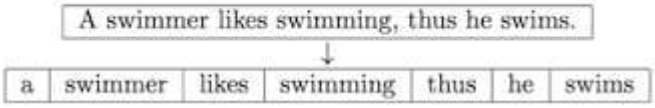
I Tarea Programada

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 cuándo 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 información con respecto a la sintaxis y uso de expresiones regulares en Python 3.5.1, puede consultar el siguiente vínculo: 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

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>
```

XML

Es un estándar ampliamente soportado para describir datos. XML es comúnmente usado para intercambiar datos entre aplicaciones sobre internet. Permite crear marcas para virtualmente cualquier tipo de información, lo cual posibilita la creación de nuevos lenguajes de marcas para describir cualquier tipo de datos, como fórmulas matemáticas, música, noticias, recetas, reportes financieros, entre muchos otros.

Una de las características más importantes de XML es que describe los datos de forma tal que sean entendibles tanto para los humanos como para las computadoras.

A continuación se detalla un ejemplo de un documento XML:

```
<Books>
    <Book ISBN="0553212419">
        <title>Sherlock Holmes: Complete Novels...
        <author>Sir Arthur Conan Doyle</author>
    </Book>
    <Book ISBN="0743273567">
        <title>The Great Gatsby</title>
        <author>F. Scott Fitzgerald</author>
    </Book>
    <Book ISBN="0684826976">
        <title>Undaunted Courage</title>
        <author>Stephen E. Ambrose</author>
    </Book>
    <Book ISBN="0743203178">
        <title>Nothing Like It In the World</title>
        <author>Stephen E. Ambrose</author>
    </Book>
</Books>
```

Todo documento XML está compuesto de elementos que especifican la estructura del documento. Algunas de sus características son las siguientes:

- Los documentos XML delimitan los elementos con marcas o etiquetas de inicio y fin. Una marca de inicio consiste del nombre del elemento entre corchetes angulares. Ejemplo: **<author>**
- Una marca de cierre consiste del nombre del elemento precedido por un forward slash (/) entre los corchetes angulares. Ejemplo: **</author>**
- Las etiquetas inicial y final de un elemento encierran el texto que representa los datos Ejemplo: **<author>Stephen E. Ambrose</author>**
- Cada documento XML debe tener exactamente un elemento raíz que contiene todos los demás elementos Ejemplo: **<Books>**

Para mayor información con respecto a este tema puede acceder al siguiente recurso: http://www.w3schools.com/xml/

Por hacer:

Implementar una solución computacional ***sin*** interfaz gráfica, es decir, ejecutándose desde el Shell de Python.

- **Fase1:** El programa debe abrir un archivo de texto (formato txt) el cual será analizado para desarrollar todos los procesos esperados. La fase 1 tiene la responsabilidad de proveer los mecanismos para que el usuario ingrese la entrada de datos, en este caso el texto.

- **Fase 2:** Mostrar un menú que permita tokenizar el texto. Para lo cual debe analizar el texto y crear las listas respectivas de acuerdo a lo indicado en el tema de Tokenización. Cuide el manejo de errores en caso de no encontrarse el archivo deseado.

- **Fase 3:** Sólo si el archivo logra tokenizarse se podrán habilitar las siguientes opciones del menú en el directorio actual:
    - Generar Html.
    - Generar un XML.
    - Generar Binario.

- **Fase 4:** Terminar.

  De lo contrario debería indicar el error e iniciar nuevamente el proceso.

Detalle de la 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 y vía; algunas de ellas, en la actualidad, han entrado en desuso: cabe y so.
- Pronombres
    - yo, me, mí, conmigo, nosotros, nosotras, nos, tú, te, ti, contigo, vosotros, vosotras, vos, él, ella, se, consigo, le, 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 ado, 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: Salida de Datos.
Esta fase es la responsable de construir la salida de los datos después realizar la tokenización y ordenamiento.

- ***Generar Html.***
  El formato de salida corresponde a un archivo .html.

Este archivo debe tener tres secciones a saber:

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

Debe crearse la siguiente estructura visual para la tabla de análisis del documento:

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

Considere los datos deben mostrarse ordenadamente.

El **reporte** debe indicar:
El texto original tiene X tokens de los cuales hay: __ artículos, __ preposiciones, __pronombres, __verbos, __números y __ 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.

- ***Generar un XML.***

Debe crear un XML del **Documento** leído y sus 6 **partes**: artículos, preposiciones, pronombres, verbos, números y sin clasificar. Adicionando su **contenido** según corresponda.
Se espera entonces, algo similar a:

