

Ejercicios Arrays

[Descargar estos ejercicios](#)

Índice

1. [Ejercicio 1](#)
2. [Ejercicio 2](#)
3. [Ejercicio 3](#)
4. [Ejercicio 4](#)
5. [Ejercicio 5](#)
6.  [Ejercicio 6](#)
7. [Ejercicio 7](#)
8.  [Ejercicio 8](#)
9.  [Ejercicio 9](#)
10.  [Ejercicio 10](#)
11. [Ejercicio 11](#)
12. [Ejercicio 12](#)

Ejercicio 1

Realiza un programa que asigne datos **aleatoriamente** a un array de n elementos, y a continuación muestre el contenido de dicho array utilizando la instrucción **foreach**.

Nota: Para generar un número aleatorio, puedes usar la función:

```
Random random=new Random();  
int numero=random.Next(minValue,  maxValue+1);
```

Ejercicio 2

Rellena un array de **10 números** de tipo **double**, de forma aleatoria, y visualiza los que estén en una **posición** que sea múltiplo de cuatro.

Para generar un número real entre **0 y 100** puedes hacer:

```
Random semilla = new Random();  
double numeroReal = semilla.NextDouble() * 100d;
```

Ejercicio 3

Programa que sume los **valores** de un array de 10 elementos reales.

Ejercicio 4

Rellena un array de **10 caracteres** de forma aleatoria y luego sobre el mismo array modifíca, de forma que los elementos que estén en mayúsculas pasen a ser minúscula y los minúscula a mayúscula. Visualiza el array con la instrucción **foreach**.

Nota: Para generar caracteres aleatorios podéis generar números entre el rango de la tabla ASCII 65-122 y descartar los que no sean letras, y posteriormente castearlos a char. Para saber si un carácter está en mayúscula o minúsculas puedes usar:

```
bool char.IsLower(char c);  
bool char.IsUpper(char c);
```

Ejercicio 5

Carga un array numérico de diez elementos, primero visualízalo con la instrucción **foreach** y luego visualiza los elementos cuyo **contenido** sea par, indicando su posición.

✓ Ejercicio 6

Crea un array de 10 elementos, visualiza **el elemento mayor de la serie y la posición que ocupa**. Si hay varios iguales, sólo el primero.

Ejercicio 7

Crea un array aleatorio de enteros con 10 elementos y llámalo **V**.

- Con los elementos **pares** crea un array **P**, ordénalo en sentido **creciente** y visualízalo.
- Con los elementos **impares** crea un array **I** y tras ordenarlo en sentido **decreciente**, visualízalo.

Nota 1: Crea los métodos necesarios para evitar repetir código.

Nota 2: Las siguientes funciones de utilidad te pueden ser útiles para ordenar...

```
void Array.Sort(arrayAOrdenar);  
void Array.Reverse(arrayAinvertir);
```

✓ Ejercicio 8

Implementa un programa en C#, que dado un número entero sin signo introducido por teclado, me diga si es **capicúa**. Un ejemplo de ejecución sería...

```
Número: 1234321  
Es capicúa.
```

Nota: Puedes usar el siguiente código para leer un número en forma de array de caracteres.

```
char[] numero = Console.ReadLine().ToCharArray();
```



Ejercicio 9

Crea un menú con tres opciones:

1. Registrarse en el sistema.
2. Entrar al sistema.
3. Salir del programa.

1. Si seleccionamos la opción *'Registrarse en el sistema'*, aparecerá una ventana que te pida una contraseña, que tendrás que repetir para darla como válida.

Mientras que estás escribiendo la contraseña solamente se verán los caracteres *****

Registrarse:

Contraseña: ****

Comprobar Contraseña: *****

Las contraseñas se leerán directamente en un **array de caracteres de longitud máxima 20** y tras la validar el proceso de registro viendo que las dos contraseñas son iguales, **la contraseña la guardaremos un array de caracteres** no almacenandose ni como String o StringBuilder en ningún momento.

