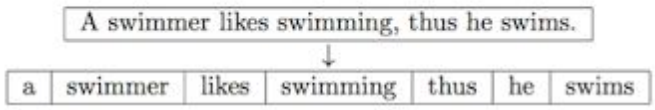
Instituto Tecnológico de Costa Rica
IC-1801 Taller de Programación
Semestre I, 2016

II Tarea Programada

Tokenización

Es el proceso de extraer de un flujo de texto las palabras, frases, símbolos u otros elementos significativos, cada uno de estos elementos es llamado token. La tokenización es útil en análisis lingüístico (donde funciona como una forma de segmentación del texto) y en ciencias de la computación, donde forma parte del análisis léxico del texto.



Archivos

Los archivos son conjuntos de datos residentes en almacenamiento secundario, como discos, que mantienen la información aun cuando se apague el computador. Los datos almacenados en archivos se conocen como datos persistentes.

Python ve cada archivo como un flujo secuencial de caracteres, donde una marca de EOF (*End of File*) determina el fin del archivo.

Las posibles operaciones con archivos son: apertura del archivo, lectura, escritura y cerrado del archivo. Para mayor detalle referirse al capítulo 10 del libro *Introducción a la Programación en Python* del Profesor Jaime Solano.

Expresiones regulares

Es una secuencia de caracteres que forma un patrón de búsqueda, principalmente utilizada para la búsqueda de patrones de cadenas de caracteres u operaciones de sustituciones.

En el área de la programación las expresiones regulares son un método por medio del cual se pueden realizar búsquedas dentro de cadenas de caracteres. Sin importar la amplitud de la búsqueda requerida de un patrón definido de caracteres, las expresiones regulares proporcionan una solución práctica al problema. Adicionalmente, un uso derivado de la búsqueda de patrones es la validación de un formato específico en una cadena de caracteres dada, como por ejemplo fechas o identificadores.

Por lo tanto, en esta tarea, el uso de expresiones regulares es recomendado para determinar cuando una palabra es un verbo en presente mediante sus clasificaciones: infinitivo (terminado en ar, er, ir), participio (terminado en ando, iendo) y gerundio (terminado ado, ido, to, so, cho.)

Para mayor informacion con respecto a la sintáxis y uso de expresiones regulares en Python 3.5.1, puede consultar el siguiente vinculo: https://docs.python.org/3/library/re.html

Ejemplo de expresión regular para validar una fecha en formato “dd/mm/aaaa”:

```
# Importar la librerIa para el manejo de expresiones regulares
import re

def esFechaValida(fecha):
    expresionRegularFecha = "[0-9]{2}\/[0-9]{1}[0-2]{1}\/[0-9]{4}"
    if re.match(expresionRegularFecha, fecha):
        return True
    else:
        return False
```

Referencias adicionales sobre expresiones regulares

https://platzi.com/blog/expresiones-regulares-python/

http://python-para-impacientes.blogspot.com/2014/02/expresiones-regulares.html

Interfaz gráfica (tkinter)

Las aplicaciones para los usuarios finales, son más atractivas e intuitivas si se cuenta con una interfaz gráfica(GUI), es por ello que los lenguajes de programación proveen herramientas para agilizar el proceso.

Python en nuestro caso, cuenta con el módulo Tkinter, de tal manera que nos dota de un conjunto de librerías para el desarrollo de Interfaces de usuario, por ejemplo: ventanas, botones, etiquetas y cajas de texto, entre otros.

Algoritmos de ordenamiento alfabético

En computación y matemáticas un **algoritmo de ordenamiento** es un algoritmo que pone elementos de una lista o un vector en una secuencia dada por una relación de orden, es decir, el resultado de salida ha de ser un reordenamiento de la entrada que satisfaga la relación de orden dada. Las relaciones de orden más usadas son el orden numérico y el orden lexicográfico.

La relación de orden lexicográfico es conocida principalmente por su aplicación en el ordenamiento de cadenas de caracteres, por ejemplo en diccionarios o en la guía telefónica.

En esta tarea debe implementar un algoritmo de ordenamiento basado en una relación de orden **lexicográfico** a fin de ordenar las listas alfabéticamente..

Ejemplo de ordenamiento alfabético:

Para una lista dada: a = ["Luis", "Anabel", "Juan", "Ana"]

El resultado del ordenamiento debe ser:

['Ana', 'Anabel', 'Juan', 'Luis']

HTML 5

Antecedentes

Los orígenes de la Web

Internet no solo ha marcado uno de los más importantes avances tecnológicos del siglo XX, sino que también ha acompañado un cambio cultural de trascendencia que, en pleno siglo XXI, se mantiene en constante evolución. Pero toda historia tiene un comienzo, e Internet también lo tuvo, mucho antes de ser un fenómeno masivo.

