

DEAKIN UNIVERSITY

OBJECT ORIENTED DEVELOPMENT

ONTRACK SUBMISSION

C# Essentials: Arrays and Lists

Submitted By:
Romil BIJARNIA
s222528574
2024/08/06 05:54

Tutor:
Jotham BARAZANI

Outcome	Weight
Evaluate Code	◆◆◆◆◆
Principles	◆◆◆◆◆
Build Programs	◆◆◆◆◆
Design	◆◆◆◆◆
Justify	◆◆◆◆◆

nicee assignment

August 6, 2024



```
1  // Task 3.1
2
3  // 1. Create and Initialize a Double Array
4  using System;
5
6  class Program
7  {
8      static void Main()
9      {
10         // Declare and initialize the array
11         double[] myArray = new double[10];
12
13         // Assign values to each element
14         myArray[0] = 1.0;
15         myArray[1] = 1.1;
16         myArray[2] = 1.2;
17         myArray[3] = 1.3;
18         myArray[4] = 1.4;
19         myArray[5] = 1.5;
20         myArray[6] = 1.6;
21         myArray[7] = 1.7;
22         myArray[8] = 1.8;
23         myArray[9] = 1.9;
24
25         // Print each element to the console
26         for (int i = 0; i < myArray.Length; i++)
27         {
28             Console.WriteLine("Element at index " + i + ": " + myArray[i]);
29         }
30     }
31 }
32
33
34 // 2. Use a Loop to Assign Values and Print Them
35 using System;
36
37 class Program
38 {
39     static void Main()
40     {
41         // Declare the integer array
42         int[] intArray = new int[10];
43
44         // Use a loop to assign values to each element
45         for (int i = 0; i < intArray.Length; i++)
46         {
47             intArray[i] = i;
48         }
49
50         // Use a loop to print each element
51         for (int i = 0; i < intArray.Length; i++)
52         {
53             Console.WriteLine("Element at index " + i + ": " + intArray[i]);
```

```
54     }
55 }
56 }
57
58
59 // 3. Calculate the Total and Average of Array Elements
60 using System;
61
62 class Program
63 {
64     static void Main()
65     {
66         // Declare and initialize the array
67         int[] grades = { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };
68
69         // Calculate the total
70         int total = 0;
71         for (int i = 0; i < grades.Length; i++)
72         {
73             total += grades[i];
74         }
75
76         // Calculate the average
77         double average = (double)total / grades.Length;
78
79         // Print the results
80         Console.WriteLine("Total: " + total);
81         Console.WriteLine("Number of elements: " + grades.Length);
82         Console.WriteLine("Average: " + average);
83     }
84 }
85
86
87 // 4. Store User Input in an Array
88 using System;
89
90 class Program
91 {
92     static void Main()
93     {
94         // Declare the array to store student names
95         string[] studentNames = new string[6];
96
97         // Collect input from the user
98         for (int i = 0; i < studentNames.Length; i++)
99         {
100             Console.Write("Enter student name " + (i + 1) + ": ");
101             studentNames[i] = Console.ReadLine();
102         }
103
104         // Print each student name
105         Console.WriteLine("Student names:");
106         for (int i = 0; i < studentNames.Length; i++)
```

```
107         {
108             Console.WriteLine("Student " + (i + 1) + ": " + studentNames[i]);
109         }
110     }
111 }
112
113
114 // 5. Find the Largest and Smallest Values in an Array
115 using System;
116
117 class Program
118 {
119     static void Main()
120     {
121         // Declare and initialize the array
122         double[] values = { 1.0, 2.5, 3.8, 4.2, 5.9, 6.1, 7.3, 8.4, 9.5, 10.7 };
123
124         // Find the largest and smallest values
125         double largest = values[0];
126         double smallest = values[0];
127
128         for (int i = 1; i < values.Length; i++)
129         {
130             if (values[i] > largest)
131                 largest = values[i];
132             if (values[i] < smallest)
133                 smallest = values[i];
134         }
135
136         // Print the array
137         Console.WriteLine("Array elements:");
138         for (int i = 0; i < values.Length; i++)
139         {
140             Console.WriteLine("Element at index " + i + ": " + values[i]);
141         }
142
143         // Print the largest and smallest values
144         Console.WriteLine("Largest value: " + largest);
145         Console.WriteLine("Smallest value: " + smallest);
146     }
147 }
148
149
150 // 6. Work with Multi-Dimensional Arrays
151 using System;
152
153 class Program
154 {
155     static void Main()
156     {
157         // Declare and initialize the 2-dimensional array
158         int[,] myArray = new int[3, 4]
159         {
```

```
160         { 1, 2, 3, 4 },
161         { 1, 1, 1, 1 },
162         { 2, 2, 2, 2 }
163     };
164
165     // Print the elements using nested loops
166     for (int i = 0; i < myArray.GetLength(0); i++)
167     {
168         for (int j = 0; j < myArray.GetLength(1); j++)
169         {
170             Console.Write(myArray[i, j] + " ");
171         }
172         Console.WriteLine();
173     }
174 }
175 }
176
177
178 // 7. Work with Lists
179 using System;
