

Facility Name <Type>, Address, City [Phone Number]
Mishawaka Police Department <Police Station>, 200 N Church St, Mishawaka [574-258-1768]
South Bend Fire Department <Fire Station>, 1222 S Michigan St, South Bend [574-253-9491]

! Search Items !
Please select the field you would like to work with:
1) Facility Name
2) Facility Type
3) Street Address
4) City
5) Phone Number
6) Back
Choice: 6

! Main Interactive Menu !
```

Display all items

```
file:///C:/Users/Dan/Box Sync/2014-2015 Summer/CSCI-C 490 (C# Programming)/Project/Phase 2/cView-P2-DanCassidy-Consol...
! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 5

! Display All Items !

Facility Name <Type>, Address, City [Phone Number]
Mishawaka Police Department <Police Station>, 200 N Church St, Mishawaka [574-258-1768]
South Bend Fire Department <Fire Station>, 1222 S Michigan St, South Bend [574-253-9491]
St Jo Co Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]

! Main Interactive Menu !
```

Attempting to delete an item: no input, cancelled, and invalid item number

```
file:///C:/Users/Dan/Box Sync/2014-2015 Summer/CSCI-C 490 (C# Programming)/Project/Phase 2/cView-P2-DanCassidy-Consol...
! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 4

! Delete Item -- Existing Items !
Item  Facility Name (Type), Address, City [Phone Number]
1  Mishawaka Police Department (Police Station), 200 N Church St, Mishawaka [574-258-1768]
2  South Bend Fire Department (Fire Station), 1222 S Michigan St, South Bend [574-253-9491]
3  St Jo Co Library Lakeville Branch (Library), 120 N Michigan St, Lakeville [574-784-3446]
Select item (0 to cancel):

! Delete Item -- Results !
Cancelled.

! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 4

! Delete Item -- Existing Items !
Item  Facility Name (Type), Address, City [Phone Number]
1  Mishawaka Police Department (Police Station), 200 N Church St, Mishawaka [574-258-1768]
2  South Bend Fire Department (Fire Station), 1222 S Michigan St, South Bend [574-253-9491]
3  St Jo Co Library Lakeville Branch (Library), 120 N Michigan St, Lakeville [574-784-3446]
Select item (0 to cancel): 0

! Delete Item -- Results !
Cancelled.

! Main Interactive Menu !
Please select an option:
1) Add New Item
2) Modify Item
3) Search Items
4) Delete Item
5) Display All Items
6) Exit
Choice: 4

! Delete Item -- Existing Items !
Item  Facility Name (Type), Address, City [Phone Number]
1  Mishawaka Police Department (Police Station), 200 N Church St, Mishawaka [574-258-1768]
2  South Bend Fire Department (Fire Station), 1222 S Michigan St, South Bend [574-253-9491]
3  St Jo Co Library Lakeville Branch (Library), 120 N Michigan St, Lakeville [574-784-3446]
Select item (0 to cancel): 4

! Delete Item -- Results !
Invalid item.

! Main Interactive Menu !
```

Deleting the items

```
file:///C:/Users/Dan/Box Sync/2014-2015 Summer/CSCI-C 490 (C# Programming)/Project/Phase 2/cView-P2-DanCassidy-Consol...
! Main Interactive Menu !
Please select an option:
1> Add New Item
2> Modify Item
3> Search Items
4> Delete Item
5> Display All Items
6> Exit
Choice: 4

! Delete Item -- Existing Items !
Item Facility Name <Type>, Address, City [Phone Number]
1 Mishawaka Police Department <Police Station>, 200 N Church St, Mishawaka [574-258-1768]
2 South Bend Fire Department <Fire Station>, 1222 S Michigan St, South Bend [574-253-9491]
3 St Jo Co Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]
Select item (<0 to cancel>): 1

! Delete Item -- Results !
Item 1 has been deleted.
Facility Name <Type>, Address, City [Phone Number]
South Bend Fire Department <Fire Station>, 1222 S Michigan St, South Bend [574-253-9491]
St Jo Co Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]

! Main Interactive Menu !
Please select an option:
1> Add New Item
2> Modify Item
3> Search Items
4> Delete Item
5> Display All Items
6> Exit
Choice: 4

! Delete Item -- Existing Items !
Item Facility Name <Type>, Address, City [Phone Number]
1 South Bend Fire Department <Fire Station>, 1222 S Michigan St, South Bend [574-253-9491]
2 St Jo Co Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]
Select item (<0 to cancel>): 1

! Delete Item -- Results !
Item 1 has been deleted.
Facility Name <Type>, Address, City [Phone Number]
St Jo Co Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]

! Main Interactive Menu !
Please select an option:
1> Add New Item
2> Modify Item
3> Search Items
4> Delete Item
5> Display All Items
6> Exit
Choice: 4

! Delete Item -- Existing Items !
Item Facility Name <Type>, Address, City [Phone Number]
1 St Jo Co Library Lakeville Branch <Library>, 120 N Michigan St, Lakeville [574-784-3446]
Select item (<0 to cancel>): 1

! Delete Item -- Results !
Item 1 has been deleted.
Facility Name <Type>, Address, City [Phone Number]
No items currently stored.

! Main Interactive Menu !
```