Documentación PIA E3
Equipo 5

En esta fase de nuestro proyecto PIA tuvimos que crear un nuevo script (llamado Script 2 Semana 3.py) el cual analizará los datos recabados de los archivos de texto del script principal. Para esto tuvimos que modificar el código original ya que teníamos poca información en los archivos txt.

Modificaciones script original:

```
while True:
        opcion= int(input(f"\nDesea guardar la lista de IDs en un archivo de texto? [1-Si/2-No]: "))
        if opcion==1:
            with open (f"Asteroides_IDs.txt","w") as file:
                for i in range(inicial, final + 1):
                    asteroide = asteroides_totales[i]
                    if (inicial == 0):
                        file.write(f"\n{i+1}. ID: {asteroide['id']} | Nombre: {asteroide['name']}")
                    else:
                        file.write(f"\n{i}. ID: {asteroide['id']} | Nombre: {asteroide['name']}")
            print("Listo!")
            break
        elif opcion==2:
            print("Regresando al menú principal…")
            break
        else:
            print("Opcion invalida. ¡Pruebe de nuevo!")
```

El cambio principal fueron las líneas de código en las cuales permite que el usuario también guardará un archivo de texto con las listas con las IDs seleccionas. Todos los archivos utilizados en el script principal nos servirán para la representación visual y análisis del nuevo script.

Explicación segundo script:

```
import re
from statistics import mean

# Lectura y validación de IDs y nombres
def leer_ids_nombres(ruta_ids):
    asteroides = []
    patron = re.compile(r'ID:\s*(\d+)\s*\|\s*Nombre:\s*(.+)')

    with open(ruta_ids, 'r') as archivo:
        for linea in archivo:
            linea = linea.strip()
            if not linea:
                continue  # Saltar líneas vacías
            match = patron.search(linea)
            if match:
                id_ = match.group(1)
                nombre = match.group(2)
                asteroides.append({
                    "ID": id_,
                    "Nombre": nombre
                })
            else:
                print(f"[Advertencia] Línea no válida: {linea}")

    return asteroides
```

Primero se importan las librerías de re (expresiones regulares) y la de statistics que nos permitirá encontrar el promedio del diámetro ya que la API nos ofrece su mínimo y máximo estimado.
Las expresiones regulares nos permiten validar que la ID ingresada sea válida.

En resumen, el script recibe como input la ID del NEO que queremos visualizar luego el programa busca en nuestros archivos los archivos con la información recolectada (en el script principal) y finalmente los muestra y proporciona opciones para hacer cálculos.

Nota: La función eval() no fue vista en clase pero al investigar sobre opciones de convertir strings a diccionarios se consideró implementar. La fuente consultada para su ejecución: https://www.pythonparatodo.com/?p=129

Operaciones:

Las operaciones pueden calcularse al momento de que el usuario decida seleccionar una de las opciones del menú secundario.

1. Calcular promedio de diámetros estimados: como la API ofrece múltiples medidas de diámetro, es posible acceder al diccionario compuesto y calcular su promedio con la librería de statistics.

2. Aproximar magnitud H (se relaciona con el brillo) la luminosidad del sol: se utilizan los datos del archivo de texto.

Fuentes utilizadas para la documentación del módulo de statistics: https://www.geeksforgeeks.org/python-statistics-mean-function/