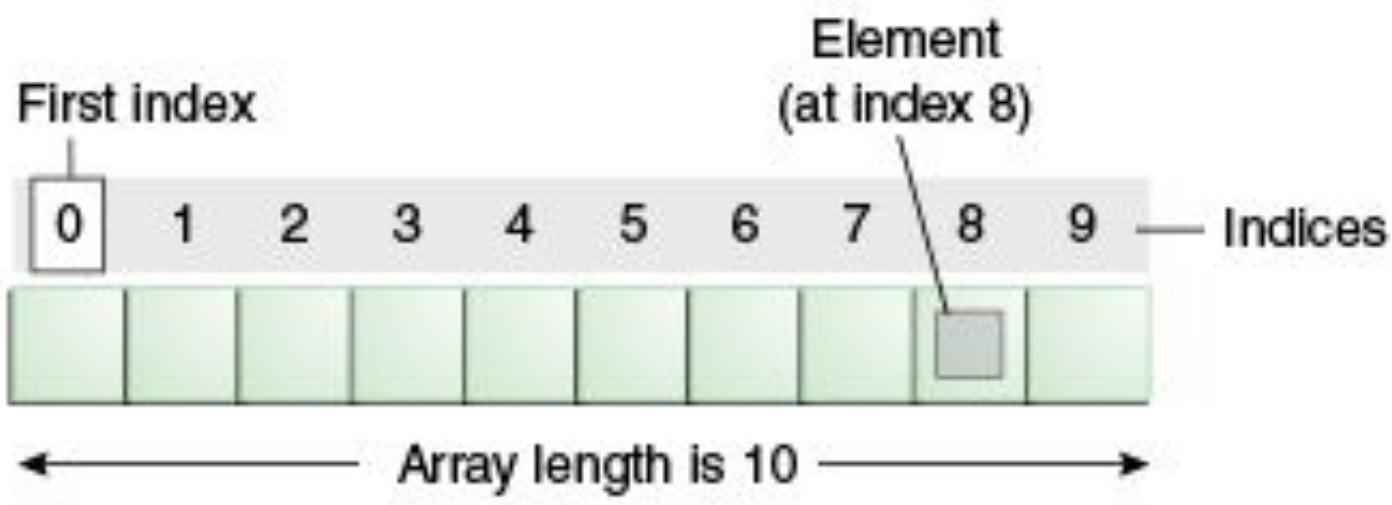


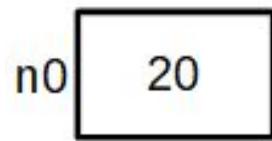
# Introducció a les taules



# Variables i taules

Imagina que un programa necessita un valor enter.

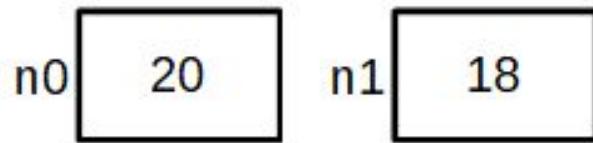
```
int n0 = 20;
```



# Variables i taules

Imagina que un programa necessita dos valors enters.

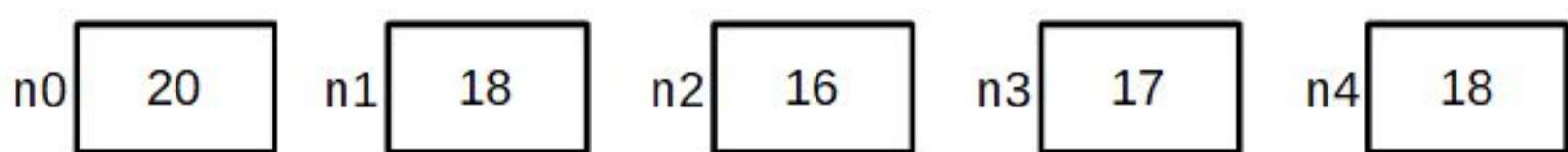
```
int n0 = 20;  
int n1 = 18;
```



# Variables i taules

Imagina que un programa necessita cinc valors enters.

```
int n0 = 20;  
int n1 = 18;  
int n2 = 16;  
int n3 = 17;  
int n4 = 18;
```



# Variables i taules

Imagina que un programa necessita deu valors enters.

```
int n0 = 20;      int n5 = 15;  
int n1 = 18;      int n6 = 21;  
int n2 = 16;      int n7 = 19;  
int n3 = 17;      int n8 = 11;  
int n4 = 18;      int n9 = 13;
```



Imagina que un programa necessita 1000 valors!

