TEAM#4: **8/10**

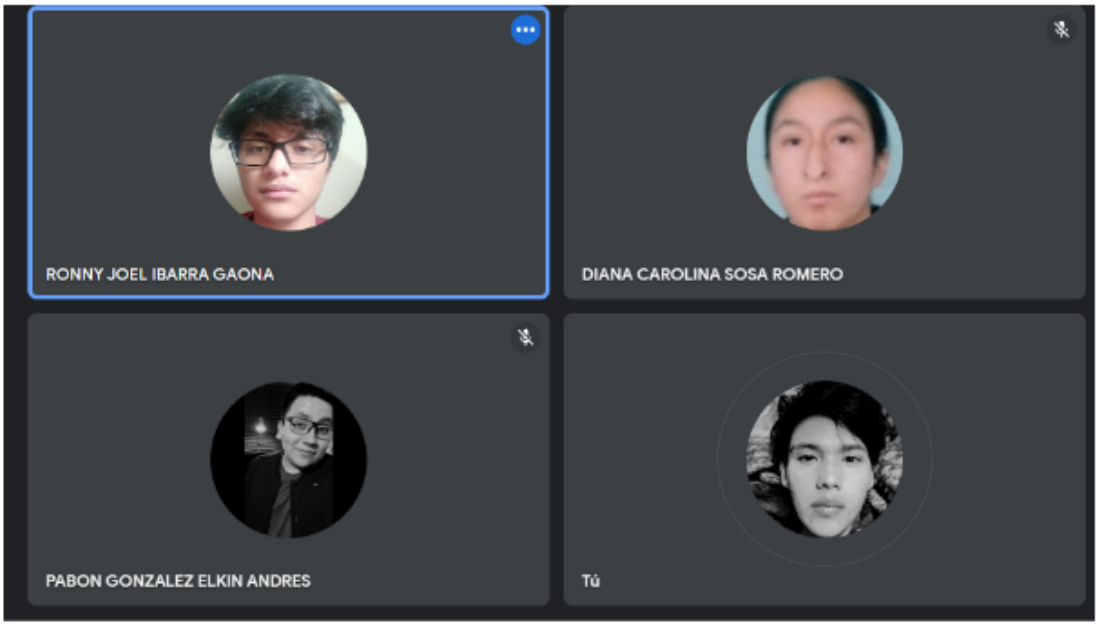
Caiza Pullotasig Widinson Danilo 4

Ibarra Gaona Ronny Joel 4

Pabon Gonzalez Elkin Andres 4

Sosa Romero Diana Carolina 4

Meet:



¿Qué es Pitfall?

> "Pitfall" es un conjunto de malos hábitos o prácticas realizadas por un programador al momento de escribir su código, causando que sea confuso o difícil de comprender y que a la larga cause problemas en su mantenimiento o revisión por terceros.

Description

-Long Message Chain

Descripción del problema:

Un objeto tiene que delegar una llamada de método a un objeto contenido es decir es redundante, que a su vez tiene que delegar la llamada de método a un objeto que contiene, y así sucesivamente. Esto puede ocurrir

cuando el patrón decorador o una relación de composición es recursiva y se usa incorrectamente.

Consecuencias:

Bajo rendimiento gasto innecesario

Al momento de ejecutar el código

Complejidad accidental en el flujo de control en tiempo de ejecución.

Causas:

Uso inapropiado del patrón decorador o abuso de cualquier relación de composición recursiva en el código.

Evitar:

Lo primero que debemos hacer es razonar exhaustivamente en cómo se van a utilizar los decoradores o las composiciones recursivas y si estas se van a producir con largas frecuencias de cadenas de mensajes.

Reconocimiento:

Para hacer un buen reconocimiento debemos tener en cuenta los siguiente:

- Recorridos en inspecciones de diseño/código
- Prueba de rendimiento en el tiempo de ejecución y sus puntos de referencia.

Liberación:

Para la liberación lo que debemos de hacer es refactorizar(hacer un cambio en la estructura interna del software para hacerlo más fácil), también se puede utilizar estrategias, métodos de plantilla u otros patrones de diseño que no darán como resultado largas cadenas de mensajes y limitar la profundidad de la composición para los decoradores o relaciones de composición recursiva

Ejemplo

```
payrollService.GetDataForMonth(2022,
5).GenerateReport().ConvertToSpreadsheet().Print()
;
```

El ejemplo anterior contiene cuatro llamadas encadenadas. La solución es aplicar una refactorización llamada ocultar al delegado. Esta refactorización consiste en ocultar todas las llamadas bajo un único método. Por lo tanto, al aplicar esta refactorización, podría convertir nuestro primer ejemplo de C# en lo siguiente:

```
payrollService.PrintSpreadsheetForMonth(2022, 5);
```