Gestión de la Información en la Web
Curso 2024-25
Práctica 3 – Formato XML

Fecha de entrega: domingo 20 de octubre de 2024

Entrega de la práctica
La entrega de la práctica se realizará a través del Campus Virtual de la asignatura mediante un fichero `pr3.py`. El esqueleto de este fichero se puede descargar del Campus Virtual.

Lenguaje de programación
Python 3.11 o superior.

Calificación
Se medirá la corrección mediante tests de unidad. Además de la corrección, se valorará la **calidad, concisión y claridad del código**, la incorporación de **comentarios** explicativos, su **eficiencia** tanto en tiempo como en memoria y la puntuación obtenida en Pylint.

Declaración de autoría e integridad
Todos los ficheros entregados contendrán una cabecera en la que se indique la asignatura, la práctica, el grupo y los autores. Esta cabecera también contendrá la siguiente declaración de integridad:

> *Declaramos que esta solución es fruto exclusivamente de nuestro trabajo personal. No hemos sido ayudados por ninguna otra persona o sistema automático ni hemos obtenido la solución de fuentes externas, y tampoco hemos compartido nuestra solución con otras personas de manera directa o indirecta. Declaramos además que no hemos realizado de manera deshonesta ninguna otra actividad que pueda mejorar nuestros resultados ni perjudicar los resultados de los demás.*

No se corregirá ningún fichero que no venga acompañado de dicha cabecera.

El objetivo de esta práctica es crear una serie de funciones para extraer información a partir de un fichero XML con información sobre los restaurantes de la ciudad de Madrid. Para ello se va a usar el catálogo «Restaurantes con perfil turístico de la ciudad de Madrid» del portal de datos abiertos del Ayuntamiento de Madrid. El conjunto de datos incluye información sobre restaurantes de la ciudad de Madrid divididos por tipos de cocina, especialidades, servicios, etc. Así mismo, se incluyen los datos básicos de cada restaurante, una descripción, su geoposición y dirección postal, así como un conjunto de fotografías relativas a cada punto. En los casos en los que aplica, se ofrecen horarios de apertura y costes de acceso si los tuviera. La información se encuentra en: `https://datos.madrid.es/sites/v/index.jsp?vgnextoid=ce33a73970504510VgnVCM2000001f4a900aRCRD&vgnextchannel=374512b9ace9f310VgnVCM1000001` y el conjunto de datos XML que hay que procesar está situado en: `https://datos.madrid.es/egob/catalogo/300033-10037079-turismo-restauracion.xml`. Junto al enunciado de la práctica se proporciona el fichero XML a tratar y una versión más fácil de leer *por humanos* `restaurantes_v1_es_pretty.xml`.

Antes de empezar a escribir las funciones, el primer paso será revisar el fichero XML y entender la estructura de sus entradas.

Funciones para extraer información del fichero XML

1) Nombres de restaurantes [2pt]

Escribe una función `nombres_restaurantes(filename)` **usando SAX** que recibe la ruta del fichero XML y devuelve una **lista ordenada** alfabéticamente con los nombres de todos los restaurantes que aparecen. Ten en cuenta que el fichero XML contiene texto HTML escapado (por ejemplo tildes y otros símbolos) que deberás «desescapar». Para ello utiliza la función `html.unescape` en la biblioteca predeterminada de Python `html`. También debes eliminar los posibles espacios al principio y final que pueda haber en los nombres de restaurantes.

Un ejemplo de la salida de esta función sería:

```
['"Tres Peces" El Ventorrillo Murciano',
 '100 Montaditos',
 '11 Nudos Terraza Nordés',
 '19 Sushi Bar',
 '80º',
 '99 Sushi Bar (Hermosilla)',
 ...
]
```

2) Subcategorías de cocina [3pt]

Escribe una función `subcategorias(filename)` **usando SAX** que recibe la ruta del fichero XML y devuelve un **conjunto** de todas las subcategorías que existen en el fichero de restaurantes. Como las subcategorías pertenecen a una categoría, queremos generar cadenas del estilo `'CATEGORIA > SUBCATEGORIA'`.

