

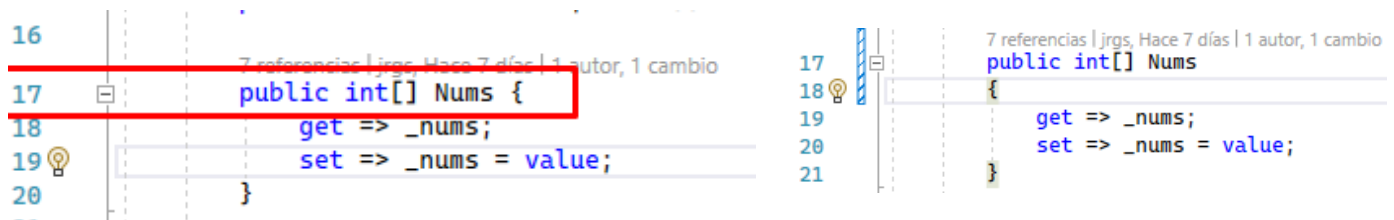
1. **(DOSSIER - 1,5 pt)** Realizar el diseño de pruebas (caja negra) para el constructor con parámetro de la clase *loto*.
2. **(COMMIT - 1,5 pt)** Por último, crear los métodos de prueba para aquellos casos en que se generan errores en el punto anterior.

Después de cada paso indicado como COMMIT se deberá de realizar un *commit* (sólo de manera local). Una vez terminado, deberéis de subir vuestro proyecto a GitHub y enviar la *PullRequest* correspondiente.

Como en el examen anterior y las prácticas, se deberá incluir un dossier explicativo para los puntos (1), (3) y (4). Os recomiendo guardar el dossier en el mismo directorio del proyecto para que automáticamente se haga también *commit* del mismo.

Punto 1.

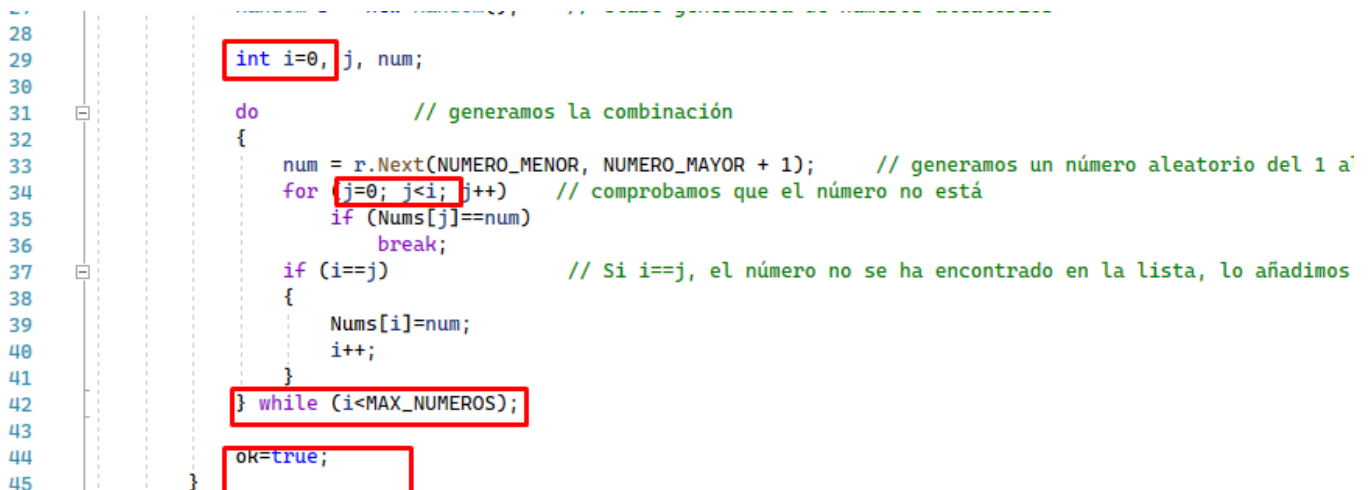
Identacion de llaves



```
16
17 public int[] Nums {
18     get => _nums;
19     set => _nums = value;
20 }

17
18 public int[] Nums
19 {
20     get => _nums;
21     set => _nums = value;
22 }
```

--Faltan el espacio alrededor de los operadores



```
28
29 int i=0, j, num;
30
31 do // generamos la combinación
32 {
33     num = r.Next(NUMERO_MENOR, NUMERO_MAYOR + 1); // generamos un número aleatorio del 1 a
34     for (j=0; j<i; j++) // comprobamos que el número no está
35     {
36         if (Nums[j]==num)
37             break;
38         if (i==j) // Si i==j, el número no se ha encontrado en la lista, lo añadimos
39         {
40             Nums[i]=num;
41             i++;
42         }
43     } while (i<MAX_NUMEROS);
44     OK=true;
45 }
```

