**Manual de Usuario**

Instituto Tecnológico de Costa Rica

Ingeniería en Computación

-Curso:

Taller de Programación

-Docente:

Luis Pablo Soto Chaves

-Estudiantes:

José Navarro Acuña

Josué Suárez Campos

-Segunda tarea programada:

Tokenización

1er Semestre, 2016.

# **Introducción**

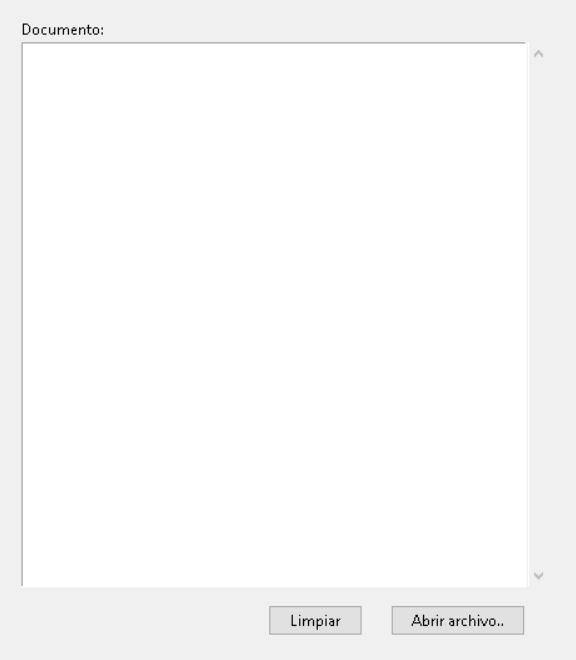
El presente documento es un manual de usuario de un software que permite clasificar los elementos de una oración y mostrarlos en pantalla

Este manual de usuario está compuesto por secciones. La sección 2 muestra cómo se puede utilizar el software y las funciones de cada herramienta.

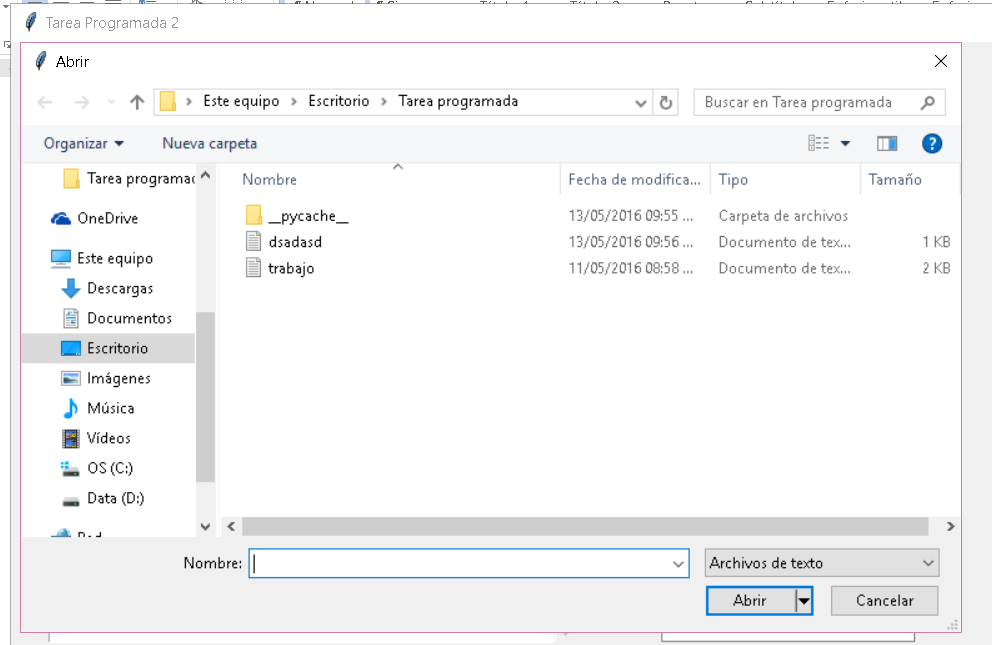
El software contiene una sección que explica el proceso y la codificación que sigue el software para determinar los resultados requeridos (Sección 3). Por último se muestra en la sección 4 las fuentes bibliográficas y digitales utilizadas para desarrollar este proyecto.

