Instituto Tecnológico de Costa Rica

Microprocesadores y microcontroladores/ MT-7003

Tarea 1

Profesor:
Rodolfo Piedra Camacho

Grupo 1

Estudiante:
José Cubero Mora / 2019061903
Moisés Salas Ramírez / 2019185151
Marco Alfaro Zamora / 2019038346

05 de agosto del 2022

Preguntas Teóricas Tarea 1

1) ¿Diferencie la herramienta Git de Github?

Git es una herramienta o sistema de control de versiones, la cual permite gestionar o administrar el historial de un código fuente. Además, Git debe ser instalado en los ordenadores para su uso, por lo que no se necesita estar conectado a internet. En cambio, Github es un repositorio de Git, el cual está guardado en la nube, por lo cual sí requiere conexión a internet para utilizarse. También, Github permite la creación de proyectos en conjunto con otras personas, el seguimiento de errores, y la carga y descarga de recursos.

2) ¿Qué es un branch?

Una rama o "branch" en Git, es una versión o una línea de desarrollo de un código original para trabajar de forma paralela al código del proyecto, es decir, los branch permiten realizar prueba a un código dudoso sin que este se fusione con la versión principal en la que se está trabajando.

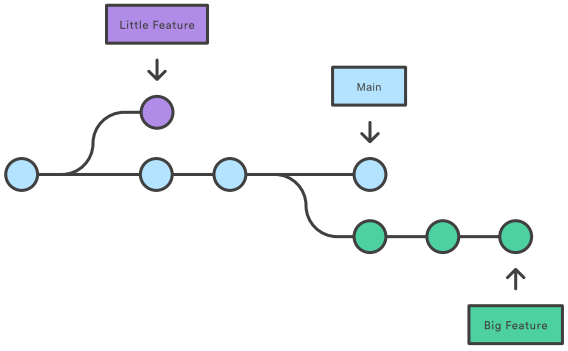

Figura X. Ejemplo de la ramificación de código en Git.

En la figura anterior se puede observar que a partir de la rama principal llamada "Main", se creó una rama de desarrollo para una función pequeña, y otra rama de desarrollo para una función grande, la creación de estas ramas permite trabajar de forma paralela en ambas ramas sin interferir una en otra, o en el propio Main.

3) ¿Como se crea un nuevo Branch?

El comando `"git branch"` permite la creación de nuevas ramas o branchs, así como enumerarlas y eliminarlas, además de poder cambiarles el nombre. Para crear una nueva

rama, se usa específicamente el comando "`git branch <branch>`, donde `<branch>` corresponde al nombre que se le dará a la rama.

4) ¿Qué es un commit?

El commit se refiere a confirmaciones que se encargan de capturar exactamente el estado del proyecto en un determinado momento. Los commits son sumamente importantes ya que sirven como herramienta para recordar cambios que se hayan realizado en los proyectos con cualquier versión y con fechas anteriores. Además, son útiles ya que permiten revertir el progreso del proyecto a estas últimas versiones. Una característica importante es que, en caso de que varios commits dentro del proyecto sean modificados, estos no se sobrescribirán entre sí. Cabe destacar que estas confirmaciones son almacenadas en el repositorio local de Git.

5) ¿Qué es la operación "git cherry-pick"?

Es un comando que permite elegir una confirmación de una determinada rama y aplicarla a otra, lo cual es útil en ciertos casos, como, por ejemplo, en el caso de que se haya aplicado una confirmación a una rama equivocada y se deba aplicar a otra; o al realizar un proyecto colaborativo con distintos sectores, los cuales trabajan en torno a un mismo código, por lo que en algunas ocasiones algún sector necesita aplicar una confirmación realizada por otro sector. Otro caso puede ser al solucionar errores, en donde alguno de los desarrolladores de un proyecto efectúe una confirmación con la solución del error, la cual puede ser aplicada directamente a la rama principal del proyecto mediante el git cherry-pick.

6) Explique que 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.

Si Git descubre que un cambio hecho en una rama entra en conflicto con un cambio hecho en otra, te pide que resuelvas un merge conflict. Un merge conflict puede ocurrir cuando las ramas que se pretenden unir editan la misma línea del código Main de forma diferente, o cuando una rama modifica un archivo y otra lo elimina.

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

Es una prueba a bajo nivel, es decir, cercana al código fuente, la cual consiste en probar individualmente las funciones y métodos de las clases o módulos empleados. Específicamente, las pruebas unitarias verifican el nombre del método, el nombre y tipo de sus parámetros, y el valor y tipo del resultado por retornar. Al ser tan específicas, tienen bajo costo y se pueden ejecutar de forma rápida mediante un servidor de integración continua. Se debe considerar que, al hacer este tipo de prueba, la función o método por probar debe aislarse hasta no poder desglosarse más, es decir, debe ser totalmente independiente de las otras funciones.

8) Bajo el contexto de pytest. ¿Qué es un "assert"?

Bajo el contexto de pytest, assert funciona para asegurarse de que el código está libre de errores. Assert permite testear el código de manera automática, en donde se llama a una función cuyo resultado ya se debe de conocer previamente, y de esta forma se comprueba que efectivamente retorna lo que se esperaba. Cuando estos tests son ejecutados en un entorno de automatización de pruebas, ese entorno captura los AssertionError que se produzcan, para así tener una especie de informe que indique cuáles tests fallaron, y así poder buscar solución.

9) ¿Qué es Flake 8?

Antes de explicar qué es Flake8 se debe de definir qué son los linters. Los linters son programas que se encargan de la detección de errores en el código. Existen dos categorías, los linters lógicos y de estilo. El primer tipo se refiere a los programas encargados de buscar errores lógicos en el código, o que podrían llegar a dar resultados no esperados. En el caso de los linters de estilo, su función es la de buscar código que no se encuentre conforme a alguna convención de código. En el caso de Flake8, este es un linter de Python, el cuál es capaz de detectar errores lógicos y de estilo. Flake8 utiliza tres paquetes, PyFlakes (encargado de revisión lógica), pycodestyle (encargado de revisión de estilo) y Ned Batchelder's McCabe (encargado de revisión de complejidad del código).

10) Explique la funcionalidad de parametrización de pytest.

El testeo en el código es una herramienta sumamente poderosa, ya que permite el desarrollo de un código de alta calidad. Sin embargo, entre mayor y más complejo es el código, el testeo se vuelve también una tarea más complicada. Es por esto por lo que la parametrización viene a simplificar este proceso. La funcionalidad de la parametrización radica en que, si se requiere un gran conjunto de valores de entrada para verificar toda la funcionalidad del código, la parametrización permite realizar esto de una forma muy rápida, en vez de tener que hacerlo de forma “manual”.