Metodo en minúsculas,

```
75 1 referencia | as hace 7 días | 1 autor, 1 cambio  
76 public int comprobar(int[] premi)  
77 {  
78     int a=0; // número de aciertos  
79     for (int i=0; i<MAX_NUMEROS; i++)  
80         for (int j=0; j<MAX_NUMEROS; j++)  
81             if (premi[i]==Nms[j]) a++;  
82     return a;  
83 }
```

```
75 1 referencia | cambios | 0 autores, 0 cambios  
76 public int Comprobar(int[] premi)  
77 {  
78     int a=0; // número de aciertos  
79     for (int i=0; i<MAX_NUMEROS; i++)  
80         for (int j=0; j<MAX_NUMEROS; j++)  
81             if (premi[i]==Nms[j]) a++;  
82     return a;  
83 }  
84  
85 }
```

Identacion de código

```
68  
69  
70 ok=true;  
71  
72 }
```

```
68  
69  
70 ok=true;  
71  
72 }
```

Iteración de a no acorde con norma se pasa a m, tampoco hay llaves

```
88 public int Comprobar(int[] premi)  
89 {  
90     int a=0; // número de aciertos  
91     for (int i=0; i<MAX_NUMEROS; i++)  
92         for (int j=0; j<MAX_NUMEROS; j++)  
93             if (premi[i]==Nms[j]) a++;  
94     return a;  
95 }
```

```
88 1 referencia | alexquirros, hace 9 minutos | 1 autor, 2 cambios  
89 public int Comprobar(int[] premi)  
90 {  
91     int m=0; // número de aciertos  
92     for (int i=0; i<MAX_NUMEROS; i++)  
93     {  
94         for (int j = 0; j < MAX_NUMEROS; j++)  
95         {  
96             if (premi[i] == Nms[j])  
97                 m++;  
98         }  
99     }  
100     return m;  
101 }  
102 }
```

Refactorización código.

1-- Nombre Array poco descriptivo

```
13  
14 private int[] _nums = new int[MAX_NUMEROS]; // numeros de la combinación  
15 public bool ok = false; // combinación válida (si es aleatoria, siempre es válida, :  
16  
13  
14 private int[] listanumeros = new int[MAX_NUMEROS];  
15 public bool ok = false; // combinación válida (s:  
16
```

Nombres también poco descriptivos,

```
29
30 Random r = new Random(); // clase generadora
31
32 int i= 0, j, num;
33
34 do // generamos la c
35 {
36     num = r.Next(NUMERO_MENOR, N
37     for (j= 0; j<i; j++) // c
38         if (Nms[j]==num)
39             break;
40     if (i==j) // S
41     {
42         Nms[i]= num;
43         i++;
44     }
45 } while (i< MAX_NUMEROS);
46
47 validacion = true;
48
49
50
51 int i= 0, j, numero;
52
53 do // generamos la combinación
54 {
55     numero = r.Next(NUMERO_MENOR, NUMERO_MAYOR + 1); // generamos un nú
56     for (j= 0; j<i; j++) // comprobamos que el número no está
57         if (Nms[j]==numero)
58             break;
59     if (i==j) // Si i==j, el número no se ha encontrado en la
60     {
61         Nms[i]= numero;
62         i++;
63     }
64 } while (i< MAX_NUMEROS);
65
66 validacion = true;
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Código repetido extraemos método

```
1 referencia | alexquiross, Hace 41 minutos | 1 autor, 1 cambio
private void btValidar_Click(object sender, EventArgs e)
{
    int[] nums = new int[6];
    for (int i = 0; i < 6; i++)
        nums[i] = Convert.ToInt32(combinacion[i].Text);
    miLoto = new Loto(nums);
    if (miLoto.validacion)
        MessageBox.Show("Combinación válida");
    else
        MessageBox.Show("Combinación no válida");
}

1 referencia | alexquiross, Hace 41 minutos | 1 autor, 1 cambio
private void btComprobar_Click(object sender, EventArgs e)
{
    int[] nums = new int[6];
    for (int i = 0; i < 6; i++)
        nums[i] = Convert.ToInt32(combinacion[i].Text);
    miLoto = new Loto(nums);
    if (miLoto.validacion)
    {
        nums = new int[6];
        for (int i = 0; i < 6; i++)
            nums[i] = Convert.ToInt32(combinacion[i].Text);
        int cantidad = miLoto.comprobar(nums);
    }
}

2 referencias | 0 cambios | 0 autores, 0 cambios
private int[] Convertir()
{
    int[] nums = new int[6];
    for (int i = 0; i < 6; i++)
        nums[i] = Convert.ToInt32(combinacion[i].Text);
    return nums;
}

1 referencia | alexquiross, Hace 44 minutos | 1 autor, 1 cambio
private void btComprobar_Click(object sender, EventArgs e)
{
    int[] nums = Convertir();
    miLoto = new Loto(nums);
    if (miLoto.validacion)
    {
        nums = new int[6];
    }
}
```

Caja negra

Números del array

1. < 0 clase no valida.
2. 0 && =6 clase valida.
3. +6 clase no valida.
4. Array vacio.

Números loto

1. < 1 clase no valida.
2. 1 && 49 clase valida.
1. +49 clase no valida

Array	Valor introducido	V esperado	resultado
Array null	0	0	Excepción
Array menos valores	4	0	Excepcion
Array entre 0y 6 valores	6	6	valida
Array + 6 valores	7	0	Excepcion
Valores de loteria	Valor introducido	V esperado	resultado
Numero menor a 1	-1	0	Excepción
Numero entre 1 && 49	35	35	Excepcion
Numero > a 50	51	0	Excepción

.