TEC
Tecnológico
de Costa Rica

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Mecatrónica

Microprocesadores y Microcontroladores

Tarea 1: Programming Tools. Parte Teórica

Alumnos:
Javier Araya Vega - 2022202152
Jose Arrieta Gutiérrez - 2019069570

Profesor:
Ing. Rodolfo Piedra Camacho

Grupo 1

26 de febrero de 2026

Resolución de Preguntas

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

   Git facilita un historial de cambios y modificaciones que sufre un proyecto a lo largo del tiempo. Por lo tanto, además de permitir guardar el progreso que se hace en este de manera segura, es posible trabajar en un código con otras personas, donde se pueden observar los cambios que se realizaron, el por qué de estos, cuándo y quién los realizó [1].

2. **¿Qué es un commit?**

   El comando git commit es, básicamente, el crear una memoria del estado actual de los archivos y del proyecto en ese instante, con tal de tener guardado todos los cambios y modificaciones que se le hicieron al momento. Es una medida de precaución que permite tener un punto de control al cual siempre volver en caso de que algo salga mal con futuras modificaciones. Generalmente, dicho comando viene acompañado de otro, llamada "git add", que prepara la escritura de los elementos [2].

3. **¿Qué es un branch?**

   Un "branch"se encarga de crear una linea de desarrollo alternativa, lo que permite a los programadores dividir el trabajo principal para desarrollar nuevas funciones, corregir errores o experimentar en un entorno distinto al del proyecto principal. Los cambios realizados en una rama no afectan a la rama principal, por lo que no hay riesgo de estropear lo que ya funcionaba a la hora de probar características nuevas [3]

4. **¿En el contexto de github, qué es un Pull Request?**

   Un "Pull Request.[es] un mecanismo de colaboración de Github, el cual, tal como su nombre lo indica, consiste en una solicitud para que otros colaboradores revisen el código modificado y los cambios planteados para el proyecto por un usuario en un branch, antes de fusionarlos con la principal. Se pueden dejar comentarios, recomendaciones y otros tipos de retroalimentación. Esto mejora y agiliza el trabajo en equipo y permite la coordinación entre los miembros del grupon de trabajo [4]

5. **¿Si un usuario quiere "Actualizar su repositorio contra el Branch master", qué debe de hacer?**

   Para actualizar una rama local de trabajo con los últimos cambios que existen en la rama principal , el usuario debe realizar un proceso de sincronización mediante comandos. Los pasos serían:

   a) Asegurarse de estar en la rama que se desea actualizar mediante git checkout Nombre-Rama.

   b) Descargar y unit los cambios del servidor con git pull origin master (o main, dependiendo de la nomenclatura del proyecto) [5].

6. **¿Bajo que condiciones una herramienta como Git necesita apoyo de un humano para saber como integrar cambios locales con cambios remotos?**

   Se necesitaría intervención humana, principalmente, en casos donde haya un conflicto a la hora de hacer un merge. Por ejemplo, se podría dar la situación en que dos desarrolladores intentan cambiar la misma línea de código y lo hacen de manera distinta. Por otro lado, podría ser el caso que alguien modifica un archivo, y otra persona lo elimina. En ambos ejemplos, git tendría dificultades a la hora de seleccionar a cuál ajuste darle prioridad, y no habría más opción que el grupo lo resuelva manualment [6]

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

   El Unit testing es un nivel de prueba que se realiza a componente individuales dentro del código. Sirve para verificar el funcionamiento correcto de ciertos elementos, tales como funciones, clases y demás, antes de unirlas con otras. Esto reduce las necesidades de hacer un debugging exhaustivo a largo plazo, y confirmar que las bases del código actúan como deben [7]

