Sebastián Picado González (2018135190) y Milton Cárdenas Fley (2018109300)

# Tarea #1
# GitHub, Pytest y Flake 8

## Preguntas teóricas

1. Diferencie la herramienta Git de Github

   La diferencia principal entre Git y Github es que Git es una herramienta de código abierto que se instala localmente para el control de versiones del código fuente, el cual permite mantener un registro de todo el historial de cambios de los proyectos en los que se trabaja; mientras que Github es un servicio en línea al que los usuarios que utilizan Git pueden acceder para cargar o descargar recursos, por lo que funciona como un servicio de alojamiento para los repositorios de Git.

2. ¿Qué es un branch?

   Un branch (rama) es el acto de cambiar la ejecución a una secuencia de instrucción diferente como resultado de ejecutar una instrucción de bifurcación. En otras palabras, un branch genera ramificaciones dentro del código lo que permite que en un instante se esté ejecutando una secuencia de código y en otro instante se esté ejecutando otra secuencia, la cual no afectará a la primera.

   Un branch en la herramienta Git puede ser visto como un apuntador hacia un commit.

3. ¿Qué es un commit?

   Un commit (confirmación) es un comando que se utiliza en Git para guardar los cambios que se han realizado en ese momento del proyecto. Consiste, básicamente, en una actualización de los archivos.

4. ¿Qué es la operación cherry-pick?

   Cherry-pick es un comando el cual permite seleccionar commits individuales de cualquier branch y añadirlas al actual branch HEAD de trabajo. En otras palabras, la ejecución de cherry-pick es el acto de elegir una commit de una branch y aplicarla a otra.

A pesar de ser una herramienta muy útil, esta puede generar duplicaciones de commits, por lo que, incluso en casos donde se puede usar cherry-pick es común que se elijan métodos tradicionales de fusión.

**5.** ¿Qué hace el comando git stash?

El comando git stash almacena temporalmente los cambios que se hayan efectuado en el código en el que se está trabajando para que el usuario se pueda dedicar a otra tarea y, posteriormente, pueda regresar a dicho código sin que se hayan perdido los cambios.

El comando resulta especialmente práctico cuando se está trabajando en un código y se requiere mover rápidamente de tarea, pero los cambios que se han realizado en el código no se consideran listos para guardarlos con un commit.

**6.** Compare las operaciones git fetch y git pull

Primero que todo, ambos comandos son utilizados para descargar contenido desde un repositorio remoto; sin embargo, presentan sus diferencias.

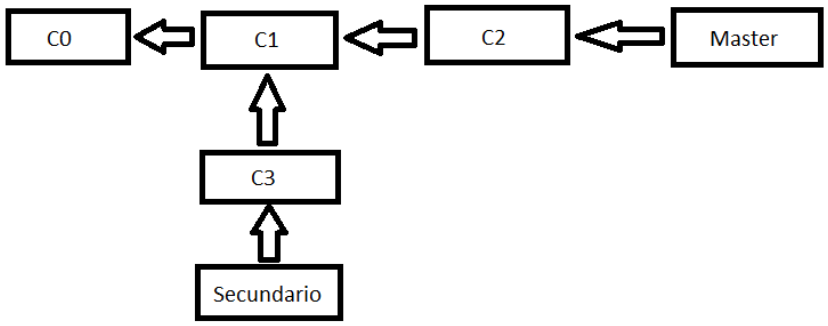
Mientras que git fetch solamente descarga el contenido desde un repositorio remoto, ya que, no integra ninguno de estos datos en los archivos de trabajo; git pull descarga el contenido y sí lo integra directamente en los archivos de trabajo actuales.

Además, con git fetch se puede estar seguro de que nunca se manipulará, destruirá o arruinará ningún archivo local; no obstante, con git pull puede producirse algo conocido como "conflicto de fusión", dado al intento de fusionar los cambios remotos con los locales.

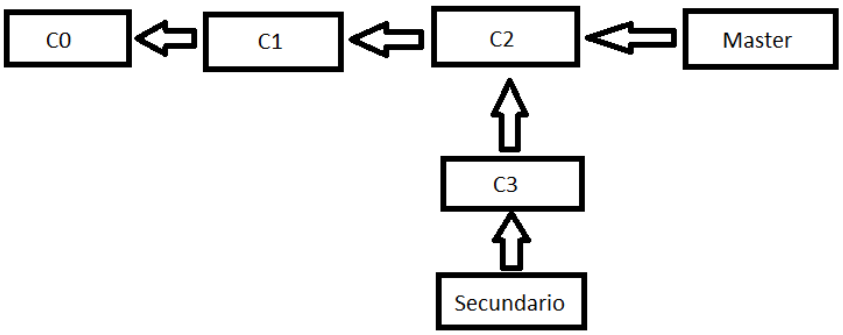
**7.** Asumiendo que usted está en un Branch llamado "secundario" y su Branch principal se llama "master" ¿Qué resultado espera de hacer git rebase master? ¿Qué resultado espera de hacer git rebase origin/master?

El comando git rebase es una herramienta de reorganización que permite capturar los cambios en un Branch determinado, y reaplicarlos encima de otro Branch. De

esta manera, si se tiene un Branch "secundario" y el Branch "master" de la siguiente manera:




Al realizar el comando git rebase master, desde el Branch "secundario", se espera que el commit C3 se reaplique ahora a C2 (sobre "master"):




El comando git rebase origin/master realiza la misma tarea, con la excepción de que se utiliza el Branch "master" del repositorio remoto que se esté utilizando.

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

Es una forma de comprobar el correcto funcionamiento de una unidad o componente individual de un software. El propósito de realizarla es validar que cada unidad del código del software funcione de la manera esperada. Para ejecutarlas se aísla una sección de código y se verifica su exactitud. En estos casos cuando se menciona unidad se puede estar haciendo referencia a una función, método, procedimiento, módulo u objeto individual.

Son de gran importancia, ya que permiten detectar errores en etapas tempranas del proceso de desarrollo de software. En caso de que estos errores se detectaran en etapas más avanzadas, la reparación sería más costosa; por lo que, pruebas unitarias bien ejecutadas se traducen en un ahorro de tiempo y dinero.

**9.** Bajo el contexto de pytest. ¿Qué es un "assert"?

Un "assert" es un comando utilizado para verificar si una cierta premisa se cumple o no. Al utilizar dicho comando dentro del código y correrlo, se realizará un test que terminará en éxito si la afirmación es correcta, o fallará en caso contrario.

Por ejemplo, al utilizar "assert test(3) == 4", se realizará una prueba en la que se verificará si la función "test" al recibir un 3, retorna un 4.

**10.** ¿Qué es Flake 8?

Para responder esta pregunta, primero es necesario aclarar que un linter es una herramienta que analiza el código fuente en busca de errores. Son capaces de detectar errores de sintaxis, problemas estructurales como el uso de variables indefinidas y violaciones a las buenas prácticas o guías de estilo de programación.

Flake 8 es una librería de Python que funciona como linter, al llamarse ejecuta las herramientas PyFlakes, pycodestyle y Ned Batchelder's script. Ayuda a revisar que el código cumpla la guía de estilo única descrita en el PEP 8 y a verificar la complejidad ciclomática de la misma.