Ten en cuenta que un único restaurante puede tener varias subcategorías, por ejemplo:

```
<categorias>
       <categoria>
         <item name="idCategoria">7102</item>
         <item name="Categoria">Internacional</item>
         <subcategorias>
           <subcategoria>
             <item name="idSubCategoria">7462</item>
             <item name="SubCategoria">Fusión</item>
           </subcategoria>
           <subcategoria>
```

```
            <item name="idSubCategoria">7146</item>
            <item name="SubCategoria">Internacional</item>
          </subcategoria>
          <subcategoria>
            <item name="idSubCategoria">7116</item>
            <item name="SubCategoria">Japonesa</item>
          </subcategoria>
          <subcategoria>
            <item name="idSubCategoria">7119</item>
            <item name="SubCategoria">Peruana</item>
          </subcategoria>
        </subcategorias>
      </categoria>
    </categorias>
```

Este restaurante debería generar las siguientes 4 subcategorías en el conjunto:

- `'Internacional > Fusión'`
- `'Internacional > Internacional'`
- `'Internacional > Japonesa'`
- `'Internacional > Peruana'`

Un ejemplo de la salida de esta función sería:

```
{'Española > Andaluza',
 'Española > Arrocería',
 'Española > Asador / Parrilla',
 'Española > Asturiana',
 'Española > Balear',
 'Española > Canaria',
 ...
}
```

3) Información de un restaurante [2.5pt]

Escribe una función `info_restaurante(filename, name)` **utilizando DOM o ElementTree** que recibe la ruta del fichero XML y un nombre de restaurante y devuelve un diccionario Python con la información básica del restaurante:

- Nombre del restuarante
- Descripción (campo `body`)
- Correo electrónico
- Web
- Número de teléfono
- Horario

Si alguno de estos campos no existe, la función no debe fallar pero el diccionario contendrá `None` en esa entrada. De la misma manera, si no hay ningún restaurante con ese nombre se devolverá `None` en lugar de un diccionario. Al igual que en apartados anteriores, debéis «desescapar» las cadenas que tengan texto HTML escapado.

Un ejemplo de invocación a la función sería el siguiente diccionario:

```
{'nombre': 'Hasaku Nikkei',
 'descripcion': '<p><strong>La apuesta culinaria por la cocina nikkei del chef ... </p>',
 'email': None,
 'web': 'https://www.esmadrid.com/restaurantes/hasaku-nikkei',
 'phone': '(+34) 91 210 54 43',
 'horario': '<p>Mar - Sáb: 13:00 - 16:00h/20:00 - 01:00 h</p><p>Domingo: 13:00 - 16:00 h</p>'}
```

4) Restaurantes por cercanía [2.5pt]

Escribe una función `busqueda_cercania(filename, lugar, n)`, **utilizando DOM o ElementTree**, que recibe la ruta del fichero XML, un lugar expresado como una cadena de texto soportada por GeoPY y un número n de kilómetros, y devuelve una lista de parejas `(distancia, nombre_restaurante)` con aquellos restaurantes que están como mucho a n kilómetros de distancia del lugar indicado. **La lista generada debe estar ordenada de lugar más cercano a más lejano**, y debes «desescapar» el texto HTML escapado en los nombres de los restaurantes. Para el manejo de lugares y distancias debes utilizar la biblioteca GeoPy, tal y como vimos en la práctica 2.

Un ejemplo del resultado invocación

```
busqueda_cercania('restaurantes_v1_es.xml',
                  'Profesor José García Santesmases 9, Madrid, España', 3)
```

sería la siguiente lista:

```
[(1.5539707436564782, 'Restaurante Café de Oriente Museo del Traje'),
 (1.6565954679430905, 'Sal Gorda'),
 (2.011533065363515, 'Marmalé'),
 (2.0150300085925226, 'Gobolem (San Francisco)'),
 (2.0548691024420425, 'José Luis (San Francisco)'),
 (2.078395829609302, 'Desde 1911'),
 ...
]
```