s

Universidad Nacional Autónoma de México

Facultad de Ciencias

Modelado y programación

Góngora Ramírez Dania Paula

Manuel Jesus Casillas Olivier

Carballido Camacateco Ana Lilia

README.

1. Ejecución

Para ejecutar el proyecto se necesitan las bibliotecas csv, json y requests de python, en la versión 3.8.10, por lo que para usar esto es necesario instalar pip3. Además, para la compilación, es necesario pararse en la carpeta programa y se debe escribir:

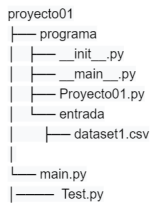
python3 Proyecto01.py

Así como para los test, pararse en la carpeta pruebas y esribir:

python3 Test.py

Por último es necesario instalar el paquete DOTENV para poder ingresar la api key necesaria para acceder a OpenWeather (la indicación de en qué línea de código se debe ingresar la api key viene en el cuerpo del código fuente.

2. Estructura



3. Definición del problema

El aeropuerto de la Ciudad de México nos contrata para entregar el informe del clima, tanto de la ciudad de origen como la ciudad de destino, esto para 3 mil tickets que salen el mismo día que se corre el algoritmo de forma que no sea interactivo.

4. Analisis del problema

Para este problema contamos con lo siguiente:
Datos de entrada:
El archivo dataset1.csv, el cual contienen los datos de las ciudades de origen y destino en clave IATA, junto con los datos de longitud y latitud de cada ciudad.

Datos de salida:

Queremos obtener los datos del clima de cada ciudad; temperatura, humedad, presión y una descripción breve.

Se necesita acceder a los datos del clima de la forma más precisa y directa posible de cada ciudad.

Para consultar estos datos se utilizó la API OpenWeatherMap la cual regresa estos datos en formato XML,HTML o JSON, por lo que es necesario saber manejar este formato para acceder a la información requerida.

La información de entrada que nos proporciona el dataset son los datos suficientes para resolver el problema, ya que nos proporciona las coordenadas de cada ciudad, con estas coordenadas podemos hacer las peticiones a la API.

Como el programa no es interactivo, no se contemplan más entradas, pero es importante mostrar la información de manera clara. En este caso, el aeropuerto de la ciudad de México nos contrató, por lo tanto se mostrará la información en español.

5. Selección de la mejor alternativa

La mejor alternativa sería crear una función que defina un caché de datos con las peticiones por medio de la API a OpenWeather. De esta manera, el conjunto de datos podrá estar guardado temporalmente en la memoria para su posterior uso. Por último, bastaría con la definición de un método que defina a cada elemento del clima (ya sea temperatura, humedad, etc) una variable que tome los datos recogidos por el caché para ser imprimidos al usuario. Nosostros consideramos que es incoveniente y poco eficiente mostrar cada uno de los 3000 boletos, si se necesita el clima basta con mostrar los datos de cada ciudad, la persona que necesite la información le bastara con buscar su ciudad entre los datos, por lo que la salida de nuestro programa muestra los datos de cada ciudad.

6. Pensamiento a futuro

El mantenimiento de dicho programa podría realizarse actualizando, si no a tiempo real, puede que cada hora los datos del clima por cada boleto al ser éstos sumamente variables respecto al tiempo. De igual manera, se podría actualizar la información de cada vuelo por día o por semana, borrando el caché anterior y llamando a un nuevo método que periódicamente esté solicitando mediante la clave IATA las API's correspondientes. Por último, el proyecto en sí puede actualizarse a una versión mejorada (2.0, por ejemplo) que incluya una interfaz gráfica agradable al usuario y mejoras en tiempos de carga, para ello se necesitará realizar muchisimas más pruebas y métodos que simplifiquen los bloques de código haciendo uso de nuevas bibliotecas o paqueterías.

Podría considerarse que dicho proyecto fuera de libre acceso (gratuito para cualquier usuario interesado) una vez terminada la licencia de uso con el aereopuerto.

