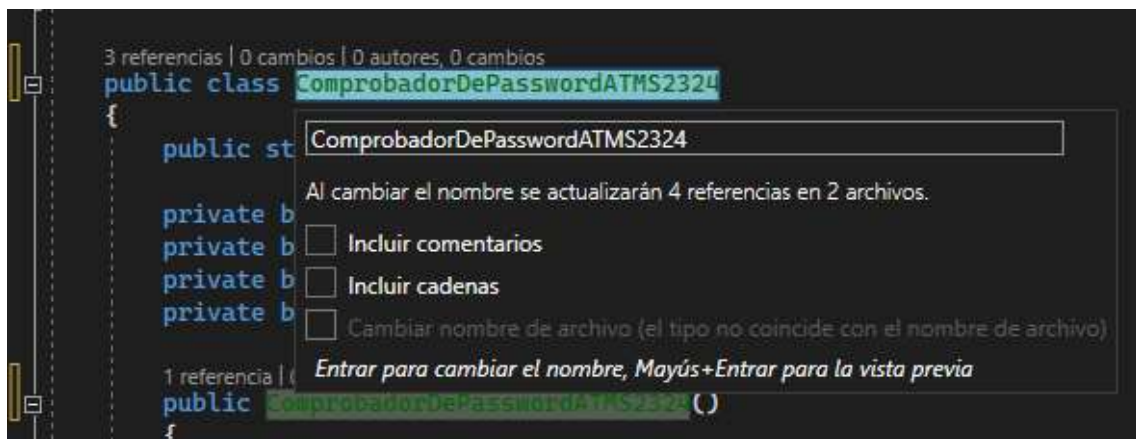


## DOSSIER – EXAMEN 2ª EVALUACIÓN ENTORNOS DE DESARROLLO

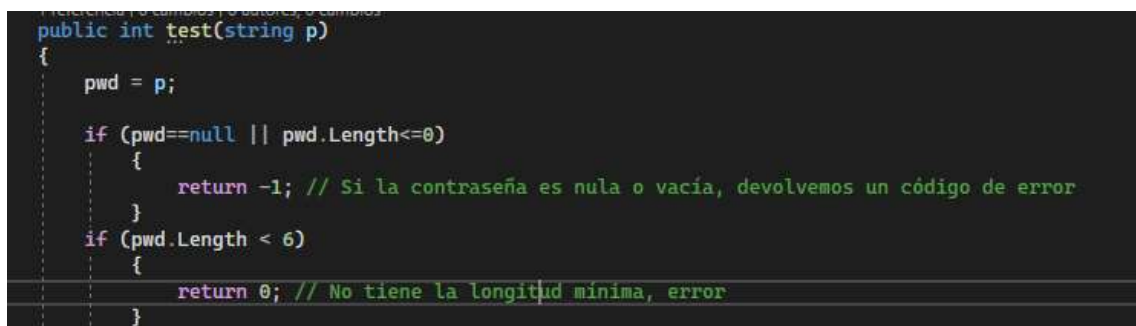
### 1.- CINCO ERRORES DE NORMAS DE ESTILO CLASE `comprobadorDePasswordATMS2324`

1.1 - Poner el nombre de la clase y del constructor en PasCal y en mayúsculas el nombre del método `Test()`.

*Sí me funciona Refactorizar-Cambiar nombre en Visual Studio.*

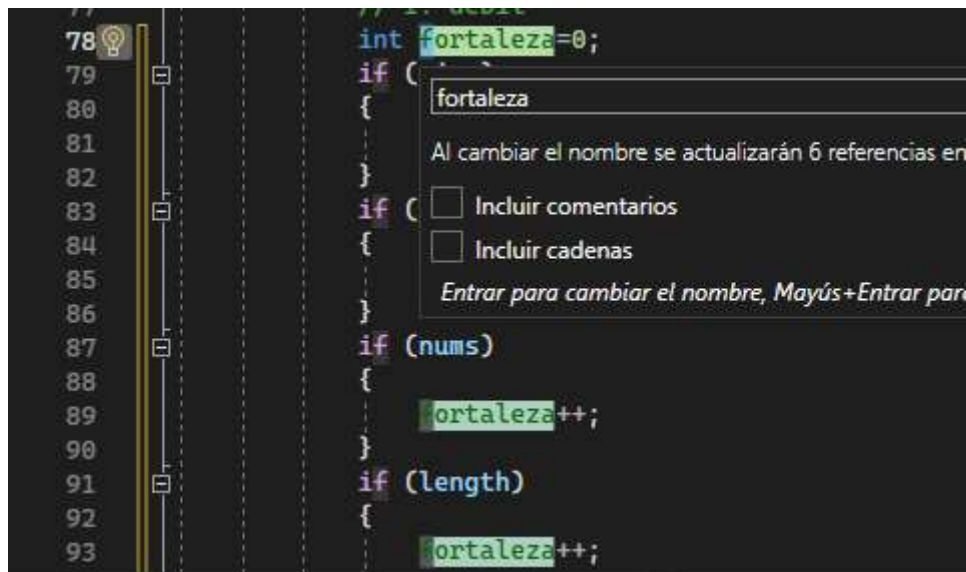


1.2 - Establecer llaves delimitando los bloques `if`, línea 30. Lo mismo sucede en la línea 40 y 76 y siguientes.



