Hernán I. Fernández Castiglione
Carné: 2022086706
Fecha: 02/23/2025

Preguntas Teóricas

1. **¿Explique la principal utilidad de Git como herramienta de desarrollo de código?**

Git es un sistema de control de versiones que permite gestionar el historial de cambios en un proyecto de software. Facilita la colaboración entre desarrolladores, permitiendo trabajar en paralelo, rastrear cambios, revertir errores y mantener múltiples versiones del código.

2. **¿Qué es un branch?**

Un *branch* (rama) en Git es una línea de desarrollo separada del historial principal del proyecto. Permite trabajar en nuevas características o correcciones sin afectar el código principal hasta que los cambios sean aprobados y fusionados.

3. **En el contexto de GitHub, ¿qué es un Pull Request?**

Un *Pull Request* (PR) es una solicitud para fusionar cambios de una rama a otra dentro de un repositorio en GitHub. Permite revisar y discutir cambios antes de integrarlos en la rama principal (*main* o *master*).

4. **¿Qué es un commit?**

Un *commit* es un registro de cambios en el repositorio. Representa una instantánea del estado actual del código en un momento dado, con un mensaje descriptivo que ayuda a entender las modificaciones realizadas.

5. **Describa lo que sucede al ejecutar las siguientes operaciones: git fetch y git rebase origin/master**

    - git fetch: Descarga los últimos cambios del repositorio remoto sin aplicarlos a la rama actual.
    - git rebase origin/master: Reaplica los commits de la rama actual sobre la versión más reciente de la rama master, reescribiendo la historia para mantener un historial lineal.

6. **Explique qué es un “merge conflict” o “rebase conflict” en el contexto de tratar de hacer merge a un Pull Request o de completar una operación git rebase.**

Ocurre cuando Git no puede combinar automáticamente los cambios porque existen modificaciones incompatibles en el mismo archivo dentro de diferentes

ramas. Es necesario resolver manualmente los conflictos antes de completar el *merge* o *rebase*.

7. **¿Qué es una Prueba Unitaria o *Unittest* en el contexto de desarrollo de software?**

Es una técnica de pruebas en la que se evalúan unidades individuales de código (como funciones o métodos) de manera aislada, para asegurar que cumplen correctamente con su propósito.

8. **Bajo el contexto de *pytest*, ¿cuál es la utilidad de un assert?**

En *pytest*, assert se usa para verificar si una condición es verdadera. Si la condición es falsa, la prueba falla, lo que indica un posible error en el código.

9. **¿Qué es *Flake8*?**

Flake8 es una herramienta que analiza el código en busca de errores de estilo y sintaxis según las convenciones de Python (PEP 8). Ayuda a mejorar la calidad y legibilidad del código.

10. **Explique la funcionalidad de parametrización de *pytest*.**

La parametrización en *pytest* permite ejecutar una misma prueba varias veces con diferentes valores de entrada, lo que ayuda a evaluar múltiples casos sin duplicar código.