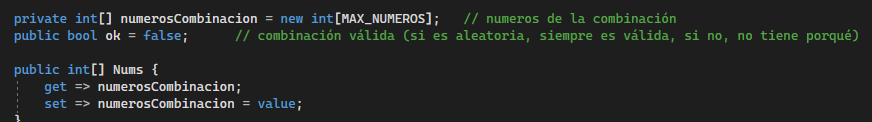
# Informe Examen segunda evaluación Entornos de desarrollo

## Errores normas de estilo

* La definición del campo “\_nums” en la línea 14 es un nombre poco descriptivo y no debería empezar por guion bajo. **Se ha cambiado a la notación camelCase con un nombre más descriptivo (“numerosCombinacion”)**.



* La definición de la variable *booleana* “ok” en la línea 15 es muy poco descriptiva. **Se le ha dado un nombre más descriptivo en notación camelCase (“esValida”)**.



* En la línea 17, el nombre de la propiedad “Nums” no es lo suficientemente descriptivo. **Se le ha dado un nombre más descriptivo en notación PasCal (“Combinacion”)**. *Se había nombrado “Premiados” previamente, pero se ha cambiado posteriormente*.



* En la línea 26, se ha cambiado el nombre de la variable aleatoria “r”. **Se le ha dado un nombre más descriptivo en notación camelCase (numeroAleatorio)**.



* La definición de la variable “num” en la línea 28 es poco descriptiva. **Se le ha dado un nombre más descriptivo en notación camelCase (numero)**.



* Se ha definido una variable por línea, pasando a tener tres líneas distintas a partir de la línea 28. Además, se han añadido espacios entre “i”, “=” y “0”.



* En la línea 50, el parámetro “misnums” no sigue las reglas de estilo ya que está todo en minúsculas y no tiene un nombre suficientemente descriptivo. **Se le ha dado un nombre más descriptivo en notación camelCase (“misNumeros”)**.



* En la línea 76, el método “comprobar” no tiene un formato adecuado y el nombre podría ser más descriptivo. **Se le ha dado un nombre más descriptivo en notación PasCal (“ComprobarAciertos”)**.



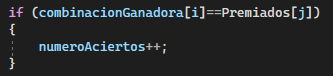
* En la misma línea 76, el nombre del *array* (“premi”) que se pasa por parámetro es poco descriptivo. **Se le ha dado un nombre más descriptivo en notación camelCase (“combinacionGanadora”)**.



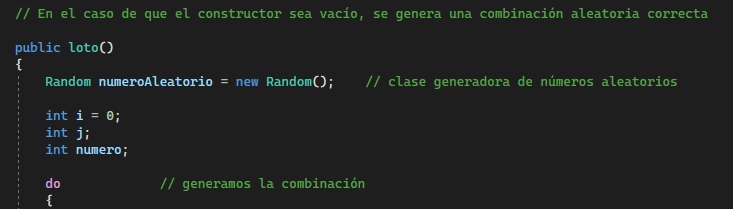
* En la línea 78, la variable que cuenta el número de aciertos (“a”) tiene un nombre genérico y muy poco descriptivo. **Se le ha dado un nombre más descriptivo en notación camelCase (“numeroAciertos”)**.

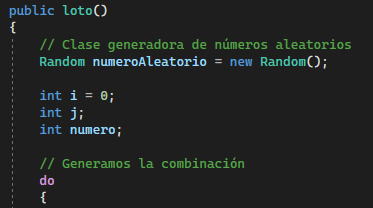


* En la línea 81, se ha añadido un salto de línea junto con llaves tras la sentencia “if” ya que aparecía todo en la misma línea lo cual no es correcto según las reglas de estilo.



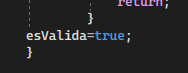
* Se han realizado correcciones también en los comentarios, eliminándose los que se encuentran en la misma línea que el código lo cual no es recomendable. Un ejemplo es las líneas 14, 15, 27 o la línea 33. A continuación se puede ver el cambio en estas dos últimas líneas.



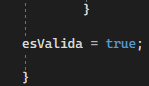


* Además, se han corregido otros errores generales que se repetían en el código como falta de espacios o saltos de línea como en la línea 70.

