Resolucion ejercicio tipo examen PPS

Pasos previos

1. Crear el entorno

```
python3 -m venv nombreEntorno
```

2. Activarlo

```
source nombreEntorno/bin/activate
```

3. Instalar dependencias

```
pip install pytest coverage pylint sphinx
```

4. Guardar dependencias

```
pip freeze > requirements.txt
```

Ejercicio 1 - Tests unitarios y cobertura

1. Crear carpeta

```
mkdir ejercicio_01
```

2. Generamos archivo de pruebas unitarias

```
touch ejercicio_01/test_procesa.py
```

3. Escribimos lo siguiente:

```
import pytest
 from procesa import procesa, cuentaVocales, limpia

 def test_procesa_cuenta():
 assert procesa("cuenta", "hola mundo") == 4
```

```
 assert procesa("cuenta", "PYTHON") == 1
 assert procesa("cuenta", "") == None

 def test_procesa_limpia():
     assert procesa("limpia", "Hola, mundo!") == "Holamundo"
     assert procesa("limpia", "123 !@# abc") == "123abc"
     assert procesa("limpia", "") == None

 def test_procesa_operacion_no_permitida():
     assert procesa("invalida", "texto") == "Operación no permitida"

 def test_cuentaVocales():
     assert cuentaVocales("AEIOUaeiou") == 10
     assert cuentaVocales("xyz") == 0
     assert cuentaVocales("") == None

 def test_limpia():
     assert limpia("Texto con espacios, y signos!") ==
"Textoconespaciosysignos"
     assert limpia("123 456") == "123456"
     assert limpia("") == None
```

4. Ejecutar tests

```
coverage run -m pytest ejercicio_01/test_procesa.py
coverage html
```

Ejercicio 2: Pruebas de integración y cobertura

1. Creamos carpeta

```
mkdir ejercicio_02
```

2. Generamos archivo de tests

```
touch ejercicio_02/test_app01.py
```

Con el siguiente contenido

```
import subprocess

def test_app_cuenta():
    result = subprocess.run(["python3", "app.py", "cuenta", "hola
mundo"], capture_output=True, text=True)
    assert "> 4" in result.stdout
```

```
def test_app_limpia():
    result = subprocess.run(["python3", "app.py", "limpia", "hola,
mundo!"], capture_output=True, text=True)
    assert "> holamundo" in result.stdout

def test_app_operacion_invalida():
    result = subprocess.run(["python3", "app.py", "desconocida",
"texto"], capture_output=True, text=True)
    assert "> Operación no permitida" in result.stdout
```

3. Ejecutamos las pruebas

```
coverage run -m pytest ejercicio_02/test_app01.py
coverage html
```

Ejercicio 3: Linter y documentación

1. Creamos carpeta

```
mkdir ejercicio_03
```

2. Aplicamos un linter (Pylint) para verificar la calidad del código:

```
pylint procesa.py app.py
```

3. Corrige los errores o advertencias que indique el linter.

4. Documenta el código añadiendo docstrings en cada función de procesa.py y app.py:

```
def procesa(operacion: str, cadena: str) -> str:
"""
Procesa una cadena según la operación indicada.

Args:
    operacion (str): Tipo de operación ('cuenta' o 'limpia').
    cadena (str): La cadena a procesar.

Returns:
    str: Resultado de la operación o mensaje de error.
"""
```

5. Genera la documentación HTML usando Sphinx:

- Inicializamos:

```
sphinx-quickstart
```

- Edita index.rst para incluir procesa.py y app.py.
- Generamos la documentación

```
make html
```