La historia cuenta que el antecesor de Internet fue el proyecto conocido como ARPANET, una red descentralizada que algunos organismos estadounidenses utilizaron a partir de la década del sesenta. Sin embargo, el gran cambio se produciría entre fines de los ochenta y principios de los noventa, con la llegada de lo que se conoce como World Wide Web, es decir WWW, el sistema que se encarga de permitir la distribución de información mediante hipertexto.

De la mano de este cambio, comienza a popularizarse Internet en la población. Los usuarios ahora podían acceder a contenidos de la gran red, tan solo con disponer de una conexión mediante un módem y un navegador con la capacidad de interpretar contenidos de hipertexto. Esta etapa de Internet, que comprende aproximadamente desde principios de los noventa hasta el año 2003, es considerada como Web 1.0.

El concepto de este primer paradigma de la Web responde a la idea de una web “estática” o de una “sola vía”, donde el usuario es solo un “espectador” que recibe o lee contenidos, publicados por el Webmaster o dueño del sitio. Este paradigma se modificaría de manera sustancial con la llegada de la denominada Web 2.0.

Web 2.0

Los cambios en la Web no solo responden a temas tecnológicos, sino que estos van de la mano con la evolución de los hábitos de los usuarios, las tendencias en los modos de navegación, las necesidades del mercado y hasta con aspectos culturales que también influyen en este conjunto.

La Web 2.0 representa principalmente un cambio cultural en Internet. Los usuarios, cansados de un rol pasivo, comienzan a buscar alternativas de participación. Nace una web social, donde los blogs, las redes sociales y las aplicaciones online son las estrellas.Esto ocurre a partir del año 2004.

Web 3.0

El concepto de Web 3.0 es, quizás, más complejo de definir y discutido que el caso de sus predecesores: la Web 1.0 y 2.0. Existen diversas características que la definen, entre las cuales podemos mencionar: semántica, geolocalización, Web 3D, accesibilidad desde diversos dispositivos y también inteligencia artificial.

La Web semántica, como muchas veces se define a la Web 3.0, se refiere al uso de etiquetas o bien de metadatos para otorgar un significado semántico a los elementos de la Web. Esto posibilita cierta automatización y la posibilidad de utilizar, con un mayor nivel de eficiencia, los agentes inteligentes que pueden realizar detección de contenidos.

Las características de geolocalización, muy empleadas en los equipos móviles, también han llegado a nuestro escritorio. Aunque aún pueden no ser tan precisas, las técnicas cada vez son más depuradas, y las mejoras en este campo no detienen su avance. Poder identificar a una persona, un dispositivo o cualquier elemento de manera geoespacial abre todo un mundo de posibilidades en el campo de la informática y, en especial, para todo lo referente a Realidad Aumentada.

La posibilidad de acceder desde distintos dispositivos es una realidad para una gran cantidad de usuarios y un desafío muy importante para diseñadores y desarrolladores web. Los usuarios ya no están limitados a utilizar Internet desde una computadora de escritorio, ni siquiera dependen de una laptop. Teléfonos móviles, tablets, lectores de libros electrónicos y consolas de videojuegos son solo algunas de las posibilidades que se presentan para que el usuario pueda acceder a Internet en cualquier momento y desde cualquier lugar.

W3C

El World Wide Web Consortium (W3C) es el ente o consorcio, de alcance internacional, que se encarga de crear las reglas que se utilizan como recomendaciones fundamentales para la estandarización de los principales lenguajes y tecnologías utilizados en Internet, como el caso de HTML, CSS, XML, DOM y SVG

Lenguajes de etiquetas

Los lenguajes de etiquetas, también conocidos como lenguajes de marcado o de marcas, son los que nos permiten estructurar un documento mediante el uso de etiquetas. Un ejemplo muy popular de un lenguaje de etiquetas es HTML. Algunos otros son: XML, SGML, entre otros.

HTML

HTML (HyperText Markup Language o lenguaje de marcado de hipertexto) es el lenguaje de etiquetas que funciona como una de las piedras angulares de la World Wide Web. Aunque la evolución de Internet nos ha traído muchos avances en lo que se refiere a tecnología (Web 2.0 y Web 3.0, mediantes), el lenguaje de etiquetas que se popularizó en la década del noventa sigue siendo fundamental para el desarrollo web, ya que es el que comprenden e interpretan los navegadores.

