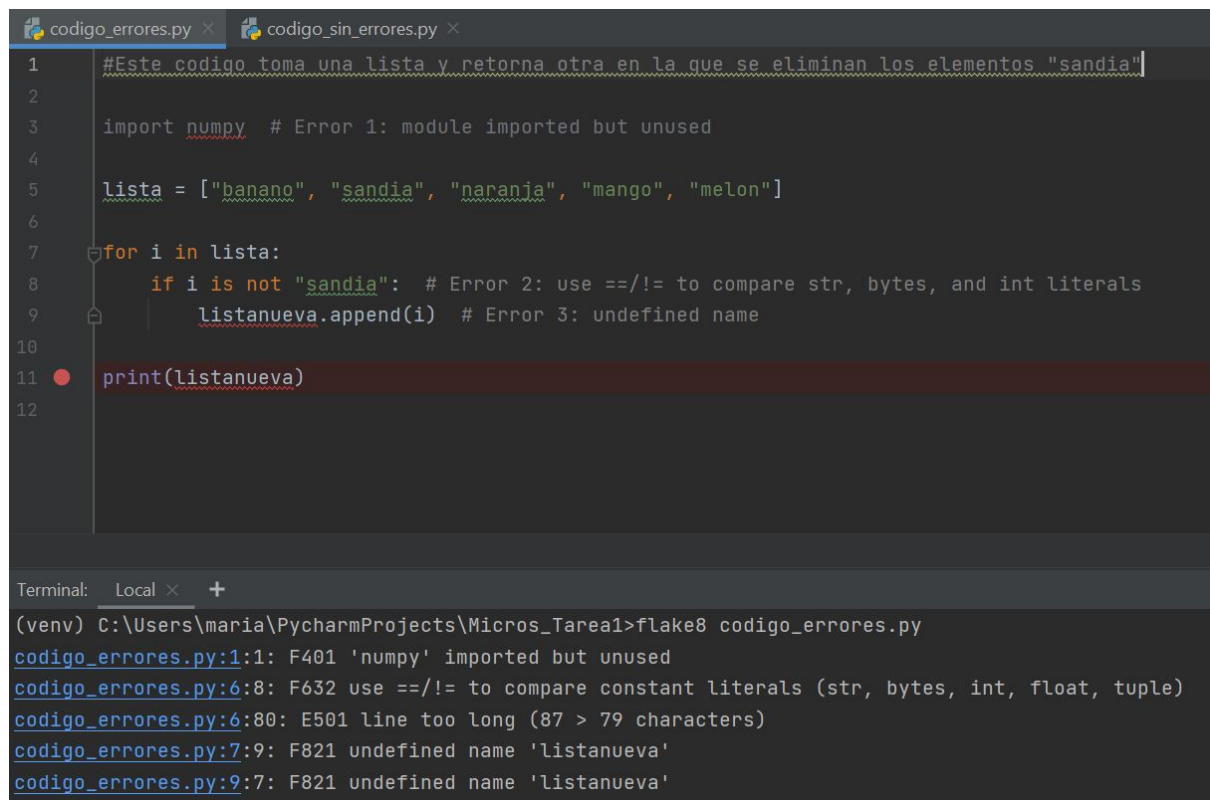


Revisión de código con errores utilizando Flake8

Esta sección corresponde a los puntos 6, 7 y 8 del enunciado de la tarea. En la Fig. 1 se muestra el código realizado con sus errores comentados y abajo se pueden observar los resultados obtenidos con Flake8. Dichos resultados fueron los esperados: undefined name, module imported but unused y use ==/!= to compare constant literals.