180 using System.Collections.Generic;
181
182 class Program
183 {
184     static void Main()
185     {
186         // Create a list to store names
187         List<string> studentNames = new List<string>();
188
189         // Generate a random number of students
190         Random random = new Random();
191         int numStudents = random.Next(1, 13);
192
193         // Collect names from the user
194         for (int i = 0; i < numStudents; i++)
195         {
196             Console.Write("Enter student name " + (i + 1) + ": ");
197             studentNames.Add(Console.ReadLine());
198         }
199
200         // Print the list contents
201         Console.WriteLine("Student names:");
202         foreach (string name in studentNames)
203         {
204             Console.WriteLine(name);
205         }
206     }
207 }
208
209
210 // 8. Palindrome Method
211 using System;
212
```

```
213 class Program
214 {
215     static void Main()
216     {
217         // Test the Palindrome method
218         int[] testArray1 = { 1, 2, 2, 1 };
219         int[] testArray2 = { 3, 2, 1 };
220
221         Console.WriteLine(Palindrome(testArray1)); // Output: True
222         Console.WriteLine(Palindrome(testArray2)); // Output: False
223     }
224
225     static bool Palindrome(int[] array)
226     {
227         if (array.Length < 1)
228             return false;
229
230         for (int i = 0; i < array.Length / 2; i++)
231         {
232             if (array[i] != array[array.Length - 1 - i])
233                 return false;
234         }
235         return true;
236     }
237 }
238
239
240 9. Merge Method
241 using System;
242 using System.Collections.Generic;
243
244 class Program
245 {
246     static void Main()
247     {
248         // Test the Merge method with example lists
249         List<int> listA = new List<int> { 1, 2, 2, 5 };
250         List<int> listB = new List<int> { 1, 3, 4, 5, 7 };
251
252         List<int> mergedList = Merge(listA, listB);
253
254         if (mergedList != null)
255         {
256             Console.WriteLine("Merged list:");
257             foreach (int item in mergedList)
258             {
259                 Console.Write(item + " ");
260             }
261             Console.WriteLine();
262         }
263         else
264         {
265             Console.WriteLine("One or both lists are not sorted.");
266         }
267     }
268 }
```

```
266     }
267 }
268
269 static List<int> Merge(List<int> listA, List<int> listB)
270 {
271     // Check if both lists are sorted
272     if (!IsSorted(listA) || !IsSorted(listB))
273         return null;
274
275     // Create a new list to hold the merged results
276     List<int> mergedList = new List<int>();
277
278     int i = 0, j = 0;
279
280     // Merge the two lists
281     while (i < listA.Count && j < listB.Count)
282     {
283         if (listA[i] <= listB[j])
284         {
285             mergedList.Add(listA[i]);
286             i++;
287         }
288         else
289         {
290             mergedList.Add(listB[j]);
291             j++;
292         }
293     }
294
295     // Add any remaining elements from listA
296     while (i < listA.Count)
297     {
298         mergedList.Add(listA[i]);
299         i++;
300     }
301
302     // Add any remaining elements from listB
303     while (j < listB.Count)
304     {
305         mergedList.Add(listB[j]);
306         j++;
307     }
308
309     return mergedList;
310 }
311
312 static bool IsSorted(List<int> list)
313 {
314     for (int i = 1; i < list.Count; i++)
315     {
316         if (list[i] < list[i - 1])
317             return false;
318     }
```

```
319         return true;
320     }
321 }
322
323
324 // 10. Array Conversion Method
325 using System;
326 using System.Collections.Generic;
327
328 class Program
329 {
330     static void Main()
331     {
332         // Test the ArrayConversion method with an example 2D array
333         int[,] array2D = new int[,]
334         {
335             { 0, 2, 4, 0, 9, 5 },
336             { 7, 1, 3, 3, 2, 1 },
337             { 1, 3, 9, 8, 5, 6 },
338             { 4, 6, 7, 9, 1, 0 }
339         };
340
341         int[] convertedArray = ArrayConversion(array2D);
342         Console.WriteLine("Converted array:");
343         foreach (int item in convertedArray)
344         {
345             Console.Write(item + " ");
346         }
347         Console.WriteLine();
348     }
349
350     static int[] ArrayConversion(int[,] array)
351     {
352         List<int> oddValues = new List<int>();
353
354         // Iterate through the array column by column
355         for (int col = 0; col < array.GetLength(1); col++)
356         {
357             for (int row = 0; row < array.GetLength(0); row++)
358             {
359                 if (array[row, col] % 2 != 0)
360                 {
361                     oddValues.Add(array[row, col]);
362                 }
363             }
364         }
365
366         // Convert the list of odd values to an array and return it
367         return oddValues.ToArray();
368     }
369 }
```