HTML5

HTML5 plantea una evolución necesaria para HTML, que luego de más de una década en la versión 4.01 necesitaba, de manera imperiosa, una renovación para estar al día con las necesidades del desarrollo web actual.

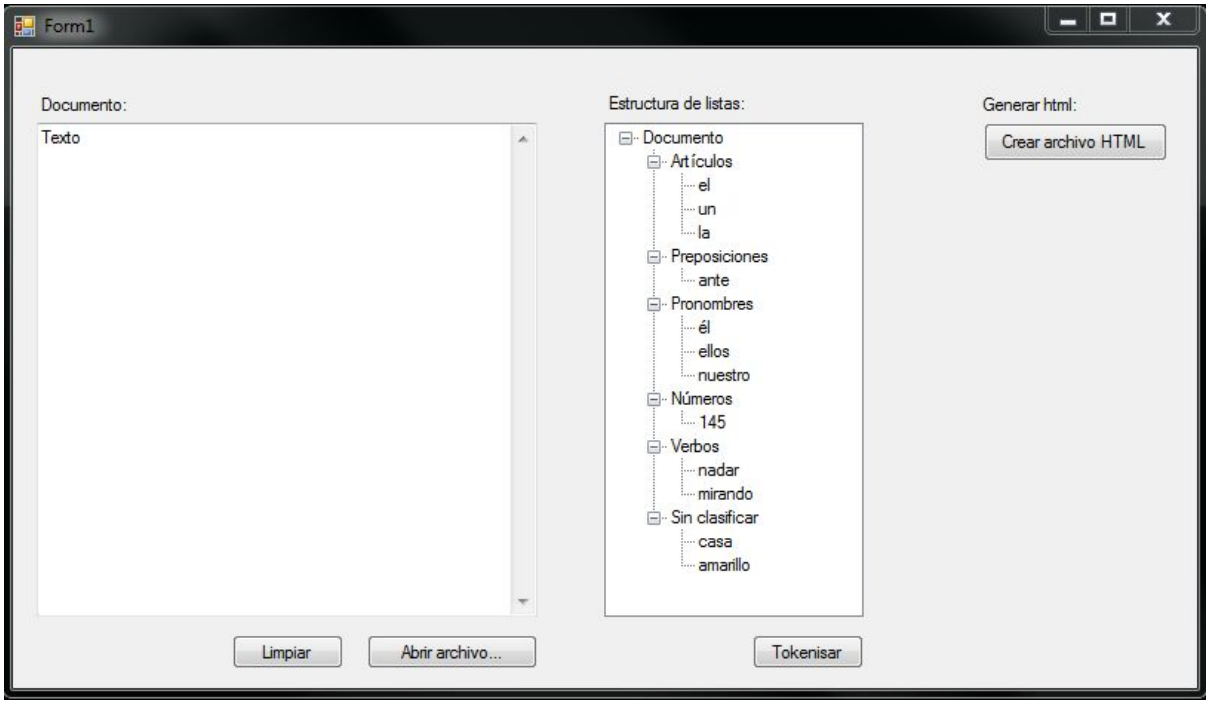
En HTML5, se destacan sus características semánticas, las posibilidades multimedia que incorpora, las nuevas funciones para formulario y las características que se definen para poder integrarse con tecnologías que permitirán abrir una nueva etapa en Internet, en lo que se refiere a la arquitectura de las aplicaciones. Por estos motivos, HTML5 es considerado como uno de los motores más importantes de la Web 3.0.

Ejemplo de estructura básica de un documento en formato HTML5

```
<!DOCTYPE html>

<html lang="es">

<head>
<title>Titulo de la web</title>
<meta charset="utf-8" />
<link rel="stylesheet" href="estilos.css" />
<link rel="shortcut icon" href="/favicon.ico" />
<link rel="alternate" title="Pozolería RSS" type="applicat
</head>

<body>
    <header>
       <h1>Mi sitio web</h1>
       <p>Mi sitio web creado en html5</p>
    </header>
    <section>
       <article>
           <h2>Titilo de contenido<h2>
           <p> Contenido (ademas de imagenes, citas, videc
       </article>
    </section>
    <aside>
       <h3>Titulo de contenido</h3>
           <p>contenido</p>
    </aside>
    <footer>
        Creado por mi el 2011
    </footer>
</body>
</html>
```

Por hacer:

Implementar una solución computacional con interfaz gráfica de usuario, la misma debe mostrar inicialmente una ventana similar a la siguiente:



- **Fase1:** En la sección indicada con el título *Documento*, existen dos formas de ingresar el documento, la primera de ellas es mediante la digitación manual del texto y la otra mediante la función de abrir un archivo de texto (formato txt) mediante la cual se abrirá el archivo y se desplegara su contenido en el área designada.