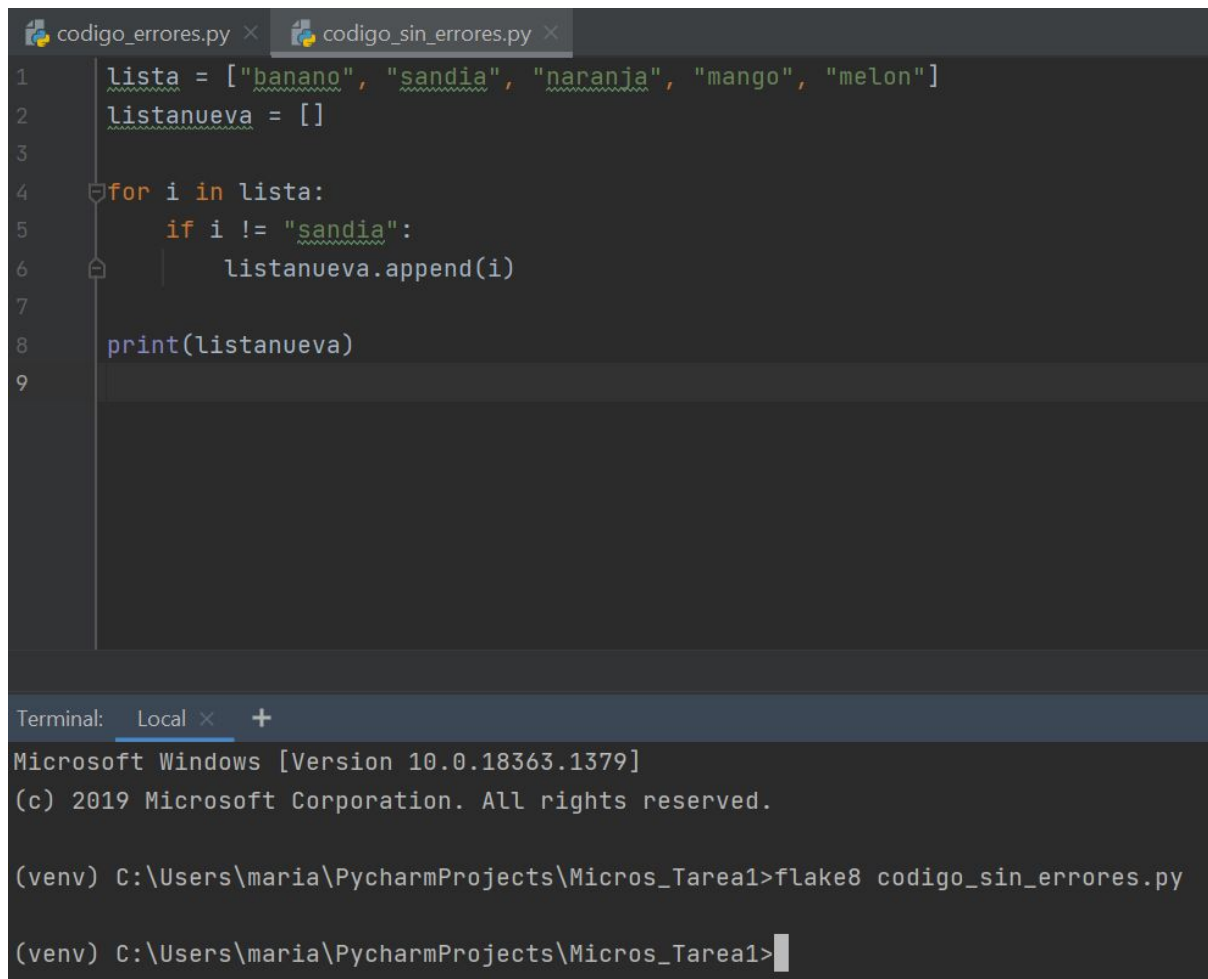


```
1 #Este código toma una lista y retorna otra en la que se eliminan los elementos "sandia"
2
3 import numpy # Error 1: module imported but unused
4
5 lista = ["banano", "sandia", "naranja", "mango", "melon"]
6
7 for i in lista:
8     if i is not "sandia": # Error 2: use ==/!= to compare str, bytes, and int literals
9         listanueva.append(i) # Error 3: undefined name
10
11 print(listanueva)
12
```

```
Terminal: Local x +
(venv) C:\Users\maria\PycharmProjects\Micros_Tarea1>flake8 codigoErrores.py
codigoErrores.py:1:1: F401 'numpy' imported but unused
codigoErrores.py:6:8: F632 use ==/!= to compare constant literals (str, bytes, int, float, tuple)
codigoErrores.py:6:80: E501 line too long (87 > 79 characters)
codigoErrores.py:7:9: F821 undefined name 'listanueva'
codigoErrores.py:9:7: F821 undefined name 'listanueva'
```