Imagina que un programa necessita 10000 valors!

Imagina que un programa necessita 1000000 valors!

# Taules

Les taules (arrays) són unes estructures de dades que ens permeten guardar més d'un valor del mateix tipus.

Les taules tenen una longitud fixe, que es defineix en crear la taula. Un cop creada la taula, la longitud no es pot modificar.

```
int[] nums = new int[10];
```

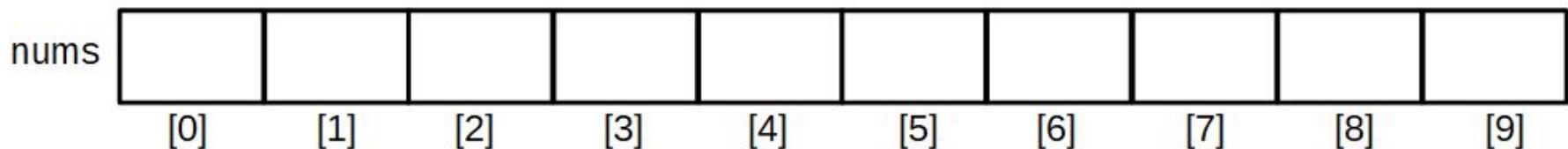


Es crea una taula que conté 10 posicions buides.

```
nums.Length => 10
```

# Assignar valors a un a taula

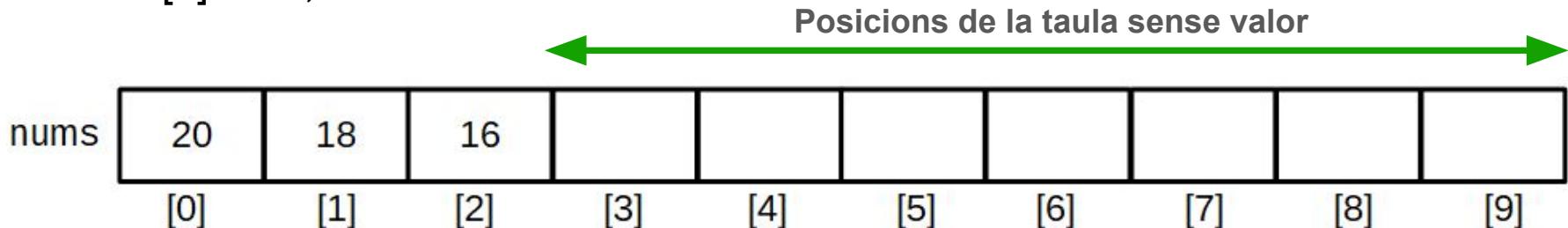
Els valors d'una taula es poden accedir a través del seu **índex**.



nums[0] = 20;

nums[1] = 18;

nums[2] = 16;



# Inicialitzar taules

En crear una nova taula, es pot inicialitzar amb uns valors per defecte.

```
int[] nums = new int[10] { 2, 8, 16, 17, 16, 19, 21, 15, 23, 20 };
```

nums	2	8	16	17	16	19	21	15	23	20
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

També pots inicialitzar una taula de la següent manera:

```
int[] nums = { 2, 8, 16, 17, 16, 19, 21, 15, 23, 20 };
```

nums	2	8	16	17	16	19	21	15	23	20
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

# Llegir valors de la taula.

Llegir el primer valor de la taula.

```
int num = nums[0];  
Console.WriteLine(num);
```

2

Llegir el segon valor de la taula.

```
int num = nums[1];  
Console.WriteLine(num);
```

8

Llegir el tercer valor de la taula.

```
int num = nums[2];  
Console.WriteLine(num);
```

16

nums	2	8	16	17	16	19	21	15	23	20
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

# Llegir valors de la taula.

Llegir l'últim valor de la taula.

```
int num = nums[9];  
Console.WriteLine(num);
```

