```
/*
 * Author: Jonathan Cruz
 * Course: COMP-003A
 * Purpose: Array and List Activity Lecture
 *
 */

namespace COMP0003A.LectureActivity7
{
    internal class Program
    {
        static void Main(string[] args)
        {
            SectionSeparator("Arrays");
            // arrays store a collection of data of the same type (e.g., int, string, objects, etc.)
            // array1 declaration and setting values
            int[] array1 = new int[5]; // declare a single-dimensional array of 5 integers without values
            // you can set values to an index using the examples below:
            array1[0] = 5;  // sets the value for index 0 to 5
            array1[1] = 10; // sets the value for index 1 to 10
            array1[2] = 15; // sets the value for index 2 to 15
            array1[3] = 20; // sets the value for index 3 to 20
            array1[4] = 25; // sets the value for index 4 to 25

            Console.WriteLine($"Length of array1: {array1.Length}\n"); // displays the length/size of the array; output: 5


            // array2 declaration with values
            int[] array2 = new int[] { 1, 3, 5, 7, 9 }; // declares a single-dimensional array element with values

            /* arrays are zero-indexed, meaning the first 'position' starts at index 0
             * values in an array can be accessed using the syntax below:
             * arrayName[indexNumber]
             *
             * e.g.,
             * int[] array2 = new int[] { 1, 3, 5, 7, 9 };
             * array2[0] -> returns 1
             * array2[1] -> returns 3
             * array2[2] -> returns 5
             * array2[3] -> returns 7
             * array2[4] -> returns 9d
             * array2[5] -> RUNTIME ERROR: OutOfBoundsException
             *
             * It is important to note that arrays cannot be dynamically resized.
             * They are stored in sequential blocks of memory, making it extremely fast to access them.
             * Also, you will get a runtime error of OutOfBoundsException
             * when accessing an index that does not exist
             */
            Console.WriteLine("array2 Traversal");
            ArrayTraversal(array2);


            // integer array traversal
            int[] grades = new int[] { 100, 85, 92, 87, 91, 78, 89 };
            Console.WriteLine($"\ngrades Average: {GetAverage(grades)}");


            // string array traversal
            Console.WriteLine("\nstring Traversal");
            // strings are special object consisting of an array of characters
            string message = "hello";
            ArrayTraversal(message);

```

```
65              SectionSeparator("Lists");
66              // like arrays, lists store a collection of data with the same type (e.g.,
                int, string, objects, etc.)
67              // unlike arrays, lists are dynamic and can increase/decrease in size.
68              List<char> alphabet = new List<char>();// declare an empty integer List
69              alphabet.Add('A'); // adds the character at the end of the collection
70              alphabet.Add('B'); // adds the character at the end of the collection
71              alphabet.Add('C'); // adds the character at the end of the collection
72              alphabet.Add('D'); // adds the character at the end of the collection
73              alphabet.Add('E'); // adds the character at the end of the collection
74
75              Console.WriteLine($"Count of alphabet: {alphabet.Count}"); // displays the
                count/size of the list; output: 5
76
77
78              // similar to arrays, you can use the syntax below:
79              // listName[indexNumber] to access a specific content in the collection
80              Console.WriteLine($"alphabet[0]: {alphabet[0]}");
81              Console.WriteLine($"alphabet[2]: {alphabet[2]}");
82              Console.WriteLine($"alphabet[4]: {alphabet[4]}");
83
84              alphabet.Remove('C'); // removes a specific value somewhere inside the
                collection
85              Console.WriteLine("\nContents of alphabet after removing 'C':");
86              ListTraversal(alphabet);
87          }
88
89          /// <summary>
90          /// Section separator method
91          /// </summary>
92          /// <param name="section">String input for seciton name</param>
93          static void SectionSeparator(string section)
94          {
95              Console.WriteLine("".PadRight(50, '*') + $"\n\t\t{section} Section\n" + "".
                PadRight(50,'*'));
96          }
97
98          /// <summary>
99          /// Array traversal
100         /// </summary>
101         /// <param name="array">Integer array to traverse</param>
102         static void ArrayTraversal(int[] array)
103         {
104             // you can use a for-loop or other looping statements for array traversals
105             for (int i = 0; i < array.Length; i++)
106             {
107                 Console.WriteLine($"Array at index {i}: {array[i]}");
108             }
109         }
110
111         /// <summary>
112         /// Array traveral
113         /// </summary>
114         /// <param name="array">String input</param>
115         static void ArrayTraversal(string array)
116         {
117             // you can use a for-loop or other looping statements for array traversals
118             for (int i = 0; i < array.Length; i++)
119             {
120                 Console.WriteLine($"Array at index {i}: {array[i]}");
121             }
122         }
123
124         /// <summary>
125         /// Average grade calculator
126         /// </summary>
127         /// <param name="array">Integer array input</param>
128         /// <returns></returns>
129         static double GetAverage(int[] array)
```

```
130         {
131             int runningTotal = 0;
132 
133             // if you are not planning to manipulate the contents of the array,
134             // you can use a foreach loop
135             foreach (int item in array)
136             {
137                 runningTotal += item; // adds the value of the current item to the
                    runningTotal
138             }
139 
140             return runningTotal / array.Length; // returns the average
141         }
142 
143         /// <summary>
144         /// List traversal using a foreach loop
145         /// </summary>
146         /// <param name="list">Character list input</param>
147         static void ListTraversal(List<char> list)
148         {
149             // you can use a foreach-loop or other looping statements for list traversals
150             foreach (var item in list)
151             {
152                 Console.WriteLine(item);
153             }
154         }
155     }
156 }
```