Teoría de Lenguajes

Curso 2022

Laboratorio 1 – Expresiones Regulares

El propósito de este laboratorio es trabajar con expresiones regulares de una manera práctica, para lo cual se propone escribir programas en el lenguaje Python.

El trabajo está enfocado en resolver los problemas haciendo un fuerte uso de expresiones regulares, evitando sustituirlas por sentencias clásicas de programación estructurada. En general los programas se pueden resolver en menos de 15 líneas.

Modo de trabajo

- Se indicarán 5 programas a realizar.
- Se entregarán, para cada programa, archivos de entrada y archivos de salida que contienen la salida que se debería obtener para su respectiva entrada.
- El estudiante, en base a lo especificado por la letra, la entrada, y su correspondiente salida de referencia, deberá implementar su programa.
- La salida del programa del estudiante deberá ser idéntica a la salida de referencia.
- Se deberá respetar los nombres de los archivos de entrada, de salida, y de los programas Python.
- Se proporcionará un archivo comprimido que contiene lo siguiente:
    - Los archivos de entrada.
    - Los archivos con las salidas de referencia.
    - Los archivos Python en los cuales deben implementar su solución, dentro del subdirectorio “programas”.
    - Un script para facilitar las tareas de ejecutar y comparar las salidas, test.py.
    - El programa diff.exe para Windows que, tomando como entrada dos archivos, detecta si hay diferencias.

Archivos sobre los que trabajaremos

XML, o *eXtensible Markup Language*, es un lenguaje de marcado utilizado para almacenar datos de forma ordenada y estructurada. En esta tarea se trabajará utilizando como entrada archivos XML que contienen metadatos de imágenes, con el objetivo de utilizar expresiones regulares para extraer información contenida en ellos.

Ejemplo de archivo XML con metadatos de una imagen

```
<?xml version="1.0" encoding="UTF-8"?>
<Image>
      <FileSize> 85 kB </FileSize>
      <FileModifyDate> 2022-01-17T15:25:15.000+00:00 </FileModifyDate>
      <FileAccessDate> 2022-03-12T11:33:05.000+00:00 </FileAccessDate>
      <FileInodeChangeDate> 2022-05-27T17:45:12.000+00:00
</FileInodeChangeDate>
      <FileType> JPEG </FileType>
      <FileTypeExtension> jpg </FileTypeExtension>
      <MIMEType> image/jpeg </MIMEType>
      <JFIFVersion> 1.01 </JFIFVersion>
      <ResolutionUnit> inches </ResolutionUnit>
      <Resolution>
            <X> 299 </X>
            <Y> 499 </Y>
      </Resolution>
      <Comment>
           File source: http://commons.wikimedia.org/wiki/File:Yellow_Happy.jpg
      </Comment>
      <Description>
           http://commons.wikimedia.org/wiki/File:Yellow_Happy.jpg
      </Description>
      <Size>
            <X> 1200 </X>
            <Y> 1200 </Y>
      </Size>
      <EncodingProcess> Baseline DCT, Huffman coding </EncodingProcess>
      <BitsPerSample> 8 </BitsPerSample>
      <ColorComponents> 3 </ColorComponents>
      <YCbCrSubSampling> YCbCr4:4:4 (1 1) </YCbCrSubSampling>
      <ImageSize> 1200x1200 </ImageSize>
      <Megapixels> 1.4 </Megapixels>
</Image>
```

Programas a implementar

Se deben implementar los cinco programas descritos a continuación. En el entorno de trabajo que deben descargar del EVA, pueden encontrar ejemplos para cada uno de ellos.

programa1.py

Retornar la fecha de la última modificación de la imagen, con el siguiente formato:

```
15:25 del 2022-03-17
```

programa2.py

Retornar la resolución vertical y horizontal de la imagen, con el siguiente formato:

```
Resolución X: 299
Resolución Y: 499
```

programa3.py

Retornar la cantidad de propiedades que indican fechas, es decir, aquellas cuyas etiquetas contienen la palabra “date”. Se debe contemplar que pueden existir etiquetas que no aparecen en los ejemplos vistos.

programa4.py

Para cada propiedad cuyo valor contenga una url que comienza con *http* o *https* y no es de un dominio uruguayo (no termina en *.uy*), retornar el nombre de la propiedad. Se puede asumir que antes y después del link siempre habrá un espacio en blanco, y que no habrán dos links en un mismo ítem. La salida debe desplegarse como se muestra a continuación:

```
<Comment> -- http://commons.wikimedia.org/wiki/File:Yellow_Happy.jpg
<Description> -- http://commons.wikimedia.org/wiki/File:Yellow_Happy.jpg
```

programa5.py

Retornar el mismo archivo de entrada, pero removiendo las propiedades cuyo valor es únicamente un número, ya sea un número entero (como en el caso de *Resolution->X*) o un real (como en el caso de la propiedad *Megapixels* o *JFIFVersion*). Las etiquetas que queden sin ningún contenido y las líneas que solo contengan espacios también deberán ser eliminadas (aunque estuvieran así en el archivo original).

Herramientas a utilizar

Recomendamos utilizar la versión de Python 3.10 [1]. Por más información consultar su documentación [2].
Utilizaremos además el módulo de expresiones regulares de Python, llamado *re*. Sugerimos consultar los sitios [3] y [4] para aprender sobre el mismo.

Desarrollo del trabajo

Los trabajos se deben realizar en grupos de 2, 3 o 4 estudiantes. No pueden realizarse de forma individual. Si se detectan trabajos copiados, o si se publican soluciones tanto parciales como completas en los foros de consulta, se sancionará con la pérdida del laboratorio.

Entrega

El límite para la entrega es el **22 de abril a las 23:59**. Próximo a esta fecha se habilitará un formulario en EVA para realizar la entrega.

No necesitan inscribir el grupo previo a la entrega.

Se debe entregar:

- Los archivos programa{1,2,3,4,5}.py con los programas implementados
- Un archivo integrantes.txt con las cédulas, sin puntos ni dígito de verificación, y nombres, sin tildes, de los integrantes del grupo (uno por línea).

Formato del archivo *integrantes.txt*:

```
1234567,Perez,Santiago
4567123,Martinez,Veronica
3444555,Garcia Rodriguez,Juan Jose
```

Nota: observar que no hay espacios antes ni después de las comas que separan la cédula de los nombres, y los nombres de los apellidos.

Corrección

La corrección se realizará en el **sistema operativo Linux**. Si bien en la mayoría de los casos el funcionamiento de los programas es igual en Linux y Windows, recomendamos ejecutar los casos de prueba en Linux antes de entregar.

Utilizaremos los casos de prueba que les entregamos para la corrección (además de algunos extra), por lo tanto si la ejecución cancela, o hay diferencia en los archivos de salida con la salida oficial, la solución no se considerará correcta. Recordamos que la solución debe ser realizada mediante el uso de expresiones regulares, y no sentencias de programación.

Es parte de las normas usar los nombres y formatos de los programas y el archivo *integrantes.txt*, y se penalizarán las entregas que no cumplan esto.

Referencias

[1] https://www.python.org/downloads/release/python-3103/
[2] https://docs.python.org/3/
[3] https://docs.python.org/3/library/re.html
[4] https://www.w3schools.com/python/python_regex.asp