- **Fase 2:** La sección indicada como *Estructura de listas* muestra un árbol con todos los elementos de las diferentes listas. Esta representación se debe crear cuando el usuario hace click en Tokenizar. Para lo cual debe analizar el texto y crear las listas respectivas de acuerdo a lo indicado en el tema de Tokenización.

- **Fase 3:** La sección Generar Html tiene la responsabilidad de crear un archivo en el directorio actual donde se construya su contenido en formato HTML.

Fase 1:

La fase 1 tiene la responsabilidad de proveer los mecanismos para que el usuario ingrese la entrada de datos, en este caso el texto.

Su aplicación debe permitir la entrada de datos mediante dos mecanismos a saber:

1. **Entrada manual**
   Existe un control gráfico de tipo Text o similar donde el usuario ingresa el texto manualmente mediante el teclado.

2. **Entrada mediante un archivo de texto**
   El botón "Abrir archivo" debe mostrar un *Open File Dialog* que facilite la búsqueda del archivo con extensión .txt en el sistema de archivos. Se recomienda investigar el módulo tkinter.filedialog.

Fase 2:

Para efectos de esta tarea, debe considerar crear la siguiente estructura de listas:

- Una lista raíz llamada Documento que contiene a su vez 6 sublistas a saber:
  - Artículos
  - Preposiciones
  - Pronombres -
  - Verbos
  - Números
  - Tokens sin clasificar

A fin de tokenizar el texto, debe utilizar la siguiente información para determinar en qué lista se incluirá cada token:

- Artículos
  - el, la, los, las
  - Un, una, unos, unas
  - Lo, al, del

- Preposiciones
  - a, ante, bajo, cabe, con, contra, de, desde, durante, en, entre, hacia, hasta, mediante, para, por, según, sin, so, sobre, tras, versus, vía

- Pronombres
  - yo, me, mí, conmigo, nosotros, nosotras, nos, tú, te, ti, contigo, vosotros, vosotras, vos, él, ella, se, consigo, lo, la, le, los, las, les.
  - Mío, mía, míos, mías, nuestro, nuestra, nuestros, nuestras, tuyo, tuya, tuyos, vuestro, vuestra, vuestros, vuestras, suyo, suya, suyos, suyas.

- Verbos:
  - Infinitivos: verbos terminados en ar, er, ir
  - Gerundio: verbos terminados en ando, iendo
  - Participio: verbos terminados en ao, ido, to, so, cho

- Números
  - Solamente números enteros.

- Sin clasificador
  - Cualquier otro token no identificado en las categorías anteriores.

Método de tokenización

Típicamente, la tokenización ocurre a nivel de las palabras. Por lo tanto considere las siguientes indicaciones para tokenizar un archivo de texto:

I. Los signos de puntuación no deben ser incluidos en la lista de tokens.
II. Todos los caracteres alfabéticos contiguos son parte de un mismo token.
III. Los tokens son separados por espacios en blanco o signos de puntuación.
IV. Los caracteres números contiguos serán considerados como un token.

Consideraciones importantes en la Fase 2:

- Las listas deben estar ordenadas alfabéticamente de la A - Z, esto implica que deben implementar un algoritmo de ordenamiento propio.
- No es posible utilizar la función de ordenamiento de Python sort( )
- Si un token ya existe en la lista de tokens respectiva, no se debe incluir nuevamente.

Fase 3:

Esta fase es la responsable de construir la salida de los datos después realizar la tokenización y ordenamiento. El formato de salida corresponde a un archivo .html.

Este archivo debe tener dos secciones a saber:

- El contenido del archivo original en una etiqueta de párrafo.
- La tabla de análisis del documento.

Se sugiere la siguiente estructura visual para la tabla de análisis del documento:

Análisis del documento					
Artículos	**Preposiciones**	**Pronombres**	**Verbos**	**Números**	**Sin Clasificar**

El nombre del archivo html debe construirse de acuerdo al siguiente formato:

- Analisis-dd-mm-aaaa-hh-mm-ss.html
- El archivo se debe guardar en el directorio actual.

