Módulo 3: Trabajando con datos en archivos de texto

Procesamiento de archivos HTML

Conociendo el módulo BeautifulSoup

Existen varias forma de procesar archivos HTML, lo primero es saber que un archivo HTML es un archivo de texto en formato XML, se puede decirt que HTML es un tipo especial de XML o por ejemplo un factura CFDi es otro tipo de XML.

Para procesar archivos HTML podríamos usar sólo la instruccion `open()` y realizar alguna operación en esa línea, luego la siguiente línea y así hasta terminar, el incoveniente es que la información en un archivo HTML está organizada por etiquetas del tipo `html, head, body, h1` que marcan el inicio y final de un bloque pudiendo terminar en una o varias líneas.

Así que obtener información de un archivo HTML o XLM se realiza por medio de módulos o librerías especiales para ello como `lxml`, `xml` o `html5lib`, pero en particular para HTML hay un módulo que simplifica aún más el trabajo y está creado especial mente para procesar información en formato HTML llamado `BeutifulSoup`

- Sitio oficial: https://www.crummy.com/software/BeautifulSoup

Y para instalarlo se puede realizar con el siguiente comando:

```
pip install beautifulsoup4
```

In [ ]:
```
...
```

Conociendo el lenguaje de marcado HTML

Para conitnuar primer veamos un ejemplo de archivo HTML básico:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Mi página web</title>
</head>
<body>
  <h1>Este es un encabezado</h1>
  <p>Este es un párrafo</p>
</body>
</html>
```

Hoy en día se usa HTML5 y una pequeña lista de las etiquetas usadas es la siguiente:

| Etiqueta | Descripción | |---|---| | <html> | Define el inicio del documento HTML. | | <head> | Contiene información sobre el documento HTML, como el título, los enlaces a los estilos CSS y otros metadatos. | | <title> | Define el título del documento HTML. | | <link> | Enlaza un recurso externo al documento HTML, como un archivo CSS o un archivo JavaScript. | | <body> | Contiene el contenido del documento HTML, como texto, imágenes, formularios y otros elementos. | | <p> | Define un párrafo de texto. | | <h1> | <h2> | <h3> | <h4> | <h5> | <h6> | Definen encabezados de diferentes tamaños. | | | Crea una división en el documento HTML. | | <a> | Crea un enlace a otro documento o a otra parte del mismo documento. | | <img> | Inserta una imagen en el documento HTML. | | <table> | Crea una tabla en el documento HTML. | | <tr> | Define una fila en una tabla. | | <td> | Define una celda en una tabla. | | <form> | Crea un formulario en el documento HTML. | | <input> | Crea un campo de entrada en un formulario. | | <select> | Crea una lista desplegable en un formulario. | | <button> | Crea un botón en un formulario. |

Esto es importante saberlo, porque en base a éstas etiquetas haremos la búsqueda de información.

Ejemplo: Dado dado el siguiente código HTML obtén todos los links y el texto asociado a cada uno e imprime una tabla.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Mis sitios preferidos</title>
</head>
<body>
  <h1>Mis sitios preferidos</h1>
  <p>La siguiente es una lista de sitio que suelo consultar
regularmente</p>
  <ul>
    <li><a href="https://www.google.com">Google</a></li>
    <li><a href="https://www.wikipedia.org">Wikipedia</a></li>
    <li><a href="https://www.youtube.com">YouTube</a></li>
    <li><a href="https://www.facebook.com">Facebook</a></li>
    <li><a href="https://www.twitter.com">Twitter</a></li>
  </ul>
</body>
</html>
```

In [ ]:
```
from bs4 import BeautifulSoup

texto_html = """<!DOCTYPE html>
<html lang="es">
```

```
<head>
  <title>Mis sitios preferidos</title>
</head>
<body>
  <h1>Mis sitios preferidos</h1>
  <p>La siguiente es una lista de sitio que suelo consultar regularmente</p>
  <ul>
    <li><a href="https://www.google.com">Google</a></li>
    <li><a href="https://www.wikipedia.org">Wikipedia</a></li>
    <li><a href="https://www.youtube.com">YouTube</a></li>
    <li><a href="https://www.facebook.com">Facebook</a></li>
    <li><a href="https://www.twitter.com">Twitter</a></li>
  </ul>
</body>
</html>"""

# Resuleve usando un ciclo for
...
```

In [ ]:
```
# Resuleve usando listas de compresión
etiquetas_a = ...
lineas_a = ...
print( "\n".join(lineas_a) )
```

Ejemplo: Un poco más interesante, usa el archivo descargado de la Wikipedia en el módulo anterior, copialo a ésta carpeta (lo simple es bello) y obten una lista que incluya el nombre del estado y su código ISO a 3 letras (5 si se incluye el MX), impre una tabla como resultado.

In [ ]:
```
from bs4 import BeautifulSoup

# Leer archivo
# Procesar archivo usando BeautifulSoup
# Encuentra la tabla ojetivo
# Encuentra las filas objetivo
# Crea una lista de los estados usando los índices de las columnas adecuadas
# Valida que los datos de ambas columnas estén limpias
# Imprime una tabla
...
```

Manejo de Archivos CSV

Conociendo el formato de archivos CSV

A continuación se muestra un ejemplo del contenido de un archivo CSV:

```
csv
nombre,apellido,edad,sexo
Juan,Perez,21,M
Maria,Lopez,23,F
```

```
Pedro,Garcia,25,M
Ana,Martinez,27,F
```

Así que las características del formato de archivos de texto CSV se puede resumir en la siguiente tabla:

Característica	Descripción
Formato	Comma Separated Values (CSV)
Separador	Coma (,)
Filas	Cada fila representa una línea de datos
Columnas	Cada columna representa un campo de datos
Datos	Los datos pueden ser de cualquier tipo, como números, texto, fechas, etc.
Formato	Los datos pueden estar formateados de diversas maneras, como con comillas, espacios, etc.
Usos	Los archivos CSV se pueden utilizar para almacenar una gran variedad de datos, como listas de clientes, productos, ventas, etc.
Software	Los archivos CSV pueden ser abiertos por una gran variedad de software, como Microsoft Excel, OpenOffice Calc, etc.