7. Descripción del proyecto

Para el proyecto importamos varias bibliotecas de python; assert para elaborar las pruebas, importamos csv para leer el archivo dataset1.csv, requests para hacer peticiones a OpenWeatherMap. Todo sobre las bibliotecas utlizadas puede consultarse con mayor detalle en: `https://docs.python.org/es/3.8/`

Utilizamos dictreader ya que esto facilito la lectura del csv, para hacer las peticiones utilizamos los datos de latitud y longitud del csv, ya que la API indicaba que de esta forma la consulta sería más precisa, la API del web service indicaba que podiamos recibir la información requeritan en XML, HTML o JSON, por lo que decidimos importar json para manejar los datos del clima proporcionados por OpenWeatherMap

8. Recursos

Utilizamos lo siguiente: El repositorio del usuario Armando122

En el archivo proyecto01.py nos basamos en lo siguiente:

El método peticiones() se baso en peticiones(), el método lecturaCache() se copio del método lectura(), el método salidaClima() se baso en el método impresion(), todo esto del archivo proyecto01.py

Las pruebas: pruebaPeticiones() y pruebaLecturaCache() se basaron en los métodos testcache() y testpeticiones() respectivamente, todo esto del archivo tests.py

`https://github.com/Armando122/Proyectos-Modelado-y-Programacion/tree/master/Tarea01`

Ya que era nuestra primera vez utilizando python y más recursos, consultamos:

Uso del assert(). (s. f.). El Libro De Python. Recuperado 8 de marzo de 2022, de

`https://ellibrodepython.com/assert-python`

Grupman, C. (2021, 12 diciembre). Python API Tutorial: Getting Started with APIs. Dataquest. Recuperado 8 de marzo de 2022, de

`https://www.dataquest.io/blog/python-api-tutorial/`

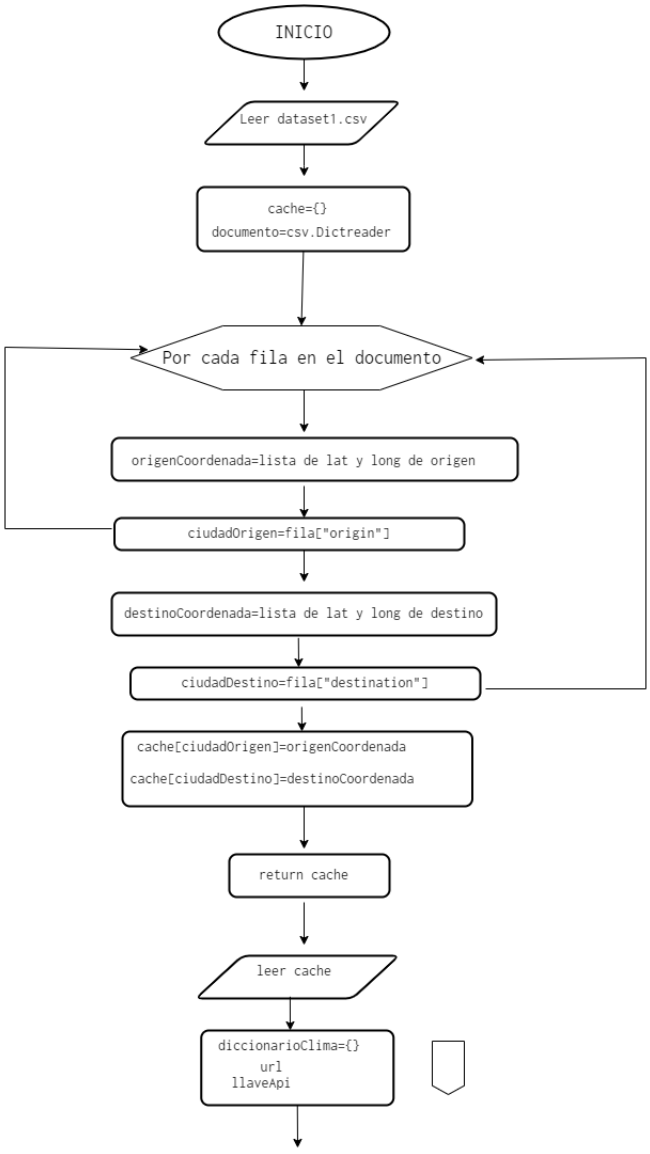
Assertions in Python. (s. f.). Tutorials Point. Recuperado 9 de marzo de 2022, de

