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

```
// Task 3.1

// 1. Create and Initialize a Double Array
using System;

class Program
{
    static void Main()
    {
        // Declare and initialize the array
        double[] myArray = new double[10];

        // Assign values to each element
        myArray[0] = 1.0;
        myArray[1] = 1.1;
        myArray[2] = 1.2;
        myArray[3] = 1.3;
        myArray[4] = 1.4;
        myArray[5] = 1.5;
        myArray[6] = 1.6;
        myArray[7] = 1.7;
        myArray[8] = 1.8;
        myArray[9] = 1.9;

        // Print each element to the console
        for (int i = 0; i < myArray.Length; i++)
        {
            Console.WriteLine("Element at index " + i + ": " + myArray[i]);
        }
    }
}


// 2. Use a Loop to Assign Values and Print Them
using System;

class Program
{
    static void Main()
    {
        // Declare the integer array
        int[] intArray = new int[10];

        // Use a loop to assign values to each element
        for (int i = 0; i < intArray.Length; i++)
        {
            intArray[i] = i;
        }

        // Use a loop to print each element
        for (int i = 0; i < intArray.Length; i++)
        {
            Console.WriteLine("Element at index " + i + ": " + intArray[i]);
```

```
        }
    }
}


// 3. Calculate the Total and Average of Array Elements
using System;

class Program
{
    static void Main()
    {
        // Declare and initialize the array
        int[] grades = { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };

        // Calculate the total
        int total = 0;
        for (int i = 0; i < grades.Length; i++)
        {
            total += grades[i];
        }

        // Calculate the average
        double average = (double)total / grades.Length;

        // Print the results
        Console.WriteLine("Total: " + total);
        Console.WriteLine("Number of elements: " + grades.Length);
        Console.WriteLine("Average: " + average);
    }
}


// 4. Store User Input in an Array
using System;

class Program
{
    static void Main()
    {
        // Declare the array to store student names
        string[] studentNames = new string[6];

        // Collect input from the user
        for (int i = 0; i < studentNames.Length; i++)
        {
            Console.Write("Enter student name " + (i + 1) + ": ");
            studentNames[i] = Console.ReadLine();
        }

        // Print each student name
        Console.WriteLine("Student names:");
        for (int i = 0; i < studentNames.Length; i++)
```

```
        {
            Console.WriteLine("Student " + (i + 1) + ": " + studentNames[i]);
        }
    }
}


// 5. Find the Largest and Smallest Values in an Array
using System;

class Program
{
    static void Main()
    {
        // Declare and initialize the array
        double[] values = { 1.0, 2.5, 3.8, 4.2, 5.9, 6.1, 7.3, 8.4, 9.5, 10.7 };

        // Find the largest and smallest values
        double largest = values[0];
        double smallest = values[0];

        for (int i = 1; i < values.Length; i++)
        {
            if (values[i] > largest)
                largest = values[i];
            if (values[i] < smallest)
                smallest = values[i];
        }

        // Print the array
        Console.WriteLine("Array elements:");
        for (int i = 0; i < values.Length; i++)
        {
            Console.WriteLine("Element at index " + i + ": " + values[i]);
        }

        // Print the largest and smallest values
        Console.WriteLine("Largest value: " + largest);
        Console.WriteLine("Smallest value: " + smallest);
    }
}


// 6. Work with Multi-Dimensional Arrays
using System;

class Program
{
    static void Main()
    {
        // Declare and initialize the 2-dimensional array
        int[,] myArray = new int[3, 4]
        {
```

```
            { 1, 2, 3, 4 },
            { 1, 1, 1, 1 },
            { 2, 2, 2, 2 }
        };

        // Print the elements using nested loops
        for (int i = 0; i < myArray.GetLength(0); i++)
        {
            for (int j = 0; j < myArray.GetLength(1); j++)
            {
                Console.Write(myArray[i, j] + " ");
            }
            Console.WriteLine();
        }
    }
}


// 7. Work with Lists
using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        // Create a list to store names
        List<string> studentNames = new List<string>();

        // Generate a random number of students
        Random random = new Random();
        int numStudents = random.Next(1, 13);

        // Collect names from the user
        for (int i = 0; i < numStudents; i++)
        {
            Console.Write("Enter student name " + (i + 1) + ": ");
            studentNames.Add(Console.ReadLine());
        }

        // Print the list contents
        Console.WriteLine("Student names:");
        foreach (string name in studentNames)
        {
            Console.WriteLine(name);
        }
    }
}


// 8. Palindrome Method
using System;

```

```
class Program
{
    static void Main()
    {
        // Test the Palindrome method
        int[] testArray1 = { 1, 2, 2, 1 };
        int[] testArray2 = { 3, 2, 1 };

        Console.WriteLine(Palindrome(testArray1)); // Output: True
        Console.WriteLine(Palindrome(testArray2)); // Output: False
    }

    static bool Palindrome(int[] array)
    {
        if (array.Length < 1)
            return false;

        for (int i = 0; i < array.Length / 2; i++)
        {
            if (array[i] != array[array.Length - 1 - i])
                return false;
        }
        return true;
    }
}


9. Merge Method
using System;
using System.Collections.Generic;

class Program
{
    static void Main()
    {
        // Test the Merge method with example lists
        List<int> listA = new List<int> { 1, 2, 2, 5 };
        List<int> listB = new List<int> { 1, 3, 4, 5, 7 };

        List<int> mergedList = Merge(listA, listB);

        if (mergedList != null)
        {
            Console.WriteLine("Merged list:");
            foreach (int item in mergedList)
            {
                Console.Write(item + " ");
            }
            Console.WriteLine();
        }
        else
        {
            Console.WriteLine("One or both lists are not sorted.");
```

```
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 }
```