¿Qué es un Coding Dojo?Un Coding Dojo es una reunión donde un grupo de personas que son programadores, que es donde se reúnen para trabajar en una programación.

2. **Diferencia entre Requerimientos, Criterios de Aceptación y Escenarios de Prueba. Dar ejemplos a partir de un problema distinto a la referencia. Referencia**
https://lorenzosolano.com/requirements-acceptance-criteria-and/

Los Criterios de Aceptación son especificaciones que se usan para asegurar que los requisitos sean verificables. Estos criterios establecen si el sistema de software ha implementado adecuadamente el requisito.

3. **De dos ejemplos de requerimientos no-funcionales, y especifique cuál es su categoría (seguridad, performance, portabilidad, etc.)**

1. **Rendimiento**: Esto es lo que capacita una experiencia fluida y una respuesta más rápida del sistema.**Categoría:**Rendimiento

2. **Seguridad:** El sistema debe ser capaz de cumplir los estándares de seguridad de la industria y para garantizar la confiabilidad de la información del usuario. **Categoría:**seguridad

4. ¿Qué es TDD?
Es una práctica de ingeniería de software que involucra otras dos prácticas: Escribir las pruebas primero y Refactorización. Para escribir las pruebas generalmente se utilizan las pruebas unitarias.

5. ¿Qué son pruebas unitarias automatizadas?
Las pruebas unitarias automatizadas son un componente fundamental en el desarrollo de software moderno.

6. ¿Cuál fue el 1er framework de pruebas unitarias y para cual lenguaje fue creado?

El primer framework de de pruebas unitarias fue" SUnit", desarrollado por Kent Beck para el lenguaje de programación Smalltalk. SUnit se creó en la década de 1990 como parte de la metodología de desarrollo Extreme Programming(XP).

7. ¿Describa los componentes de la arquitectura xUnit?

Clases de pruebas: las pruebas se organizan en clases de pruebas.

Métodos de prueba: Estos métodos contienen la lógica de la prueba en sí.

Fixture de pruebas: Los fixtures son componentes que se utilizan para establecer un contexto común para un conjunto de pruebas.

Runner (ejecutor): El ejecutor es responsable de descubrir, cargar y ejecutar las pruebas.

Afirmaciones (Assertions): Las afirmaciones son expresiones que verifican si una condición es verdadera o falsa.
Resultados de las pruebas: Después de ejecutar las pruebas, se generan informes que muestran el resultado de cada prueba.

8. Indique al menos tres desventajas de las pruebas unitarias automatizadas

Costo de mantenimiento
Tiempo de desarrollo inicial
Posible falsa sensación de seguridad

9. Indique al menos tres ventajas de las pruebas unitarias automatizadas.

Retroalimentación inmediata
Facilitan la refactorización
Aumentan la confianza en el código

10. Tomando el algoritmo de conversión de números arábigos o "decimales" a números Romanos:
*** Cree un documento donde se listen los Requerimientos, Criterios de Aceptación y Casos de Prueba para una aplicación de consola**

*** Los casos de prueba deben ser de dos categorías: End-To-End (desde el UI) y Unitarios (caja blanca, código, bajo nivel)**

Requerimientos:
La app debe aceptar como entrada el número arábigos o "decimales".
La app debe convertir el número arábigos o "decimales" a número romano.
La app debe mostrar como salida el número romano.

Criterios de aceptación:
Puesto un número arábigo, el usuario debe ingresar el número y cuando al app debe mostrar al final el número Romano que ingresaste.

Casos de prueba:
End-To-End: El usuario inicia el programa, ingresa el número arábigo y se verifica que el programa muestre el número romano.
Unitarios: Se prueba la función del programa con varios números arábigos para verificar si los números romanos son correctos.

Entrada:5

Convertidor.NúmDecimal(INT):string

Salida:V

11. Utilizando el lenguaje de su preferencia cree cinco o más casos de prueba unitarios automatizados utilizando un framework de automatización de pruebas para el algoritmo en cuestión