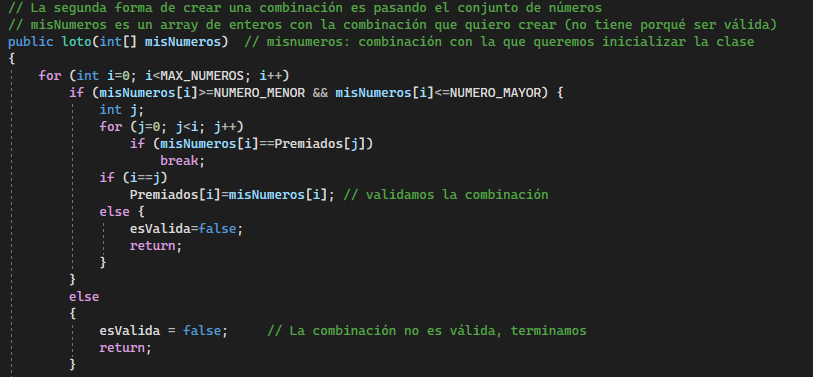
Pasando de esto:



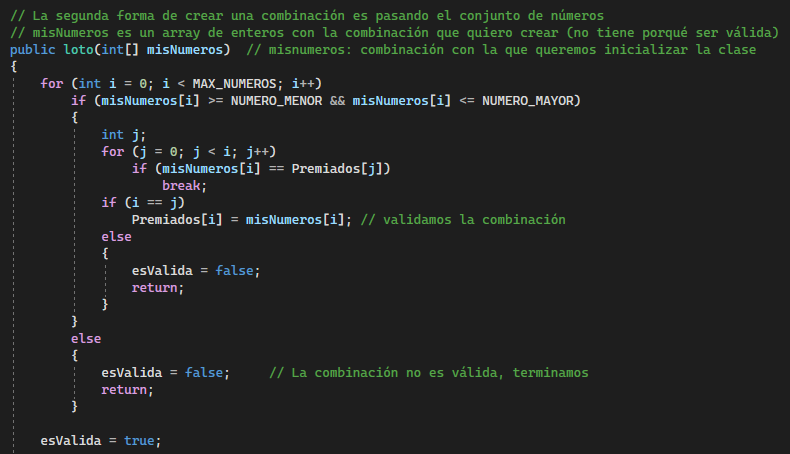
A esto:



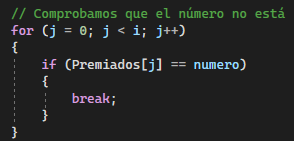
O de esto:



A esto (usando el comando: CTRL + K + D)

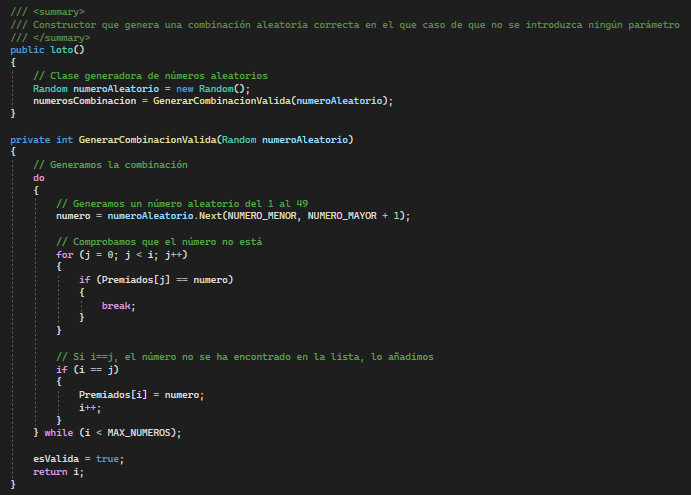


* También se han añadido llaves de apertura y cierre para ciertos bucles como el *for* que contiene dentro un *if* en la línea 41 como se puede ver a continuación.



## Patrones de refactorización

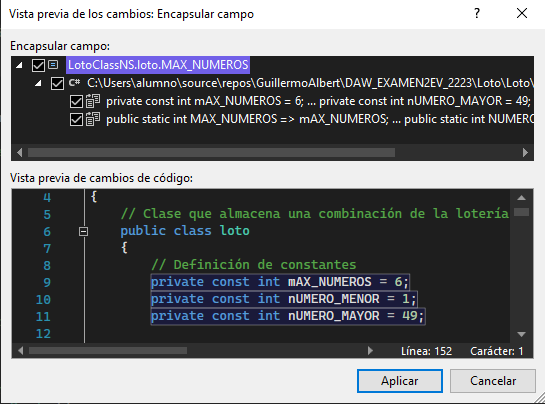
En la clase “Loto.cs” **se ha extraído el método para generar una combinación aleatoria correcta que se encontraba dentro del constructor vacío**. Esto se puede hacer con Visual Studio en la opción “Refactorizar” desde el menú superior. El nuevo método se ha nombrado “GenerarCombinacionValida” y se puede ver a continuación. También se han eliminado las variables “i”, “j” y “numero” ya que no eran necesarias. Finalmente, se guarda la combinación generada por el método en la variable “numerosCombinacion”.