Figura 1. Código con errores detectados por flake8 y errores obtenidos.

Luego se realizaron correcciones al código y se volvió a revisar con flake8. El resultado fue el esperado ya que no mostró errores y se puede observar en la Fig. 2.



The image shows a code editor with two tabs: 'codigo_errores.py' and 'codigo_sin_errores.py'. The 'codigo_sin_errores.py' tab is active, displaying the following Python code:

```
1 lista = ["banano", "sandia", "naranja", "mango", "melon"]
2 listanueva = []
3
4 for i in lista:
5     if i != "sandia":
6         listanueva.append(i)
7
8 print(listanueva)
9
```

Below the code editor is a terminal window with the following output:

```
Terminal: Local x +
Microsoft Windows [Version 10.0.18363.1379]
(c) 2019 Microsoft Corporation. All rights reserved.

(venv) C:\Users\maria\PycharmProjects\Micros_Tarea1>flake8 codigo_sin_errores.py

(venv) C:\Users\maria\PycharmProjects\Micros_Tarea1>
```

Figura 2. Código con las correcciones correspondientes y resultados de flake8.

Preguntas teóricas

1. Diferencie la herramienta Git de Github

Git es un sistema de control de versiones que permite manejar y organizar diferentes versiones de proyectos que tienen muchos archivos de código fuente. Github, en cambio, provee servicios de alojamiento y control de repositorios de Git. Ambos son gratis, open source y facilitan la colaboración en proyectos de software.

2. ¿Qué es un branch?

En un sistema de control de versiones, un branch es una copia de una línea de código. Las ramas permiten a equipos de desarrolladores de software trabajar en paralelo, ya que separan el código en desarrollo del código probado y estable. Esto es muy útil para experimentar sin afectar el código principal del proyecto, al cual se le refiere comúnmente como trunk, baseline, master o mainline [1].

3. ¿Qué es un commit?

El comando commit realiza una captura de un repositorio en un momento específico y se utiliza para guardar cambios localmente debido a que durante el desarrollo de un proyecto, utilizando estos sistemas, los cambios se guardan primero localmente y luego en el repositorio. Los commits se realizan en dos fases, primero se debe utilizar el comando add para añadir los archivos con los cambios que se desea guardar y luego el comando commit, si no se añaden los archivos previamente, el comando commit no funciona. Además de su contenido, los commits contienen más información como autor y tiempo, entre otros [2][3].

4. ¿Qué es la operación cherry-pick?

La operación cherry-pick permite elegir un commit individual de una rama y aplicarla a otra. Esto es muy útil pero se debe evitar porque crea commits duplicados, por lo que se debería realizar un merge o rebase si es posible. Un merge o rebase se diferencia del cherry-pick porque con el merge todos los commits de una rama son integrados a otra [4][5].

5. ¿Qué hace el comando git stash?

Se utiliza este comando para guardar temporalmente código, en el cual se han realizados cambios pero no están terminados y, por lo tanto, aún no se desea hacer un commit. Para realizar esto, el comando toma los cambios sin confirmar, los guarda aparte para poder acceder a ellos más tarde y, luego, los deshace en el código en el que se está trabajando [6].