# **Funcionalidades del software.**

El software tiene como función y propósito el recibir un documento o escrito del cuadro de texto, y clasificar este en los elementos que forman una oración, entre ellos, verbos, artículos, preposiciones, números, pronombres y los elementos que no pudieron ser clasificados. Para hacer esto ejecute el software y en la sección Documento escriba el texto que desea descomponer, en dado caso que tenga un archivo.txt utilice el botón Abrir Archivo y busque el documento.



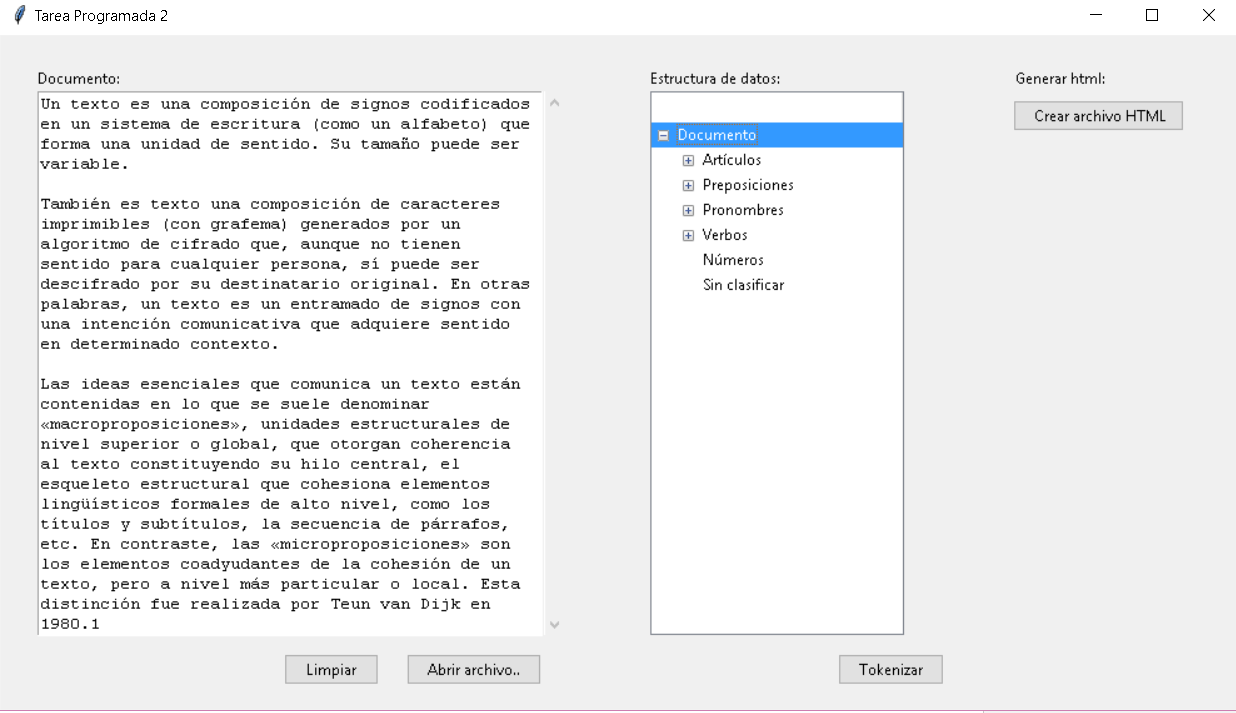
*Figura 2.1*



*Figura 2.2*

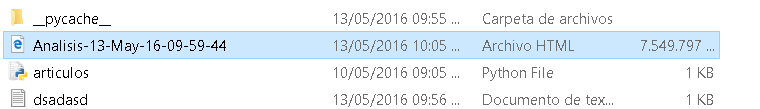
Luego de abrir el documento, este se muestra en la caja de texto como en la figura. Presione el botón Tokenizar para descomponer el texto en los elementos de una oración. Esta información la comparte en un árbol de clasificación donde vienen las palabras ordenadas según categorías como en la figura se aprecia.