```
<Documento>
    <Parte Seccion=”Artículos”>
        <Contenido>la,unos,al</Contenido>
    </Parte>
    <Parte Seccion=”…”>
        <Contenido>…</Contenido>
    </Parte>
</Documento>
```

Asigne el nombre que guste al archivo con extensión .xml

- ***Generar Binario.***

Debe crearse un archivo binario que contenga estructura de listas de la ***Fase 2.*** Dé el nombre de BD al archivo. Posteriormente muestre en el Shell todo el contenido binario, pero anteponga a cada salida el título de cada “parte sección” del documento. Usted es libre de mostrar las salidas cómo guste.

Fase 4: Terminar.

Da las gracias por usar el sistema y se sale del menú reiterativo.

Puntos a ser evaluados:

1. Correctitud de la solución computacional - 80%

Funcionalidad	**Procesos**	**Valor**	Recomiendo
Entrada de datos	Lectura y control de errores del archivo de texto original	2%	E1
Manejo de listas	Tokenizar la entrada Crear las listas indicadas de acuerdo a los criterios dados. Son 6 sublistas (5% c/u)	32%	E2
Ordenamiento de la listas	Algoritmo de ordenamiento alfabético	20%	E1
Archivo HTML	Con las secciones solicitadas: El contenido del archivo original en una etiqueta de párrafo. La tabla de análisis del documento. Reporte Dar nombre al archivo	 2% 16% 3% 2%	E1
Generar el XML	Crea el XML con 6 secciones y su respectiva información. Se guarda con extensión .XML	18%	E2
Generar el Binario	Debe crearse un archivo binario que contenga estructura de listas de la Fase 2. Muestre contenido: anteponga a cada salida el título de cada “parte sección” del documento.	3%	E2
Menú	Menú recurrente y salir	2%	E1

1. Eliminación de olores de software y buenas prácticas en programación - 5%
2. Robustez de la solución computacional (validaciones) - 5%
3. Entregar un documento con los siguientes apartados: - 10%

REQUISITO PARA REVISAR EL PROYECTO

El requisito consiste en presentar la documentación del proyecto indicada en esta sección.

La nota de la documentación del proyecto sirve para aceptar o rechazar el proyecto: se revisan los proyectos que cumplan con este requisito en un 90% o más.

Enviar vía Tec Digital, sección EVALUACIONES, en la carpeta TP1, una carpeta comprimida (.rar, .zip, etc.) que contenga las siguientes partes:

- Parte 1: Una carpeta con la Documentación del proyecto (**nombre: documentación_códigos.PDF**).
  - Portada. (2 p)
    i. Nombre del curso
    ii. Número de semestre y año lectivo
    iii. Nombres de los Estudiantes y números 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]
  - Índice. (2 p)
  - Enunciado del proyecto. Hacer referencia a un archivo adjunto. (1 p)
  - Justificación de eliminación de olores (párrafo descriptivo o ejemplos) (14 p)
  - Conclusiones del trabajo: (22 p)
    - Al menos 6 problemas encontrados y soluciones a los mismos (Plantilla_Bitacora_ResoluciónProb.xls). 6 p
    - Aprendizajes obtenidos.
      - Debe hacer un listado de todas las lecciones aprendidas producto del desarrollo de la tarea programada. Las lecciones aprendidas deben ser 8 de carácter personal y 8 de carácter técnico. Es decir 4 aportados por cada uno según corresponda.
    - Reglamento de trabajo (1 p)
    - 2 Agendas (5 p), 2 Minutas (5 p), y evidencias de asignación de responsabilidades: cronograma - (5 p).

- Estadística de tiempos (8 p): un cuadro que muestre el detalle de las actividades que realizó y las horas invertidas en cada una de ellas. La estadística permite medir el esfuerzo dedicado al trabajo en términos de actividades y tiempos, lo cual puede ser una base para calcular el esfuerzo requerido en futuros trabajos. No olvide investigar sobre el Personal Software Process (PSP) y sus implicaciones como programador.

  Ejemplos de actividades:

Actividad Realizada	**Horas**
Análisis de requerimientos	
Diseño de algoritmos	
Investigación de ...	
Programación	
Documentación interna	
Pruebas	
Elaboración del manual de usuario	
Elaboración de documentación del proyecto	
Etc.	
TOTAL	

- Manual de usuario (**nombre: manual_de_usuario_codificar.PDF**). (35 p)

  Es un documento de comunicación técnica utilizado para guiar a las personas que usan el software. Explica paso a paso cómo usar cada una de las funcionalidades del programa. Apóyese en imágenes, capturas de pantallas, menús, diagramas y los aspectos que considere van a servir como una guía útil para que el usuario pueda usar el programa. Puede tomar como referencia algún manual de usuario de alguna aplicación.

- Parte 2: Una carpeta con el Programa fuente.

Condiciones generales:

Esta tarea programada se rige por las siguientes condiciones:
La tarea debe solucionarse usando listas de strings o no se revisa la tarea.

1. El desarrollo de la tarea es estrictamente en grupos de 2 estudiantes, si hay cambio de alguna pareja debe notificarse antes de mañana a las 12md al correo de la profesora lsarmiento@tec.ac.cr
2. La tarea NO 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 o superior.
6. Debe crear programación iterativa para dar solución a esta tarea.
7. Se cuenta con 3 semanas a partir del día de entrega de la tarea.
    1. Fecha de entrega al TEC Digital: sábado 9 de octubre del 2021, antes de las 11:45 pm.
8. Debe presentarse el grupo completo a defender la tarea, en caso de no asistir, tendrá nota de 0 en el valor total de la tarea.
9. Cada miembro debe realizar a conciencia la evaluación de Habilidades Blandas y entregarla en digital antes de la revisión de la tarea programada.
10. La nota final de cada estudiante se asignará por distribución de esfuerzo y cumplimiento en función a las responsabilidades y la nota obtenida.

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

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.