1 DE NOVIEMBRE DE 2025

PROGRAMACIÓN

ACTIVIDAD 4

RESUMEN

Complejidad y Mantenibilidad: LOC, Halstead, ciclomática, índice de mantenibilidad.
Refactorizar código

GERARDO CALABRESE

UNIVERSIDAD DE SAN MARTÍN

INGENIERÍA EN SISTEMAS ESPACIALES

INDICE

1. CONTEXTO Y OBJETIVO

Esta práctica tiene por finalidad que utilices la librería radon de Python y analices el código del archivo **"Simulación de CMGs.py"**.

El **archivo "Simulación de CMGs.py"** desarrolla una simulación del control de actitud de un microsatélite utilizando giróscopos de control de momento (CMGs). El modelo emplea ecuaciones de Euler para los giróscopios, analizando variables de momento angular, angulo de gimbals y los torques de entrada y salida.

El **análisis** se basa en utilizar las cuatro métricas vistas en clase: LOC, Halstead, ciclomática e indice de mantenibilidad (MI); debiendose explicar que significa cada una de las salídas que radon da para cada una.

Además, debe identificar las dos "malas prácticas" que convenie atacar para mejorar el script.

2. SUPUESTOS Y DATOS

Supuestos:

- No modificar la lógica física del simulador durante el primer análisis.
- Las métricas son entregadas mediante la implementación de la librería radon.
- Versión de Python ≥ 3.10
- Versión radon 6.0.1
- Métrica CC por función.
- Métrica MI por archivo.

Datos:

- Código base "Simulación de CMGs.py"

3. TAREA

3.1. ANÁLISIS DE EJECUCIÓN

Dentro del script, hay dos malas prácticas generales. Identifique cuales son y detalle como las corregiría.

3.2. ANÁLISIS DE MANTENIBILIDAD

Realice el análisis de mantenibilidad y complejidad mediante las métricas LOC, Halstead, ciclomática e MI.

3.3. REFACTOR

Una vez realizado el punto 3.2, cambiar parte de la estructura interna del código sin cambiar el comportamiento observable. En otras palabras, ordene el script actual, dando mejor legibilidad, modularidad, nombres, separación de responsabilidades.

Algunos “No” que debe cumplir:

- No debe reescribir completamente el código.
- No debe agregar características.
- No debe optimizar con cambios de resultado (ejemplo, de simple a doble posición).
- No debe cambiar la interfaz pública.

Guarde las modificaciones en un archivo “Simulación de CMGs_v2.py”

Hay tres tareas de refractor que deberá modificar, dos de ellas están relacionadas con el apartado 3.1

Al finalizar la tarea de refactorizar, deberá ejecutar un nuevo análisis para medir mantenibilidad y complejidad.

4. ENTREGABLES

Los entregables de esta actividad son:

1. Archivo “Simulación de CMGs_v2.py”.
2. Documento donde detalle el análisis realizado sobre el archivo original y el obtenido luego refactorizar el código.