2. En la opción de *'Entrar al sistema'*, introduciremos una cadena por teclado, se comprobará si la cadena introducida es igual a la contraseña guardada en el array. Se avisará con un texto si la entrada ha sido correcta.

Entrar:

Introduce Contraseña: **

La contraseña es correcta/incorrecta

No podremos entrar en esta opción sin habernos registrado en el sistema y si lo hacemos deberemos indicarle al usuario que antes debería registrarse.



Nota: tendremos un método llamado **RecogeContraseña** que leerá un texto de forma oculta y al que llamaremos las veces necesarias en ambas opciones.

Para recoger contraseña utilizaremos **Console.ReadKey(true)**, que permite leer un carácter de teclado con la opción **true** no se mostrará por pantalla el eco de la tecla pulsada. Al mismo

tiempo puedes mostrar un asterisco, y tenemos el efecto deseado. Más o menos un boceto de lo que deberíamos hacer sería lo siguiente ...

```
ConsoleKeyInfo teclaFisica;
do
{
    // Leo una tecla 'física' del teclado sin eco por pantalla
    teclaFisica = Console.ReadKey(true);

    // teclaFisica.Key contiene tecla 'física' y si no es intro me la guardo
    // y muestro un asterisco
    if (teclaFisica.Key != ConsoleKey.Enter)
    {
        // Hago yo su eco
        Console.Write("*");
        // teclaFisica.KeyChar contiene el caracter que representa la tecla
        // física pulsada.
        char caracter = teclaFisica.KeyChar;
    }
    // Leo mientras no sea intro
} while (teclaFisica.Key != ConsoleKey.Enter);
```

✓ Ejercicio 10

Introduce por teclado una secuencia de **calificaciones** de los alumnos de un instituto (números enteros entre cero y diez).

- La secuencia termina con la introducción de un número **menor que cero o mayor que diez**.
- Se supone que como máximo podemos tener **25 alumnos**.
- Se trata de obtener **la frecuencia** de las notas (*número de veces que cada nota aparece*).

💡 **Pista:** Puedes usar un array para guardar las frecuencias, relacionando la posición del array con la nota del alumno. Incrementando el contenido de la posición **i**, cada vez que salga una nota **n**.

Ejemplo: Imagina que tengo 10 notas `notas = [2][4][4][5][5][5][6][7][7][9]`

En un array **de tamaño 11** llamado `frecuenciaNotas = [0][0][0][0][0][0][0][0][0][0][0]` guardaremos la cuenta de cada una de ellas. Fíjate, que los índices en ese array van del **0** al **10** que son los posibles valores válidos de notas. Por tanto, en la posición de índice **5** guardaremos la cuenta de las veces que ha aparecido la **nota 5** en `notas` que es **3**.

Al final del proceso de `notas` el array `frecuenciaNotas` quedará ...


```
0   1   2   3   4   5   6   7   8   9   10
[ 0 ][ 0 ][ 1 ][ 0 ][ 2 ][ 3 ][ 1 ][ 2 ][ 0 ][ 1 ][ 0 ]
```

Ejercicio 11

Introduce un array de 10 elementos y **desplaza** todos sus componentes una posición hacia la derecha, colocando el último en la primera posición. Visualiza el array antes y después del desplazamiento.

Ejercicio 12

Dado el array de enteros `int[] pol1 = new int[]{5, -7, -3, 0, 9}`, que representa al polinomio $9x^4 - 3x^2 - 7x + 5$ donde el índice representa al exponente del monomio y el valor su coeficiente y el array `int[] pol2 = new int[]{-1, 0, 4}` que representa al polinomio $4x^2 - 1$.

 **Nota:** Fíjate que cuando el coeficiente es 0 el monomio correspondiente no se representa:

```
+9x4 +0x3 -3x2 -7x1 +5x0
      +4x2 +0x1 -1x0
-----
+9x4 +0x3 +1x2 -7x1 +4x0
```

Implementa un algoritmo que sume los polinomios, meta el resultado en otro array y muestre el resultado de la suma.