1.3 - Nombres de los campos del método poco descriptivos (líneas 14 a 19) y la variable f en la línea 78.

*Sí me funciona Refactorizar-Cambiar nombre en Visual Studio.*



1.4 - En el constructor de la clase, asignar el valor a cada variable en una línea diferente (línea 23).

```
public ComprobadorDePasswordATMS2324()
{
    minusculas = false;
    mayusculas = false;
    numeros = false;
    longitud = false;
}
```

1.5 - Establecer espacios entre operadores y variables (línea 33, 58 y 65).

```
if (password == null || password.Length <= 0)
{
```

## 2.- DISEÑO DE PRUEBAS DE CAJA NEGRA para el método `comprobadorDePassword.test()`

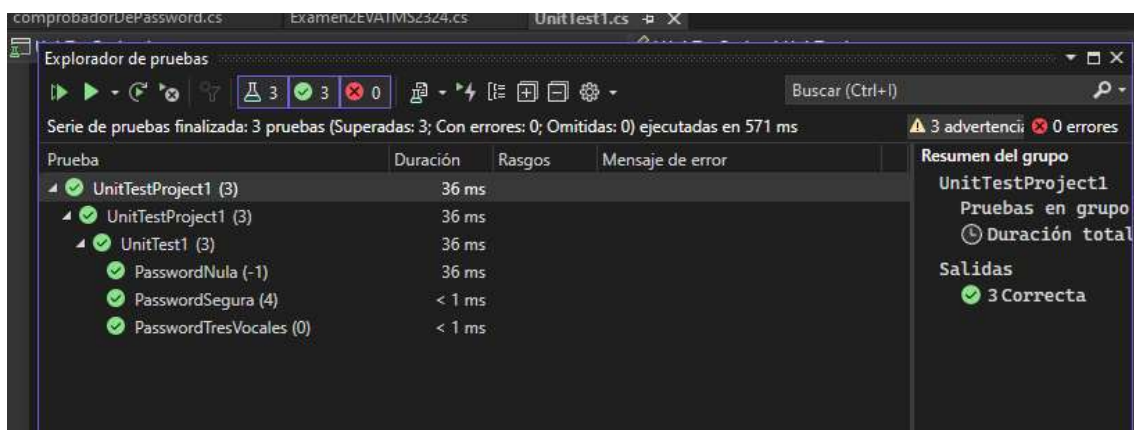
Contraseña	Devuelve	Contraseña
Nula o vacía	-1	
Menor de 6 caracteres	0	
6 caracteres todos minúsculas	1	Débil
6 caracteres: 5 minúsculas y 1 mayúscula o número	2	Normal
6 caracteres: 4 minúsculas y 2 mayúsculas	2	Normal
6 caracteres: 4 minúsculas y 2 números	2	Normal
6 caracteres: 4 minúsculas y 1 mayúscula y 1 número	3	Fuerte
6 caracteres: 3 minúsculas y 2 mayúsculas y 1 número	3	Fuerte
6 caracteres: 3 minúsculas y 1 mayúscula y 2 números	3	Fuerte
6 caracteres: 3 minúsculas y 2 mayúsculas y 2 números	3	Fuerte
Más de 12 caracteres: 12 minúsculas y 1 mayúscula	3	Fuerte
Más de 12 caracteres: 12 minúsculas y 1 mayúsculas y 1 número	3	Fuerte
Más de 12 caracteres: 12 minúsculas, 1 mayúscula y 1 número	4	Muy fuerte

## 3.- Crear los métodos de prueba que correspondan a los siguientes valores de prueba: "", "abc", "C0ntr@s3ñ@S3gur@"

(Commit)

