

Universidad Mariano Gálvez Guatemala, Campus

Jutiapa

Facultad de Ingeniería en Sistemas

Curso: Programación 1



Catedrático: Ing. Ruldin Ayala

Nombre del estudiante: Lester David Payes Méndez

Carné: 0905-24.22750

Bonus Challenge:

LLM SELECCIONADOS: CHAT GPT (GPT-4) Y DeepSeek LLM

Tipo de Licencia y validación:

Dado que la primera cosa que decían las instrucciones que tenemos que hacer es validar, lo hice de la siguiente forma:

```
public Chofer(string Name, int LaEdad, string LicenciaT)
{
    ValidarEdadLicencia(LaEdad, LicenciaT);

    Nombre = Name ?? throw new ArgumentNullException(nameof(Name));
    Edad = LaEdad;
    TipoLicencia = LicenciaT ?? throw new ArgumentNullException(nameof(LicenciaT));
}
```

En el constructor de llama a la función que cree encargada de realizar lo que me piden

```
private void ValidarEdadLicencia(int edad, string tipoLicencia)
{
    if ((tipoLicencia == "M" || tipoLicencia == "C") && edad < 16)
    {
        throw new ArgumentException("Para las licencias tipo M y C, la edad mínima es 16 años.");
    }
    if (tipoLicencia == "B" && edad < 18)
    {
        throw new ArgumentException("Para la licencia tipo B, la edad mínima es 18 años.");
    }
    if (tipoLicencia == "A" && edad < 23)
    {
        throw new ArgumentException("Para la licencia tipo A, la edad mínima es 23 años.");
    }
}
```

Dado que, si después se cambia el tipo de licencia de la persona, este código no cubre la validación nuevamente. La validación de la edad y el tipo de licencia solo se realiza en el constructor, lo que significa que cualquier cambio posterior en las propiedades `Edad` o `TipoLicencia` no será validado. Esto puede llevar a inconsistencias y errores en el programa.

Consultar con LLM:

- **Creación del Prompt**

Estoy trabajando en una clase `Chofer` en C# que tiene una propiedad `TipoLicencia`. En el constructor de la clase, se valida que el tipo de licencia esté en el rango de edad adecuado. Sin embargo, necesito asegurarme de que, si en el transcurso del programa se cambia la propiedad `TipoLicencia`, se vuelva a validar que la edad del chofer sea adecuada para el nuevo tipo de licencia. Actualmente, el código no cubre esta situación, ya que no hay ningún método que valide la edad cuando se cambia la propiedad `TipoLicencia` después de la creación del objeto. ¿Cuál es la mejor práctica para implementar esta validación en C#?

- **CHAT GPT-4**

Procedo a Darle el Código con el siguiente prompt: Hola, acá te dejaré un código de una clase llamada `Chofer`. Lo que necesito es que analices el código y luego esperes mis instrucciones es decir no hagas nada por el momento.

Luego le mandé el prompt al LLM y me devolvió una solución que me pareció buena, ya que antes el código solo validaba la edad en el momento de la creación del objeto `Chofer`, pero no se hacía ninguna validación si después se cambiaba el tipo de licencia. Esto podría generar inconsistencias si un chofer cambiaba su licencia a un tipo que no fuera adecuado para su edad.



```
using System;

public class Chofer {
    public int Edad { get; set; }

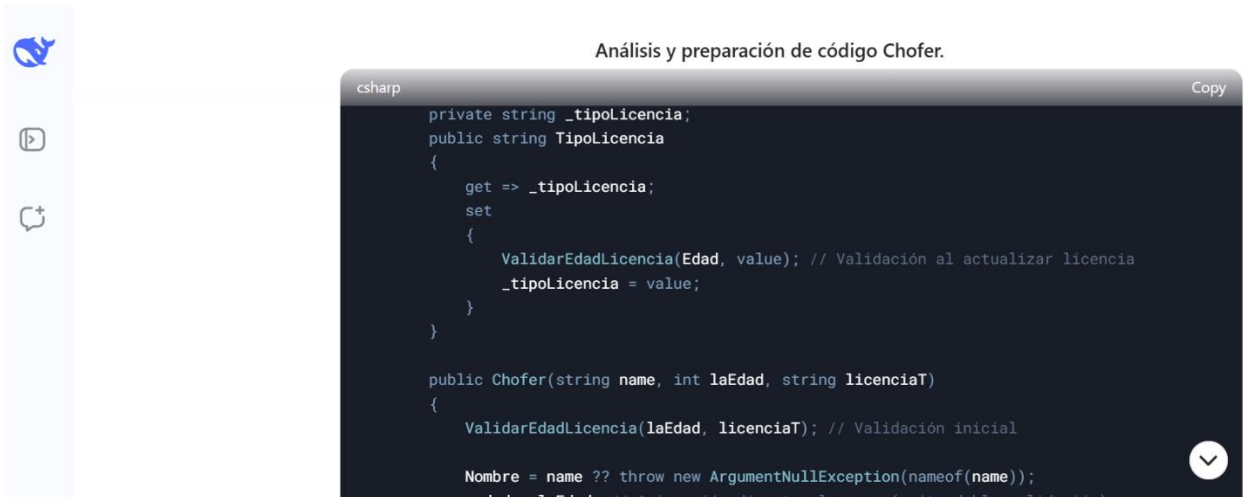
    private string tipoLicencia;
    public string TipoLicencia
    {
        get => tipoLicencia;
        set
        {
            ValidarEdadLicencia(Edad, value);
            tipoLicencia = value;
        }
    }
}
```

