Arreglos

Contenido

Pasando arreglos a los métodos Ordenar y realizar búsquedas en arreglos Arreglos de varias dimensiones La estructura de repetición foreach



Pasando arreglos a métodos

- Pasar arreglos como parámetros a los métodos especificando el nombre del arreglo sin paréntesis cuadrados
- Los arreglos son pasados por referencia
- Los elementos individuales del arreglo son pasados por valor

```
3
```

```
using System;
using System.Collections.Generic;
using System.Text;
using System. Windows. Forms;
namespace Arreglos
  class PasarArreglo
     static void Main(string[] args)
       int[] a = { 1, 2, 3, 4, 5 };
       Console.WriteLine("Efectos de pasar el arreglo entero" +
         "Ilamada por refrencia\n\nLos elementos del arreglo " +
         "original son \n\t");
       for (int i = 0; i < a.Length; i++)
          Console.WriteLine(" {0} ", a[i]);
               ModificarArreglo(a); // el arreglo es pasado por referencia
       Console. WriteLine("\n\nLos valores del arreglo modificado son:\n\t");
       // despliega los elementos del arreglo a
       for (int i = 0; i < a.Length; i++)
          Console.WriteLine(" {0} ", a[i]);
       Console.WriteLine("\n\nEfectos de pasar un elemento " +
         "del arreglo, llamada-por-valor \n\n"+
          "a[3] antes de ModificarElemento: {0}", a[3]);
              // elemento del arreglo pasado en llamada-por-valor
       ModificarElemento(a[3]);
     Console. WriteLine("\na[3] despues de Modificar Elemento: {0}", a[3]);
       Console.ReadLine();
     }
```



Pasar un arreglo a un método (1)

```
// método que modifica el arreglo que recive,
    // se modifica el arreglo original
     public static void ModificarArreglo(int[] b)
       for (int j = 0; j < b.Length; j++)
          b[i] *= 2;
    // método que modifica el entero pasado a este
    // el original no será modificado
     public static void ModificarElemento(int e)
       Console.WriteLine("\nvalor recibido en ModificarElemento: {0}", e);
       e *= 2:
       Console.WriteLine("\nvalor calulado en ModificarElemento: {0}", e);
Efectos de pasar el arreglo enterollamada por refrencia
Los elementos del arreglo original son
12345
Los valores del arreglo modificado son:
2 4 6 8 10
```

Efectos de pasar un elemento del arreglo, llamada-por-valor

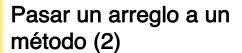
a[3] antes de ModificarElemento: 8

valor recibido en ModificarElemento: 8

valor calulado en ModificarElemento: 16

a[3] despues de ModificarElemento: 8





Ordenar y realizar búsquedas en arreglos

- .NET Framework incluye capacidades de ordenación y búsqueda de arreglos
 - Clase Array
 - Array.Sort
 - Ordena los elementos del arreglo
 - Array.IndexOf
 - Busca un elemento en el arreglo y regresa la posición de la primera ocurrencia, regresa -1 si no lo encuentra
 - Array.BinarySearch
 - Realiza una búsqueda binaria en un arreglo ordenado, mas eficiente que IndexOf

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System. Drawing;
using System.Text;
using System. Windows. Forms;
namespace Arreglos02
  public partial class frmArreglos: Form
     int[] arreglo = new int[14];
     public frmArreglos()
       InitializeComponent();
     // Procedimiento que muestra un arreglo en un cuadro de texto
     public void MostrarArreglo(int[] a, TextBox txt)
       txt.Clear();
       for(int i=0; i<arreglo.Length; i++)</pre>
         txt.Text += string.Format("[{0}] = {1}\r\n",i,arreglo[i]);
     }
     private void btnOrdenar_Click(object sender, EventArgs e)
       Array.Sort(arreglo); // Ordena un arreglo unidimensional
       MostrarArreglo(arreglo, txtOrdenado);
     }
```

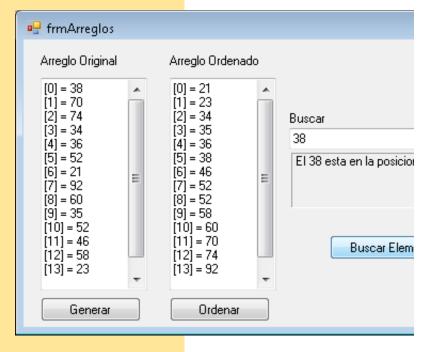


Ordenación y úsqueda en arreglos (1)

