Cadenas de texto

Uso básico

-Los textos se definen entre comillas dobles en el código.

-Se leen de teclado directamente con Console.ReadLine().

-Enlazamos texto con +.

```
string texto1 = "Hola, buenas";
string texto2 = Console.ReadLine();
string texto3 = "Has escrito " + texto2;
```

Acceso a las letras de la cadena

C# trata los textos como arrays de caracteres,
Podemos usar los corchetes [] para acceder a una letra concreta, desde 0 hasta el tamaño (Length) del texto.

```
string texto = "Hola, buenas";
char letra = texto[1];              // 'o'
for (int i = 0; i < texto.Length; i++)
{
    Console.WriteLine(texto[i]);
}
```

Extraer una subcadena

La instrucción **substring** extrae una subcadena de la cadena original,le indicamos la posición desde la que queremos empezar y la cantidad de caracteres que queremos extraer.

si no marcamos la cantidad,extraerá desde el punto marcado hasta el final de la cadena.

```
string texto = "Hola, buenas";
string subcadena1 = texto.Substring(6, 3);  // "bue"
string subcadena2 = texto.Substring(6);     // "buenas"
```

Buscar en una subcadena

La instrucción **IndexOf** indica en que posición se encuentra un texto dentro de otro.

Devuelve el valor -1 si no lo encuentra.

Podemos indicar un segundo parámetro,que indica desde que posición empezará a buscar.

```
string texto = "Hola, buenas";
int posicion = texto.IndexOf("buenas");     // 6
int posicion2 = texto.IndexOf("buenas", 7); // -1
```

Alternativas de búsqueda

-La instrucción **LastIndexOf** es similar a **IndexOf**,pero empieza a buscar por el final.

-La instrucción **Contains** indica si el texto contiene el texto indicado o no.

-Las instrucciones **StartsWith** y **EndsWith** comprueban si el texto empieza o acaba por la cadena indicada,respectivamente.

```
string texto = "Hola, buenas";
bool resultado1 = texto.Contains("buenas");     // true
bool resultado2 = texto.StartsWith("buenas");   // false
bool resultado3 = texto.EndsWith("buenas");     // true
```

Otras manipulaciones de cadenas

-La instrucción **ToUpper()** Devuelve un texto igual que el orginal,pero en mayúsculas.

-La instrucción **ToLower()** Devuelve un texto igual que el orginal,pero en minúsculas.

-La instrucción **Insert(posición,subcadena)** inserta la subcadena en la posición indicada.

-La instrucción **Remove(posición,cantidad)** quita *cantidad* caracteres desde la posición indicada.

-La instrucción **Replace(textoviejo,textonuevo)** reemplazará el texto viejo por el texto nuevo.

```
string texto = "Hola buenas";
Console.WriteLine(texto.ToUpper());
 // HOLA BUENAS
Console.WriteLine(texto.Insert(5, "muy "));
// Hola muy buenas
Console.WriteLine(texto.Replace("Hola", "Qué tal?"));
// Qué tal? buenas
```

Cualquier operación que manipule una cadena (**ToLower**, **Substring**, **Replace**...) NUNCA modifica la cadena original. SIEMPRE se obtiene una cadena alternativa como resultado.

```
string texto = "Hola, buenas";
texto.ToUpper();
Console.WriteLine(texto);          // Hola, buenas
texto = texto.ToUpper();
Console.WriteLine(texto);          // HOLA, BUENAS
```

Descomponer una cadena en fragmentos

La instrucción **Split** descompone la cadena en tantos fragmentos como haya,a partir del delimitador que indiquemos.

Obtenemos un array con todas las partes que se han extraído.

Por defecto,si no se especifica,se asume que el delimitador es un espacio en blanco.

```
string texto = "Hola buenas tardes";
string[] partes = texto.Split(' ');
// {"Hola", "buenas", "tardes"}
```

Podemos especificar un array de delimitadores,en cuyo caso se dividirá la cadena en fragmentos si encuentra alguno de los delimitadores.

```
string texto = "Hola,buenas tardes";
char[] delimitadores = {',', ' '};
string[] partes = texto.Split(delimitadores);
// {"Hola", "buenas", "tardes"}
```

Unir una cadena a partir de fragmentos

La instrucción **String.Join** une los fragmentos de un array en una sola cadena,uniéndolas por el delimitador que indiquemos.

```
string texto = "Uno Dos Tres Cuatro";
string[] partes = texto.Split();
// {"Uno", "Dos", "Tres", "Cuatro"}
Console.WriteLine(String.Join("-", partes));
// Uno-Dos-Tres-Cuatro
```

Comparación de cadenas

Podemos comparar cadenas de texto para saber si son iguales con los operados == o **!=**.

Para ver cual es mayor que otra (alfabéticamente) usamos la instrucción **CompareTo**.

texto1.CompareTo(texto2) devuelve un valor entero que será:

-Negativo si texto1 es menor (alfabéticamente) que texto2

-Positivo si texto1 es mayor (alfabéticamente) que texto2

-Cero si son iguales

```
string texto1 = "Hola", texto2 = "Buenas";
if (texto1.CompareTo(texto2) > 0)   // true (H > B)

string texto1 = "Hola", texto2 = "buenas";
if (texto1.CompareTo(texto2) > 0)
// false (las minúsculas van después en ASCII)
```

Comparación ignorando mayúsculas o minúsculas

Podemos convertir ambos textos a mayúsculas/minúsculas y luego comparar.

```
string texto1 = "Hola", texto2 = "buenas";
if (texto1.ToUpper().CompareTo(texto2.ToUpper()) > 0)
// true (H > B)
```

Podemos usar **String.Compare**, pasando los dos textos a comparar y un tercer dato que será **true** si queremos ignorar mayúsculas/minúsculas.

```
string texto1 = "Hola", texto2 = "buenas";
if (String.Compare(texto1, texto2, true) > 0)   // true (H > B)
```

Cadenas modificables

Las cadenas normales no se pueden modificar,con **StringBuilder** podemos realizar cambios sobre la cadena orginal.

Debemos incluir con ***using*** el espacio ***System.Text***.

```
using System.Text;
...
StringBuilder cadenaModificable = new
StringBuilder("hola");
cadenaModificable[0] = 'H';     // "Hola"
string textoNormal = cadenaModificable.ToString();
```

Cadenas repetitivas

Podemos crear una cadena con un símbolo repetido n veces.

```
string asteriscos = new String('*', 10);        // **********
```