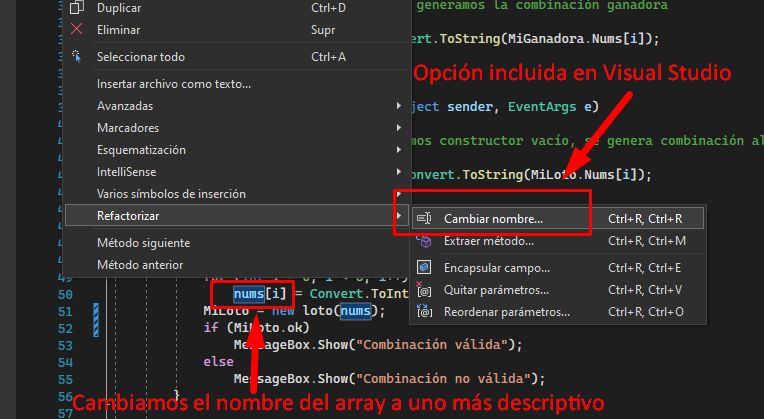
También se ha extraído el método del constructor con parámetros de la clase “Loto.cs” para comprobar la validez de los números introducidos por parámetro en el array. El método se ha llamado “ComprobarValidez”. En el constructor se comprueba si la combinación es válida. Si lo es, se asigna el valor a la combinación. Si no lo es, se lanza una excepción. Se puede ver a continuación como queda el código.



Otro error que se ha corregido el hecho de encapsular los campos de clase que están definidos como públicos cuando deberían ser privados. Esto se ha corregido en ambas clases. Encapsular un campo es una función incluida en el propio Visual Studio como se puede ver a continuación.



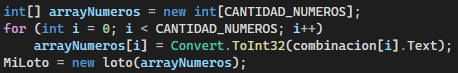
En la clase “Form1”, ahora renombrada como “Examen2EVGAG2223.cs”, se ha corregido el nombre de para hacerlas más descriptivas (opción incluida en Visual Studio). Esto ya se había realizado en la clase “Loto.cs”. El array “nums” se ha renombrado como “arrayNumeros”.



Otra de las cosas que se ha arreglado son los números mágicos. En la clase “Examen2EVGAG2223.cs”, se usa mucho el número 6 para indicar que el array de números tiene esta longitud. Sin embargo, sería conveniente reemplazarlo por una constante para mejorar la legibilidad y el mantenimiento del código. Esta opción no está implementada directamente en Visual Studio. Para ello definimos una constante “CANTIDAD\_NUMEROS” y se le asigna el valor 6. Luego reemplazamos en el código de toda la clase el número por la constante.



Aquí muestro un ejemplo de la implementación de la constante en el código de la clase.



## Diseño de pruebas (caja negra)

Para realizar el diseño de pruebas para el constructor con parámetro de la clase loto, hay que determinar las clases de equivalencia y los valores límite. Sabemos que para que la combinación introducida sea válida, debe ser una combinación de entre 1 y 6 números enteros que se encuentren entre 1 y 49. Estas son las clases de equivalencia diseñadas:

Clase A1. Array de caracteres no númericos. No válido.

Clase A2. Array de enteros.

Clase A2.1. Array de enteros con un tamaño mayor que 6. No válido.

Clase A2.2. Array de enteros no nulo con un tamaño menor que 6.

Clase A2.2.1. Array de enteros con algún valor inferior a 1 o superior a 49, no vacío y con un tamaño menor que 6. No válido.

Clase A2.2.2. Array de enteros con valores comprendidos entre 1 y 49, no vacío y con un tamaño menor que 6. **Válido**.

Los valores límite serían:

Array vacío.

Array con 6 números.

Array con 1 número.

Array con algún valor igual a 1.

Array con algún valor igual a 49.

Array con algún valor igual a 0.

Array con algún valor igual a 50.