```
private void btnGenerar_Click(object sender, EventArgs e)
// Llena el arreglo de numeros enteros aleatorios entre 0 y 100
       Random aleatorio = new Random();
       for(int i=0; i<arreglo.Length; i++)</pre>
          arreglo[i] = aleatorio.Next(0, 100);
        MostrarArreglo(arreglo, txtOriginal);
    private void btnBuscar_Click(object sender, EventArgs e)
       // efectua una búsqueda lineal en el arreglo
       int pos=Array.IndexOf(arreglo, int.Parse(txtBuscar.Text));
       if (pos < 0)
         lblRes.Text = "Número encontrado";
       else
      lblRes.Text = string.Format("El {0} esta en la posición [{1}]",
            arreglo[pos], pos);
```



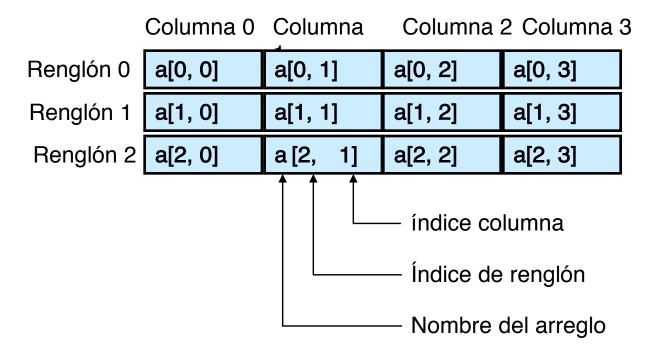
Ordenación y oúsqueda en arreglos (2)



Arreglos de varias dimensiones

- Requiere dos o mas índices para identificar un elemento en particular
- Arreglos rectangulares
 - Con frecuencia representan tablas en la cuales cada renglón es del mismo tamaño y cada columna es del mismo tamaño
 - Por convención el primer índice identifica elementos del reglón y el segundo índice identifica elementos de la columna
- Arreglos dentados (Jagged Arrays)
 - Arreglos de arreglos
 - Los arreglos que compomen a los arreglos dentados pueden ser arreglos de diferentes tamaños

Arreglos de varias dimensiones



Arreglo con doble con tres renglones y cuatro columnas



```
using System;
using System.Collections.Generic;
using System.Text;
using System. Windows. Forms;
namespace Arreglos
  class ArreglosDosDimensiones
     static void Main(string[] args)
       // declaración e inicialización de un arreglo rectangular
       int[,] arreglo1 = new int[,] { 1, 2, 3}, { 4, 5, 6};
       // declaración e inicialización de arreglos dentados
       int[][] arreglo2 = new int[3][];
       arreglo2[0] = new int[] { 1, 2 };
       arreglo2[1] = new int[] { 3 };
       arreglo2[2] = new int[] { 4, 5, 6 };
       Console.WriteLine("\nValores en arreglo1 por renglón son\n");
       // output values in arreglo1
       for (int i = 0; i < arreglo1.GetLength(0); i++)
          for (int j = 0; j < arreglo1. GetLength(1); j++)
            Console.Write(" {0} ",arreglo1[i, i]);
          Console.WriteLine();
       Console.WriteLine("\nValores en arreglo2 por renglón son\n");
       // salida de los elementos en el arreglo2
       for (int i = 0; i < arreglo2.Length; i++)
          for (int j = 0; j < arreglo2[i].Length; j++)
            Console.Write(" {0} ", arreglo2[i][j]);
          Console.WriteLine();
       Console.ReadLine();
```



Arreglo Dos Dimensiones

Valores en arreglo1 por renglón son

1 2 3 4 5 6

Valores en arreglo2 por renglón son

1 2 3 4 5 6