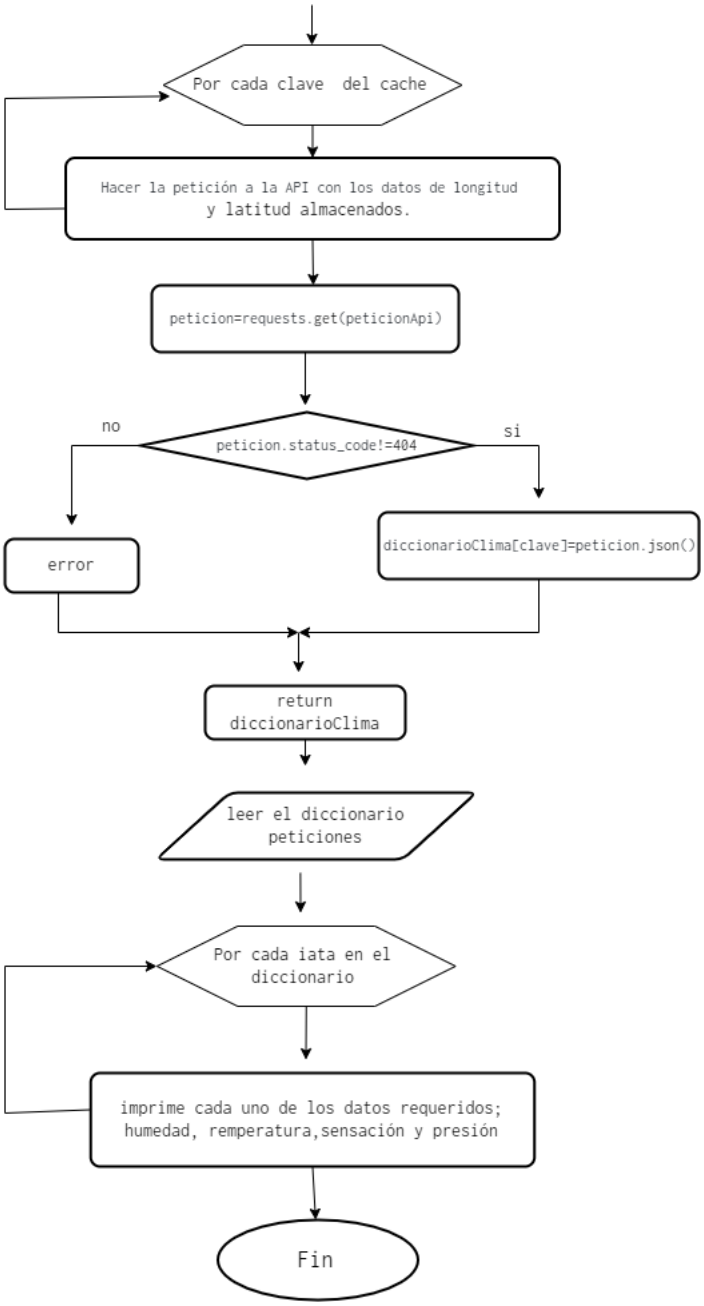
`https://www.tutorialspoint.com/python/assertions_in_python.htm`

`https://courses.cs.washington.edu/courses/cse140/13wi/csv-parsing.html`

Matthes, E. (2016). Curso Intensivo de Python (2da Edición, Vol. 1) [Libro electrónico]. ANAYA.

9. Diagrama de flujo




Por cada clave del cache
Hacer la petición a la API con los datos de longitud y latitud almacenados.
peticion=requests.get(peticionApi)
peticion.status_code!=404
no
si
error
diccionarioClima[clave]=peticion.json()
return diccionarioClima
leer el diccionario peticiones
Por cada iata en el diccionario
imprime cada uno de los datos requeridos; humedad, remperatura,sensación y presión
Fin