ó

```
int num = nums[nums.Length - 1];  
Console.WriteLine(num);
```

20

20

Llegir el penúltim valor de la taula.

```
int num = nums[nums.Length - 2];  
Console.WriteLine(num);
```

23

nums	2	8	16	17	16	19	21	15	23	20
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

## Exception: IndexOutOfRangeException

Les taules només s'han d'accedir a través dels índexs vàlids de la taula:  
 $[0 - \text{nums.Length}-1]$

El següents exemples generen errors doncs es vol accedir a zones no vàlides.

```
int num = nums[-2];
Console.WriteLine(num);
```

ó

```
int num = nums[10];
Console.WriteLine(num);
```

```
0 referencias
static void Main(string[] args)
{
    int[] nums = { 2, 8, 16, 17, 16, 19, 21, 15, 23, 20 };
    Console.WriteLine(nums[20]); ✖
}
```

Excepción no controlada

**System.IndexOutOfRangeException:** 'Index was outside the bounds of the array.'

[Mostrar pila de llamadas](#) | [Ver detalles](#) | [Copiar detalles](#)

[Configuración de excepciones](#)

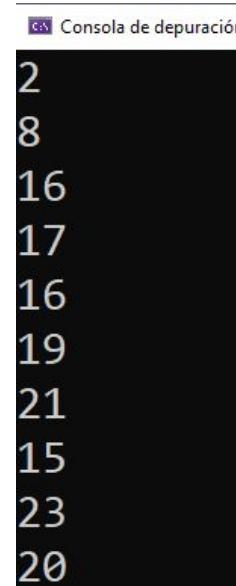
nums	2	8	16	17	16	19	21	15	23	20
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

# Recorreguts

Llistar tots els elements d'una taula.

```
int[] nums = new int[10] { 2, 8, 16, 17, 16, 19, 21, 15, 23, 20 };
```

```
for (int i = 0; i < nums.Length; i++)
{
    Console.WriteLine(nums[i]);
}
```



Consola de depuración

```
2
8
16
17
16
19
21
15
23
20
```

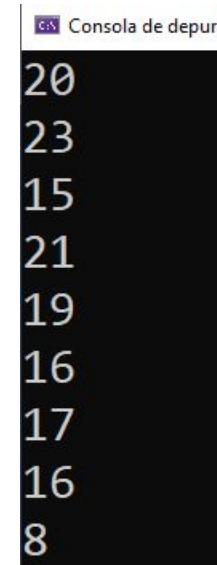
nums	2	8	16	17	16	19	21	15	23	20
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

# Recorreguts

Llistar tots els elements d'una taula al revés.

```
int[] nums = new int[10] { 2, 8, 16, 17, 16, 19, 21, 15, 23, 20 };
```

```
for (int i = nums.Length - 1; i > 0; i--)
{
    Console.WriteLine(nums[i]);
}
```



The screenshot shows a debugger's output window titled "Consola de depuración". It displays the following text:  
20  
23  
15  
21  
19  
16  
17  
16  
8

nums	2	8	16	17	16	19	21	15	23	20
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

# Cerques

Comprovar si la taula nums conté el número 17.

```
int[] nums = new int[10] { 2, 8, 16, 17, 16, 19, 21, 15, 23, 20 };
```

```
int pos = 0;
bool trobat = false;
```

```
while (!trobat && pos < nums.Length)
{
    trobat = nums[pos] == 17;
    if (!trobat) pos++;
}
```

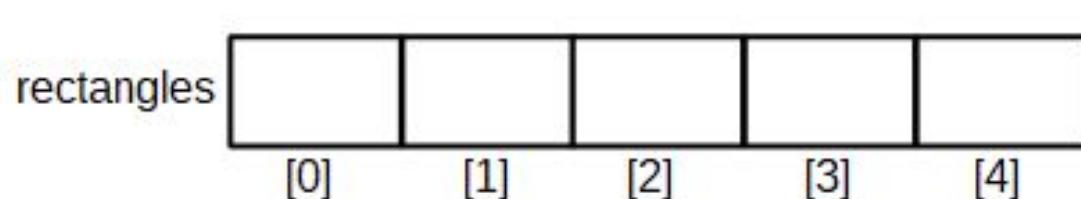
```
if (trobat)
    Console.WriteLine($"HI HA EL NÚMERO 17 A LA POSICIÓ {pos}");
else
    Console.WriteLine($"NO HI HA EL NÚMERO 17");
```