6. Compare las operaciones git fetch y git pull

El comando git fetch descarga commits, archivos y referencias de un repositorio remoto al repositorio local. Es más común utilizarla cuando se quiere observar los cambios realizados por otros. El comando git pull es, en realidad, una combinación de dos comandos, git fetch seguido de git merge. En la primera etapa de la operación git pull ejecutará un git fetch en la rama local a la que apunta HEAD. Una vez descargado el contenido, git pull entrará en un flujo de trabajo de fusión [7].

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?

Rebase en su forma más simple, es un comando que “transferirá” otra rama (Master) a la rama donde está trabajando actualmente (Secundario), aplicando todas sus confirmaciones únicas de la Rama “Secundario” en la parte superior de la Rama “Master” y reemplazando la Rama “Secundario” con esta versión revisada. De esa manera, también puede “ponerse al día” con otras ramas reescribiendo el historial de git en su git local. Por ejemplo, puede Rebase encima de Master para poner su rama al día con el estado actual del mundo en su rama de características [8].

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

Estos test permiten examinar el correcto funcionamiento de cada uno de los elementos antes de que ocupen su lugar en el concepto general de un programa. Además, ayudan a comprobar de forma relativamente rápida y fácil si el componente funciona según lo previsto por el desarrollador. Los test unitarios son una de las formas más eficaces para descubrir la mayor cantidad posible de errores del código en las fases tempranas de desarrollo del software [9].

9. Bajo el contexto de pytest. ¿Qué es un “assert”?

Las declaraciones de assert son el núcleo de lo que es un test. Para las pruebas unitarias, cada función o método de prueba debe tener una y solo una declaración de assert.

La idea general es configurar los datos de entrada requeridos, llamar a una función o método desde el código fuente y luego confirmar que el resultado se ajusta a las expectativas. El assert en pytest, comparado a otros, expande la salida para dar más contexto [10].

10. ¿Qué es Flake 8?

Flake8 es parte de herramientas que realizan análisis estático del código fuente para comprobar si hay discrepancias semánticas. En sí es la envoltura que verifica pep8, pyflakes y circular complexity. Además tiene una baja tasa de falsos positivos [11].

Referencias

- [1] B. Schiestl. (2020, February 28). Branching definition [Online]. Available: <https://www.perforce.com/blog/vcs/branching-definition-what-branch>
- [2] Tower. (2021). Git Commands [Online]. Available: <https://www.git-tower.com/learn/git/commands/git-commit>
- [3] GitHub, Inc. (2021). Git Guides: Git Commit [Online]. Available: <https://github.com/git-guides/git-commit>
- [4] Tower. (2021). The git cherry-pick command: what it is and how to use it [Online]. Available: <https://www.git-tower.com/learn/git/faq/cherry-pick>
- [5] Atlassian. (2021). Git Cherry Pick [Online]. Available: <https://www.atlassian.com/es/git/tutorials/cherry-pick>
- [6] Atlassian. (2021). Git stash [Online]. Available: <https://www.atlassian.com/es/git/tutorials/saving-changes/git-stash>
- [7] Atlassian. (2021). Git pull [Online]. Available: <https://www.atlassian.com/es/git/tutorials/syncing/git-pull>
- [8] N. LeClaire. (2014, September 14). Git rebase [Online]. Available: <https://nathanleclaire.com/blog/2014/09/14/dont-be-scared-of-git-rebase>
- [9] Ionos. (2019, March 14). Unit testing: la prueba de calidad para software [Online]. Available: <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-papel-del-unit-test-en-el-desarrollo-de-software/>
- [10] Pytest. (2015). Assert statements [Online]. Available: https://docs.pytest.org/en/reorganize-docs/new-docs/user/assert_statements.html
- [11] Dev Null. (2017, January 30). "What is Flake8 and why we should use it?" [Online]. Available: <https://medium.com/python-pandemonium/what-is-flake8-and-why-we-should-use-it-b89bd78073f2>