



Universidad Privada del Estado de México

Programación orientada a objetos

Practica evaluación parcial 1

Profesor: Neri Alejandro Alvarez Esperon

Alumno: Yonatan Amaravi Ordoñez
Rodríguez

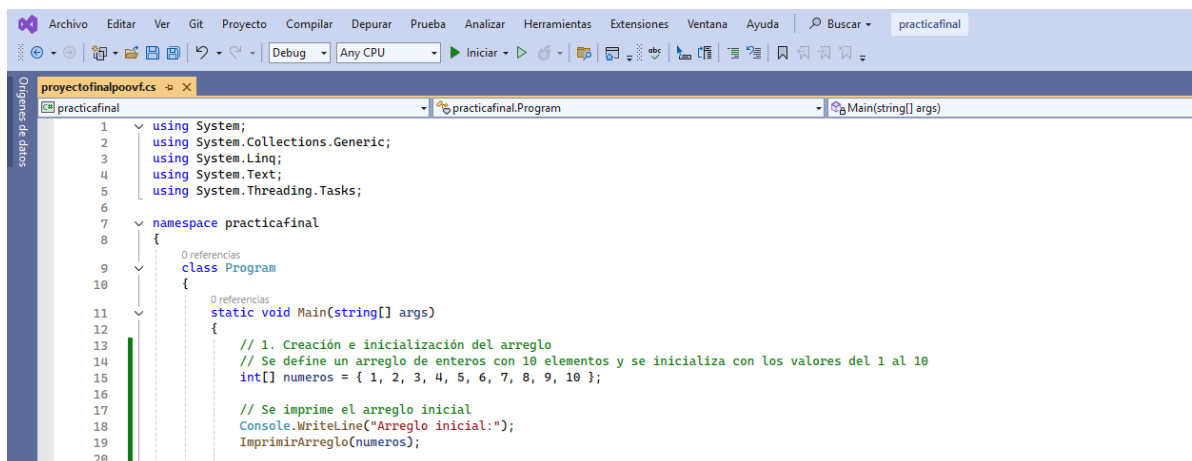
Ingeniería en sistemas computacionales

4to. Cuatrimestre

Este programa en C# realiza varias operaciones con un arreglo de enteros de manera estructurada. A continuación, se explican los resultados obtenidos en cada parte del código:

1. Creación e impresión del arreglo

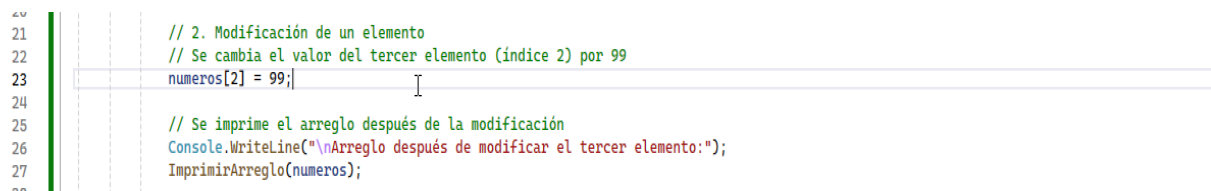
- Se crea un arreglo de enteros numeros con los valores { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }.
- Se imprime el arreglo mediante la función ImprimirArreglo(), que recorre el arreglo e imprime sus elementos separados por espacios.



```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace practicafinal
8  {
9      0 referencias
10     class Program
11     {
12         0 referencias
13         static void Main(string[] args)
14         {
15             // 1. Creación e inicialización del arreglo
16             // Se define un arreglo de enteros con 10 elementos y se inicializa con los valores del 1 al 10
17             int[] numeros = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
18
19             // Se imprime el arreglo inicial
20             Console.WriteLine("Arreglo inicial:");
21             ImprimirArreglo(numeros);
22         }
23     }
24 }
```