Ahora, con la nueva solución, he agregado una validación dentro del set de la propiedad TipoLicencia. Esto significa que, cada vez que intento cambiar el tipo de licencia del chofer, el código automáticamente valida si la nueva licencia es compatible con la edad del chofer. Esto me asegura de que no se pueda asignar un tipo de licencia inválido después de la creación del objeto, lo que mejora la fiabilidad y la consistencia del programa.

• DEEPSEEK

Procedo a Darle el Código con el siguiente prompt: Hola, acá te dejaré un código de una clase llamada Chofer. Lo que necesito es que analices el código y luego esperes mis instrucciones es decir no hagas nada por el momento.

Luego le mandé el prompt al LLM y me devolvió una solución que me pareció buena, al igual que chat gpt me brindo la misma solución y al probarlo si funciona correctamente



```
private string _tipoLicencia;
public string TipoLicencia
{
    get => _tipoLicencia;
    set
    {
        ValidarEdadLicencia(Edad, value); // Validación al actualizar licencia
        _tipoLicencia = value;
    }
}

public Chofer(string name, int laEdad, string licenciaT)
{
    ValidarEdadLicencia(laEdad, licenciaT); // Validación inicial

    Nombre = name ?? throw new ArgumentNullException(nameof(name));
    edad = laEdad; // asignación directa al campo (otra doble validación)
```

Conclusión:

Ya que ambas me brindaron la misma solución recomendando usar ambas, pero por rapidez y también la forma de explicar código es mejor DeepSeek en mi opinión

Código final de Chofer

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace plbpoo.MisClases
{
    9 referencias
    internal class Chofer : IPiloto
    {
        4 referencias
        public string Nombre { get; set; }
        3 referencias
        public int Edad { get; set; }

        private string tipoLicencia;
        3 referencias
        public string TipoLicencia
        {
            get => tipoLicencia;
            set
            {
                ValidarEdadLicencia(Edad, value);
                tipoLicencia = value;
            }
        }

        3 referencias
        public Chofer(string name, int laEdad, string licenciaT)
        {
            ValidarEdadLicencia(laEdad, licenciaT);

            Nombre = name ?? throw new ArgumentNullException(nameof(name));
            Edad = laEdad;
            tipoLicencia = licenciaT ?? throw new ArgumentNullException(nameof(licenciaT));
        }

        2 referencias
        private void ValidarEdadLicencia(int edad, string tipoLicencia)
        {
            if ((tipoLicencia == "M" || tipoLicencia == "C" && edad < 16)
            {
                throw new ArgumentException("Para las licencias tipo M y C, la edad mínima es 16 años.");
            }
            if (tipoLicencia == "B" && edad < 18)
            {
                throw new ArgumentException("Para la licencia tipo B, la edad mínima es 18 años.");
            }
            if (tipoLicencia == "A" && edad < 23)
            {
                throw new ArgumentException("Para la licencia tipo A, la edad mínima es 23 años.");
            }
        }

        1 referencia
        public void mostrarInformación()
        {
            Console.WriteLine($"El piloto es {Nombre}");
            Console.WriteLine($"Licencia tipo {TipoLicencia}");
        }
    }
}
```