

## **Tarea 1**

### **GitHub, Pytest y Flake 8**

A continuación, se presentan 2 asignaciones para entender y aplicar herramientas de desarrollo de proyectos de programación.

#### **Preguntas Teóricas (20 pts, 2pts c/u)**

**1) ¿Explique la principal utilidad de git como herramienta de desarrollo de código?**

Git es un sistema de control de versiones distribuido cuya utilidad principal es gestionar de forma segura, trazable y colaborativa la evolución del código fuente, permitiendo registrar cambios, volver a estados anteriores y trabajar en paralelo sin perder integridad.

**2) Explique la diferencia entre git y github**

Git es un sistema de control de versiones que se ejecuta localmente y permite gestionar el historial del código (commits, branches, merges).

GitHub es una plataforma en la nube que aloja repositorios Git y añade herramientas de colaboración como control de acceso, revisión de código, issues y automatizació.

**3) ¿Qué es un branch?**

Un branch es una rama independiente del historial de un repositorio Git que permite desarrollar, probar o modificar código sin afectar la versión principal.

**4) En el contexto de github. ¿Qué es un Pull Request?**

Un Pull Request (PR) es una solicitud para integrar cambios de una rama a otra dentro de un repositorio en GitHub, normalmente hacia la rama principal.

Permite revisar, discutir, validar y aprobar el código antes de fusionarlo, asegurando calidad, trazabilidad y control en el desarrollo colaborativo.

**5) ¿Qué es un commit?**

Un commit es un registro permanente de un conjunto de cambios en el código dentro de un repositorio Git.

Cada commit guarda el estado del proyecto en un momento específico, con autor, fecha y mensaje descriptivo, permitiendo trazabilidad y recuperación de versiones.

**6) Describa lo que sucede al ejecutar la siguiente operación: “git rebase main”.**

git rebase main reaplica los commits de la rama actual sobre la punta de la rama main, reescribiendo el historial.

**Descripción breve del proceso**

Git toma los commits exclusivos de la rama actual

Mueve la base de esa rama al último commit de main.

Reaplica los commits uno por uno como si se hubieran creado después de main.

**7) Explique que es un “merge conflict” y como lo resolvería.**

Un merge conflict ocurre cuando Git no puede combinar automáticamente cambios porque dos ramas modifican las mismas líneas de un archivo de forma incompatible.

Cómo resolverlo

Git marca el archivo con delimitadores (<<<<<<, =====, >>>>>>) indicando las versiones en conflicto.

Revisar qué cambios provienen de cada rama y cuál es el comportamiento correcto esperado.

Decidir qué líneas conservar, modificar o combinar, y eliminar los marcadores de conflicto.

Compilar, simular o ejecutar pruebas (en verificación: correr el testbench o regresiones).

Guardar el archivo, marcarlo como resuelto (git add) y completar el proceso (git commit o git merge --continue).

Un merge conflict no es un error de Git, sino una decisión técnica que requiere criterio del desarrollador para mantener coherencia y funcionalidad del código.

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

Una Prueba Unitaria (Unit Test) es una prueba automatizada que valida el comportamiento correcto de la unidad más pequeña de software (función, método o módulo) de forma aislada. Verifica que cada componente funcione según lo esperado, detecta errores tempranos y facilita mantenimiento y refactorización sin romper el sistema.

**9) Bajo el contexto de pytest. ¿Cuál es la utilidad de un “assert”?**

En pytest, un assert se utiliza para verificar que una condición sea verdadera durante una prueba. Permite comparar el resultado obtenido con el resultado esperado; si la condición falla, la prueba se marca como fallida y pytest muestra automáticamente el valor esperado y el valor real, facilitando la detección del error.

**10) Mencione y explique tres errores de formato detectables con Flake8**

Líneas demasiado largas (E501): Ocurre cuando una línea de código supera el límite recomendado (usualmente 79 caracteres según PEP 8).

Espacios incorrectos alrededor de operadores (E225 / E231): Se detecta cuando faltan o sobran espacios alrededor de operadores (=, +, ,, etc.).

Importaciones no utilizadas (F401): Aparece cuando se importa un módulo o símbolo que no se usa en el archivo.