```
11
```

Arreglo Doble (1)

```
using System:
using System.Collections.Generic;
using System.Text;
namespace Arregios
  class ArregloDoble
     static int[][] califs;
     static int estudiantes, examenes;
     static void Main(string[] args)
        califs = new int[3][];
        califs[0] = new int[] { 77, 68, 86, 73 };
        califs[1] = new int[] \{ 96, 87, 89, 81 \};
        califs[2] = new int[] { 70, 90, 86, 81 };
        estudiantes = califs.Length; // numero de estudiantes
        examenes = califs[0].Length; // numero de examenes
                 // relleno de linea de encabezados
        Console.Write("
                 // imprime encabezados de columnas
        for (int i = 0; i < \text{examenes}; i++)
          Console.Write(" [{0}] ",i);
                 // imprime renglones
        for (int i = 0; i < estudiantes; i++)
          Console.Write("\ncalifs[{0}] ",i);
          for (int j = 0; j < examenes; j++)
             Console.Write("{0} ",califs[i][j]);
                 Console.Write("\n\nCalificación más baja: {0}\n", Minima());
        Console.Write("Calificación más alta: {0} \n\n", Maxima());
                 for (int i = 0; i < estudiantes; i++)
         Console. WriteLine ("El promedio del estudiante {0} es {1}",
          i,Promedio(califs[i]));
        Console.ReadLine();
```

```
// encuentra la calificación más baja en el arreglo
 static public int Minima()
    int califMinima = 100;
    for (int i = 0; i < estudiantes; i++)
      for (int j = 0; j < \text{examenes}; j++)
         if (califs[i][j] < califMinima)</pre>
            califMinima = califs[i][j];
    return califMinima;
 // encuentra la calificación más alta en el arreglo
 public static int Maxima()
    int califMaxima = 0;
    for (int i = 0; i < estudiantes; i++)
       for (int j = 0; j < \text{examenes}; j++)
         if (califs[i][j] > califMaxima)
            califMaxima = califs[i][j];
    return califMaxima;
// determina el promedio para un estudiante en particular
 public static double Promedio(int[] conjuntoCalifs)
    int total = 0;
    for (int i = 0; i < conjuntoCalifs.Length; i++)</pre>
      total += conjuntoCalifs[i];
    return (double)total / conjuntoCalifs.Length;
                                                                                      Calificación más baja: 68
```



Arreglo Doble (2)

[0] [1] [2] [3] califs[0] 77 68 86 73 califs[1] 96 87 89 81 califs[2] 70 90 86 81

Calificación más alta: 96

- El promedio del estudiante 0 es 76
- El promedio del estudiante 1 es 88.25
- El promedio del estudiante 2 es 81.75

La estructura de repetición foreach

- Es utilizada para iterar através de valores en las estructuras de datos tales como arreglos
- No tiene contador
- Se utiliza una variable para representar el valor de cada elemento

```
using System;
using System.Collections.Generic;
using System.Text;
namespace Arreglos
  class ForEach
     static void Main(string[] args)
       int[,] arregloCalifs = { { 77, 68, 86, 73 },
          { 98, 87, 89, 81 }, { 70, 90, 86, 81 } };
       int califBaja = 100;
       foreach (int grade in arregloCalifs)
          if (grade < califBaja)
             califBaja = grade;
       Console. WriteLine ("La calificación mas baja es {0}",
                                                                                       califBaja);
       Console.ReadLine();
```



For Each

La calificación mas baja es 68