Conociendo el módulo `csv`

La librería estándar de Python cuenta con el módulo `csv` que permite leer y escribir listas de datos en formato CSV, como es parte de la librería estándar no es necesario instalarlo, sólo hay que importarlo.

Ejemplo: Lee el contenido del archivo `fulanitos.csv` e imprime una tabla con el resultado.

In [ ]:
```python
import csv

def print_table(datos):
    """ Imprime en forma de tabla la lista de listas contenida en datos """
    ancho_max = [0] * len(datos[0])
    for fila in datos:
        for i, col in enumerate(fila):
            if len(col) > ancho_max[i]:
                ancho_max[i] = len(col)
    for fila in datos:
        cols = [f"{val:{ancho_max[i]}}" for i, val in enumerate(fila)]
        linea = " | ".join(cols)
        print(linea)

...
```

Ejemplo: Escribe en un archivo llamado `mexico-estados-iso31662mx.csv` con la lista obtenida de los estados con su código ISO al hacer web scrapping a la página web de la Wikipedia, agrega una fila adicional con el nombre de las columnas.

In [ ]:
```
import csv

print_table(estados[:5])

...
```

Manejo de Archivos JSON

Conociendo el formato de archivos CSV

A continuación se muestra un ejemplo del contenido de un archivo CSV:

```
{
  "nombre": "Juan Pérez",
  "apellido": "López",
  "edad": 21,
  "sexo": "M",
  "fecha_nacimiento": "1999-01-01",
  "direccion": {
    "calle": "Calle 1",
    "numero": 123,
    "colonia": "Colonia Centro",
    "ciudad": "Ciudad de México",
    "estado": "CDMX",
    "pais": "México"
  },
  "telefonos": [
    "+52 55 5555 5555",
    "+52 55 5555 5556"
  ],
  "correos_electronicos": [
    "juan.perez@example.com",
    "juan.perez2@example.com"
  ]
}
```

Así que las características del formato de archivos de texto JSON se puede resumir en la siguiente tabla:

Característica	Descripción
Formato	JavaScript Object Notation (JSON)
Sintaxis	Basada en texto

Característica	Descripción
Tipos de datos	Cadena, número, booleano, objeto, array
Estructura	Objetos anidados
Portabilidad	Compatible con muchos lenguajes de programación
Usos	Intercambio de datos, serialización de objetos, representación de datos
Software	Muchos lenguajes de programación tienen soporte para JSON

Conociendo el módulo `json`

La librería estándar de Python cuenta con el módulo `json` que permite leer y escribir datos en formato JOSN, como es parte de la librería estándar no es necesario instalarlo, sólo hay que importarlo.

Ejemplo: Lee el contenido del archivo `persona.json` e imprime el resultado.

In [ ]:
```
import json

...
```

Ejemplo: El sitio The Movie DB mantiene una base de datos de películas de todo el mundos con métricas interesante, una de ellas es la lista del top de películas en base a las más votadas de todos los tiempos y puede ser consultada en la página Pelĺiculas más votadas.

Adicionalmente The Mivide DB cuenta con un acceso vía API que permite tener acceso a la base de datos en tiempo real y pueder ser consultada por todo tipo de público y para ello cuentan con una página bien documentada donde se indica el procedimiento para obtener una llave (**key**) y como hacer uso de la API, así como de la información disponible en el sitio TMDB API.

Obtén la lista de la películas más votadas usando la API consultando la documentación en el siguiente link y almacena la lista de las películas más votadas en el archivo `top-rated-movies.json`

In [ ]:
```
import json
from pprint import pprint
import requests

...
```

En el caso de que se necesite usar un proxy para el acceso a la API con el módulo requests se puede hacer lo siguiente:

```
proxies = {
    "http": "http://127.0.0.1:8080",
    "https": "https://127.0.0.1:8080"
}

response = requests.get(url, proxies=proxies)
```

Procesamiento de Textos en Lenguajes Naturales

Cuando se trabaja con datos que incluyen exto en lenguaje natural es importante hacer algunas consideraciones.

- Usar el mismo conjunto de caracteres (latin-1, iso8859-1, utf-8, ascii(
- Para los sistemas mayúsculas y minúsculas son diferentes, para un lenguaje natural el significado no cambia, así que antes de realizar algún análisis transforar todo a minúsculas o mayúsculas
- Los símbolos especiales de cada lenguaje generalmente deben ser eliminados dejando sólo las palabras y los blancos.
- En el lenguaje en Español en particulas además también hay que eliminar acentos, ya que *México* es distinto a *Mexico* y puede generar resultados incovenientes.

Ejemplo: Dados las siguientes listas de nombres, utiliza todos las herramientas vistas hasta ahorita para intentar comtejar las columnas de nombres de ambas listas. Comenta tus resultados.

Los archivos a cotejar son:

- `Lista-de-participante_Ciencia-de-Datos.csv`
- `participants_91461136067_20230808.csv`

In [ ]:
```
...
```