Nombre del commit: **Pruebas antes de refactorizar**

Antes de refactorizar

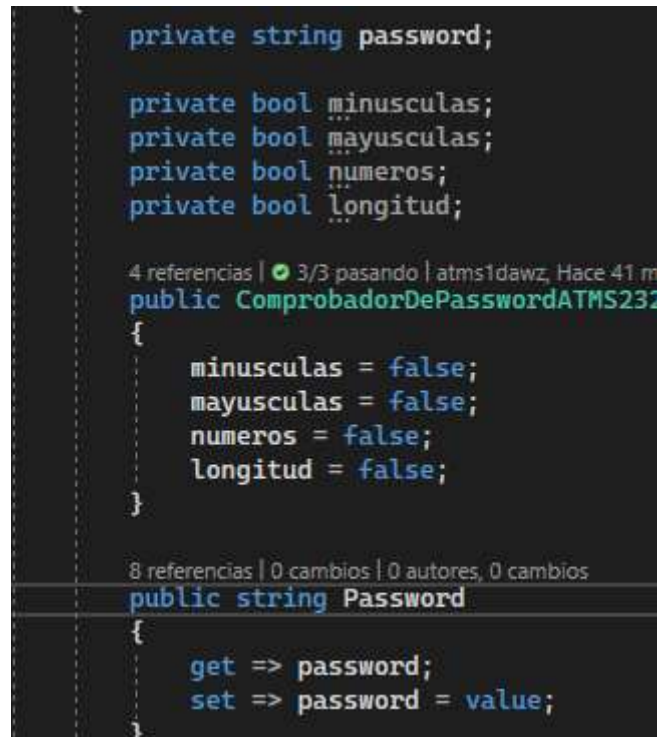


#### 4.-TRES PATRONES DE REFACTORIZACIÓN en el fichero comprobadorDePassword.cs

##### 4.1 – ENCAPSULAR CAMPO password (línea 14)

Lo realizo con Visual Studio: Menú: Editar-Refactorizar

Tras realizar esta refactorización las pruebas funcionan sin necesidad de realizar ninguna modificación en éstas.



```
private string password;

private bool minusculas;
private bool mayusculas;
private bool numeros;
private bool longitud;

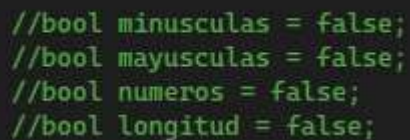
4 referencias | 3/3 pasando | atms1dawz, Hace 41 m
public ComprobadorDePasswordATMS232
{
    minusculas = false;
    mayusculas = false;
    numeros = false;
    longitud = false;
}

8 referencias | 0 cambios | 0 autores, 0 cambios
public string Password
{
    get => password;
    set => password = value;
}
```

##### 4.2 – CODIGO REPETIDO (líneas 48 a 51)

En estas líneas se repite la definición y asignación de valores a los campos de la clase; las comento.

Tras realizar esta refactorización las pruebas funcionan sin necesidad de realizar ninguna modificación en éstas.



```
//bool minusculas = false;
//bool mayusculas = false;
//bool numeros = false;
//bool longitud = false;
```

### 4.3 – NUMEROS MÁGICOS

Creo en el constructor dos constantes de tipo entero y asigno el valor -1 a la constante que representará la contraseña nula o cadena vacía y 0 para la cadena con un número inferior al número mínimo de caracteres (línea 15 y 16).

```
15 public const int passwordNula = -1;
16 public const int passwordInferior = 0;
```

Y establezco estas constantes en la primera comprobación que se hace en el método Test() (línea 41).

```
4 referencias | 3/3 pasando | 0 cambios | 0 autores, 0 cambios
public int Test(string p)
{
    Password = p;

    if (Password == null || Password.Length <= 0)
    {
        return passwordNula; // Si la contraseña es nula o vacía, devolvemos un código de error
    }
    if (Password.Length < 6)
    {
        return passwordInferior; // No tiene la longitud mínima, error
    }
}
```

Tras realizar esta refactorización las pruebas funcionan sin necesidad de realizar ninguna modificación en éstas.

### 5.- MODIFICAR LOS MÉTODOS DE PRUEBA

Tras realizar esta refactorización las pruebas funcionan sin necesidad de realizar ninguna modificación en éstas.

The screenshot shows the Visual Studio Test Explorer window for the project 'comprobadorDePassword.cs'. The 'Explorador de pruebas' (Test Explorer) pane displays the test results. At the top, it shows 'Serie de pruebas finalizada: 3 pruebas (Superadas: 3; Con errores: 0; Omitidas: 0) ejecutadas en 560 ms'. Below this, a table lists the tests and their results.

Prueba	Duración	Rasgos	Mensaje de error
✔ UnitTestProject1 (3)	36 ms		
✔ UnitTestProject1 (3)	36 ms		
✔ UnitTest1 (3)	36 ms		
✔ PasswordNula (-1)	36 ms		
✔ PasswordSegura (4)	< 1 ms		
✔ PasswordTresVocales (0)	< 1 ms		

## 6.- DOCUMENTAR EL FICHERO comprobadorDePassword.cs

*(Commit)*

*Nombre del commit:* **Documentar el fichero**