8. **Bajo el contexto de pytest. ¿Cuál es la utilidad de un "assert"?**

   El assert evalúa si una expresión lógica es verdadera (o True). En Pytest, su utilidad es verificar los resultados esperados dentro de una prueba. Si la condición evaluada resulta ser falsa (False), la prueba tira error. La ventaja de utilizar Pytest en este caso, es que realiza una reescritura de aserciones", osea, intercepta los assert de Python y genera un desglose y mensajes de error extremadamente detallados sobre por qué falló cada evaluación [8]

9. **¿Qué es Flake 8?**

   Flake 8 es una herramienta de linting para Python. Es decir, un programa que analiza el código en busca de errores o deficiencias de distintos tipos:

   a) Errores de Sintáxis o Lógica mediante **PyFlakes**.

   b) Violación de los lineamientos del PEP8 con **PyCodeStyle**.

   c) Complejidad o ramificación exagerada del código usando textbf McCabe.

   Todo esto se realiza sin necesidad de ejecutar el código y tener que esperar a que el tiempo de compilación concluya para descubrir los fallos [9].

10. **Comente sobre la utilidad de la aleatorización en pruebas de código.**

    La aleatoriedad es importante a la hora de probar el código debido a que de esta manera se pueden descubrir errores y bugs inusuales. Hacer pruebas de este tipo permite encontrar casos límites en el código y revelar otras vulnerabilidades al introducir valores válidos e inválidos, además de observar la robustez del mismo si se le somete a un nivel de estrés imprevisto. Por lo general, introducir dicha aleatorización a las pruebas, o "fuzz testing.[es] una buena práctica que se aconseja realizar para cualquier código, independientemente de qué tan simple sea [10]

Referencias

[1] GitHub Community, *DAY 1: What Is GitHub (and Why Do People Use It)?* GitHub Community Discussions, [En línea]. Disponible en: https://github.com/orgs/community/discussions/167863. [Accedido: 22-feb-2026], jul. de 2025.

[2] Atlassian, *Tutoriales de Git: git commit*, Atlassian Git Tutorial, [En línea]. Disponible en: https://www.atlassian.com/es/git/tutorials/saving-changes/git-commit. [Accedido: 22-feb-2026], 2025.

[3] Scott Chacon y Ben Straub, *Git Branching - Branches in a Nutshell*, Pro Git Book, git-scm.com, [En línea]. Disponible en: https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell. [Accedido: 22-feb-2026], 2016.

[4] GitHub, *About pull requests*, GitHub Docs, [En línea]. Disponible en: https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-pull-requests. [Accedido: 22-feb-2026], 2026.

[5] Stack Overflow, *Updating a local repository with changes from a GitHub repository*, Stack Overflow, [En línea]. Disponible en: https://stackoverflow.com/questions/1443210/updating-a-local-repository-with-changes-from-a-github-repository. [Accedido: 22-feb-2026], 2009.

[6] Scott Chacon y Ben Straub, *Git Branching - Basic Branching and Merging*, Pro Git Book, git-scm.com, [En línea]. Disponible en: https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging. [Accedido: 22-feb-2026], 2016.

[7] Microsoft, *Unit testing concepts*, Microsoft Learn, [En línea]. Disponible en: https://learn.microsoft.com/en-us/visualstudio/test/unit-test-basics. [Accedido: 22-feb-2026], nov. de 2025.

[8] Pytest Development Team, *The writing and reporting of assertions in tests*, Pytest Documentation, [En línea]. Disponible en: https://docs.pytest.org/en/latest/how-to/assert.html. [Accedido: 22-feb-2026], 2015.

[9] PyCQA, *Flake8: Your Tool For Style Guide Enforcement*, Flake8 Documentation, [En línea]. Disponible en: https://flake8.pycqa.org/en/latest/. [Accedido: 22-feb-2026], 2016.

[10] Software Engineering Stack Exchange Community, *Is the usage of random values in unit testing a good practice?* Software Engineering Stack Exchange, [En línea]. Disponible en: https://softwareengineering.stackexchange.com/questions/429601/is-the-usage-of-random-values-in-unit-testing-a-good-practice. [Accedido: 22-feb-2026], jun. de 2021.