2. Modificación del tercer elemento

- Se modifica el valor del índice 2 (tercer elemento) del arreglo, cambiándolo de 3 a 99.
- Se vuelve a imprimir el arreglo para reflejar la modificación.



```
21 // 2. Modificación de un elemento
22 // Se cambia el valor del tercer elemento (índice 2) por 99
23 numeros[2] = 99;
24
25 // Se imprime el arreglo después de la modificación
26 Console.WriteLine("\nArreglo después de modificar el tercer elemento:");
27 ImprimirArreglo(numeros);
28 }
```

3. Suma de los elementos del arreglo

- Se inicializa una variable suma = 0.
- Se recorre el arreglo con un foreach, acumulando cada valor en suma.
- La suma se imprime en la consola.

```
28
29
30 // 3. Suma de los elementos del arreglo
31 // Se inicializa una variable para almacenar la suma
32 int suma = 0;
33
34 // Se recorre el arreglo sumando cada elemento a la variable suma
35 foreach (int num in numeros)
36 {
37     suma += num;
38 }
39
40 // Se imprime la suma de los elementos del arreglo
41 Console.WriteLine("\nLa suma de los elementos del arreglo es: " + suma);
```

4. Búsqueda de un número en el arreglo

- Se usa Array.IndexOf(numeros, 99) para encontrar la posición del número 99 (o cualquier otro numero) en este caso fue 78
- La función IndexOf() devuelve el índice del primer elemento que coincide con 99, que en este caso es 2.
- Se imprime la posición del número 99.
- Si el número no estuviera en el arreglo, se imprimiría "Elemento no encontrado".

```
42 // 4. Búsqueda de un número en el arreglo
43 // Se define el número a buscar
44 int numeroBuscado = 78;
45
46 // Se usa Array.IndexOf para encontrar el índice del número buscado
47 int indice = Array.IndexOf(numeros, numeroBuscado);
48
49 // Se verifica si el número fue encontrado
50 if (indice != -1)
51 {
52     // Se imprime el índice en el que se encontró el número
53     Console.WriteLine("\nEl número " + numeroBuscado + " se encuentra en el índice: " + indice);
54 }
55 else
56 {
57     // Se imprime un mensaje si el número no fue encontrado
58     Console.WriteLine("\nElemento no encontrado");
59 }
60 }
```

Conclusión

- El programa demuestra operaciones fundamentales con arreglos en C#, incluyendo creación, modificación, iteración y búsqueda de elementos.
- Se puede observar cómo los arreglos permiten almacenar y manipular datos de manera estructurada, facilitando la ejecución de operaciones como la suma de elementos y la búsqueda de valores específicos.
- La modificación de elementos dentro del arreglo es sencilla utilizando el índice correspondiente, lo que permite actualizar valores sin necesidad de crear un nuevo arreglo.
- El uso de `foreach` facilita la iteración sobre los elementos del arreglo, mientras que `Array.IndexOf()` simplifica la búsqueda de valores sin necesidad de recorrer manualmente la estructura de datos.
- Este ejercicio es una base esencial para comprender estructuras más avanzadas en C#, como listas (`List<T>`), matrices multidimensionales y colecciones más dinámicas.
- En un contexto práctico, estos conceptos pueden aplicarse en procesamiento de datos, gestión de inventarios, análisis de información y muchas otras áreas de desarrollo de software.

Este tipo de ejercicios fortalece la lógica de programación y la comprensión del manejo de datos en estructuras estáticas como los arreglos.

Resultado en consola:

```
Consola de depuración de Mi... X + v
Arreglo inicial:
1 2 3 4 5 6 7 8 9 10

Arreglo después de modificar el tercer elemento:
1 2 99 4 5 6 7 8 9 10

La suma de los elementos del arreglo es: 151

Elemento no encontrado

C:\Users\Yonatan\source\repos\practicafinal\practicafinal\bin\Debug\practicafinal.exe (proceso 13592) se cerró con el código 0 (0x0).
Para cerrar automáticamente la consola cuando se detiene la depuración, habilite Herramientas ->Opciones ->Depuración -> Cerrar la consola automáticamente al detenerse la depuración.
Presione cualquier tecla para cerrar esta ventana. . . |
```