# Taules d'objectes

Una taula pot estar format per objectes d'una classe.

```
Rectangle[] rectangles = new Rectangle[5];
```

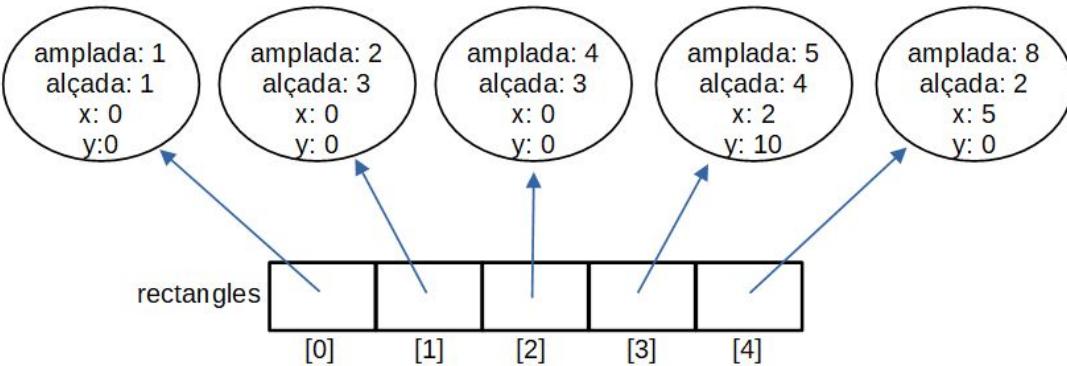


# Taules d'objectes

Omplir una taula de rectangles amb objectes de tipus Rectangle.

```
Rectangle[] rectangles = new Rectangle[5];
```

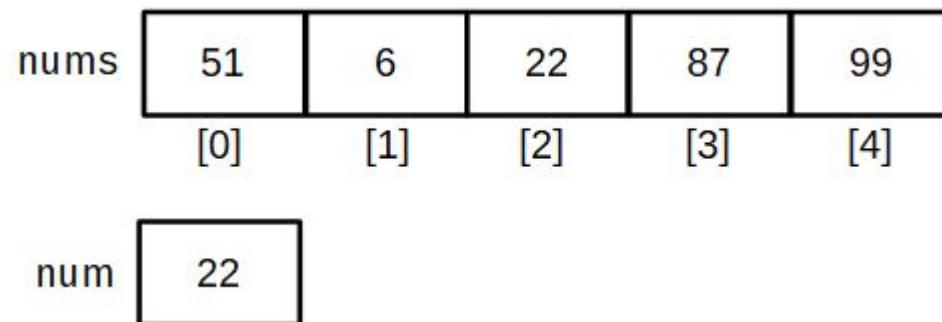
```
rectangles[0] = new Rectangle();  
rectangles[1] = new Rectangle(2, 2, 0, 0);  
rectangles[2] = new Rectangle(4, 3, 0, 0);  
rectangles[3] = new Rectangle(5, 4, 2, 10);  
rectangles[4] = new Rectangle(8, 2, 5, 0);
```



# Còpia per valor

Amb els tipus bàsics (int, double, bool, char) es realitza la **còpia per valor**.

```
int[] nums = new int[5] { 51, 6, 22, 87, 99 };
int num;
num = nums[2];
```



# Còpia per referència

Amb els objectes es realitza la **còpia per referència**.

```
Rectangle[] rectangles = new Rectangle[5];
```

```
Rectangle rec;
```

```
rectangles[0] = new Rectangle(1,1,0,0);
```

```
rectangles[1] = new Rectangle(2, 2, 0, 0);
```

```
rectangles[2] = new Rectangle(4, 3, 0, 0);
```

```
rectangles[3] = new Rectangle(5, 4, 2, 10);
```

```
rectangles[4] = new Rectangle(8, 2, 5, 0);
```

```
rec = rectangles[2];
```