Puntos a ser evaluados:

1. Correctitud de la solución computacional - 65%

Funcionalidad	Procesos	Valor
Entrada de datos	● Despliegue en pantalla ○ Ingreso manual de datos ○ Abrir archivo	 5% 5%
Manejo de listas	● Tokenizar la entrada ○ Crear las listas indicadas de acuerdo a los criterios dados ○ Despliegue en el TreeView	 25% 15%
Ordenamiento de la listas	● Algoritmo de ordenamiento alfabético	30%
Archivo HTML	● Construir un archivo html que contenga una tabla con 6 columnas, en cada columna se debe colocar la lista exhaustiva de tokens que corresponden. Por ejemplo, en la columna 1, Artículos, deben incluirse todos los tokens que se encuentran en la lista de artículos.	20%

2. Robustez de la solución computacional (validaciones) - 10%

3. Evitar los síntomas de un diseño pobre “olores del software” - 5%
    a. Rigidez
    b. Fragilidad
    c. Inmovilidad
    d. Viscosidad
    e. Complejidad innecesaria
    f. Repetición innecesaria
    g. Opacidad

4. Entregar un documento con al menos los siguientes apartados: - 20%
    a. Manual de usuario - 25 puntos
    b. Pruebas de funcionalidad - 25 puntos

Debe demostrar evidencia de todas las funcionalidades implementadas en la tarea programada. Para *esto* realice la ejecución de su tarea programada y tome **screenshots** dejando evidencia del resultado de la ejecución de cada transacción.

c. Lecciones aprendidas - 15 puntos

Debe hacer un listado de todas las lecciones aprendidas producto del desarrollo de la tarea programada. Las lecciones aprendidas pueden ser de carácter personal y/o técnico.

d. Olores del software - 25 puntos

Debe indicar cuál o cuáles estrategias se usaron para apartarse de cada uno de los olores del software vistos en clase. Las estrategias deben sustentarse con segmentos de código de la tarea programada (tome screenshots) o bien con la documentación de las decisiones que fueron tomadas durante el diseño y la implementación.

e. Minutas y evidencias de asignación de responsabilidades - 10 puntos

Debe indicar las fechas y tiempos de cada una de las reuniones que realiza el equipo asi como donde se realizan. También los temas tratados, las responsabilidades que son asignadas a los miembros del equipo y el seguimiento de los retos encontrados. Incluir un posible cronograma de cómo distribuir el trabajo a lo largo del tiempo asignado.

5. Debe entregar un CD/DVD, que contenga únicamente 2 carpetas llamadas: documentación y solución computacional, en la primera deberá incluir el documento de Word (no pdf) solicitado y en la segunda los archivos y/o carpetas necesarias para la implementación de esta tarea.

6. Entregue el CD/DVD en un sobre de manila sellado con sus calidades. La entrega es estrictamente personal en el momento que el profesor lo haya dispuesto.

a. Las calidades deben incluir:

i. Nombre del curso
ii. Número de semestre y año lectivo
iii. Nombre del Estudiante
iv. Número de carnet
v. Número de tarea programada
vi. Fecha de entrega
vii. Estatus de la entrega (definido por el responsable de la implementación de la tarea): [Deplorable|Regular|Buena|Muy Buena|Excelente|Superior]

7. La tarea será revisada en la versión de Python 3.5.1 únicamente.

Condiciones generales:

Esta tarea programada se rige por las siguientes condiciones:

Nota: El incumplimiento de alguna condición implicará una calificación de cero.

1. El desarrollo de la tarea es estrictamente en PAREJAS
2. La tarea DEBE implementarse con interfaz gráfica.
3. Debe cumplir con todo lo indicado en la sección “Puntos a ser evaluados”
4. Deberá entregarse en tiempo y forma según el plazo establecido por el profesor al momento la lectura de este documento.
5. El lenguaje de programación a utilizar es Python v3.5.1
6. Debe combinar la programación recursiva e iterativa para dar solución a esta tarea.
7. Se cuenta con 3 semanas a partir del día de entrega de la tarea.

Anexo 1: Manual de Usuario

1. Portada
2. Introducción
3. Qué funcionalidades implementadas tiene el software (estado general de la tarea)
4. Explicación paso a paso de cómo probar cada uno de los algoritmos implementados, esta explicación debe incluir el uso de casos de pruebas.
5. Pendientes de implementar y su justificación.
6. Bibliografía y Fuentes Digitales utilizadas para su investigación