 

**PIA - Entregable 2**

**Universidad Autónoma de Nuevo León**

**Facultad:** Facultad de Ciencias Físico Matemáticas

**Materia:** Programación básica

**Nombre del maestro:** Perla Marlene Viera González

**Grupo:** 073

**Fecha de entrega:** 6 de mayo de 2025

| **Matricula** | **Nombre** |
| --- | --- |
| **2136071** | **Santiago Resendiz Badillo** |
| **1909518** | **Victor Adrian Rodriguez Ortiz** |

**Documentación**

1. **Métodos de extracción de datos y herramientas empleadas.**

En nuestro proyecto trabajamos con Python como lenguaje principal y usamos el módulo requests para conectarnos a la API. Hicimos pruebas básicas con fechas únicas e intervalos de fechas variables para asegurarnos de que la información que necesitábamos se almacenara de forma correcta y de forma cronológica, una vez que logramos esto comenzamos a realizar las 52 consultas necesarias para tener almacenada la información correspondiente al año 2024. Esto nos ayudó a obtener y almacenar la información de los asteroides en el archivo .txt.

1. **Técnicas de limpieza aplicadas.**

Los datos recibidos desde la API venían en formato JSON, pero incluían mucha información irrelevante. Para simplificar, filtramos únicamente lo necesario: nombre del asteroide, fecha de aproximación, tamaño, velocidad, distancia y si era potencialmente peligroso. Esto nos permitió mantener los datos limpios, ordenados y fáciles de leer en el archivo .txt.

1. **Validaciones implementadas**

En esta etapa del proyecto trabajamos con expresiones regulares para validar la entrada del día, mes y año. Antes usábamos try y except, pero los reemplazamos por patrones definidos importando así, el módulo re. Estos patrones permitían solo valores válidos, como del 01 al 31 para días, del 01 al 12 para meses, y de 2001 a 2025 para años. También ajustamos los valores para que tuvieran siempre dos cifras, agregando un “0” al inicio cuando era necesario.

También nos aseguramos de que la información recabada sea de tipo string o flotante, dependiendo de la naturaleza del concepto

1. **Diseño lógico y estructura de los datos**

Los datos fueron organizados en diccionarios individuales, cada uno representando un asteroide con su información clave. Estos diccionarios se almacenaron en una lista general. Esta estructura nos facilitó recorrerlos y guardarlos adecuadamente en el archivo .txt.

1. **Transformaciones y archivo .txt**

Para guardar los datos, primero comprobamos que la lista de asteroides no estuviera vacía. Luego, abrimos el archivo en modo “append” ("a") para que cada nueva consulta se añadiera sin borrar la información anterior. Usamos encoding="utf-8" para evitar errores con caracteres como acentos o la letra ñ. Dentro del archivo, escribimos cada dato clave-valor en líneas separadas. Después de cada asteroide, agregamos una línea en blanco para separar visualmente cada bloque de información.

1. **Retos enfrentados.**

Uno de los retos fue cómo integrar expresiones regulares dentro del código, investigamos cómo funcionaban y las aplicamos en la validación de fechas.

Otro reto fue construir los patrones correctos para día, mes y año. Lo resolvimos investigando en internet, implementando elementos como \b, 0?, [1-9] y [12].

También nos enfrentamos a problemas con caracteres especiales al guardar el archivo .txt, pero lo solucionamos usando utf-8 como codificación.

Por último, nos enfrentamos al problema de que al realizar las consultas no mostraban los asteroides en forma cronológica e investigando en internet llegamos con la función sorted(), nos ayudó bastante porque era compatible para organizar cronológicamente el formato de nuestras fechas, al tener nuestras fechas organizadas nos permitía recorrer los asteroides de cada día y de forma cronológica.