El software los divide en artículos, preposiciones, pronombres, verbos, números y sin Clasificación. Puede presionar el botón + al lado de la categoría para ver los elementos dentro de esta.



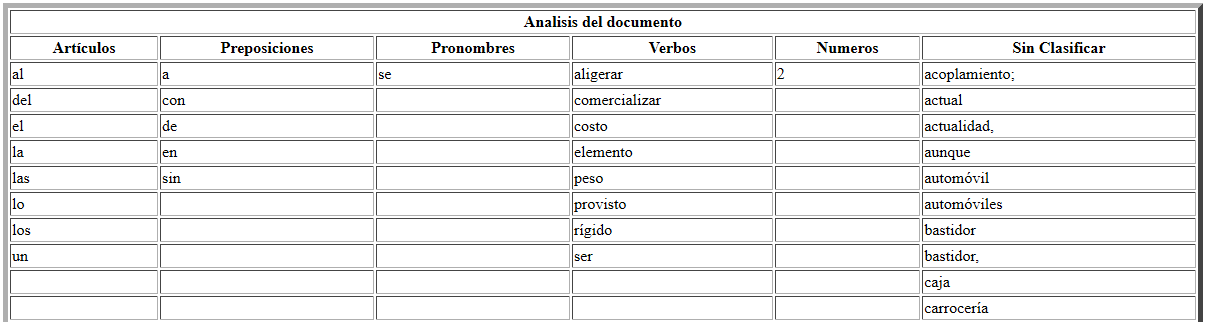
*Figura 2.3*

Si el usuario lo desea, existe la posibilidad de crear un archivo Hyper Text tMarkup Languaje (HTML) para que el navegador web interprete la codificación y muestre el resultado del análisis del texto en una tabla clasificada. Para ello diríjase al botón Crear archivo HTML, este enviará el mensaje que se ha creado con éxito. Seguidamente diríjase a la carpeta donde se encuentra el software y busque el archivo con extensión HTML llamado Análisis (fecha/hora)



*Figura 2.4*

La información se verá así:



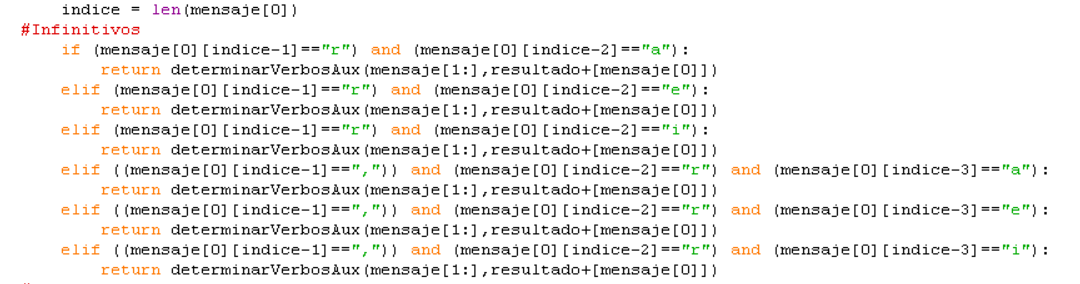
# **3. Explicación Algoritmos**

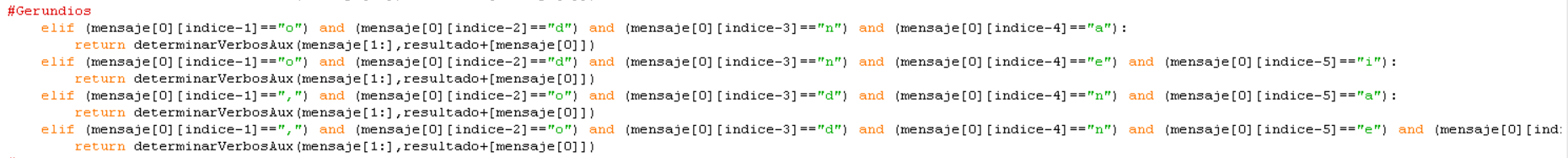
El software contiene 6 funciones dedicadas al tratamiento de las palabras de un texto y realizar su clasificación.

**Función de verbos:**

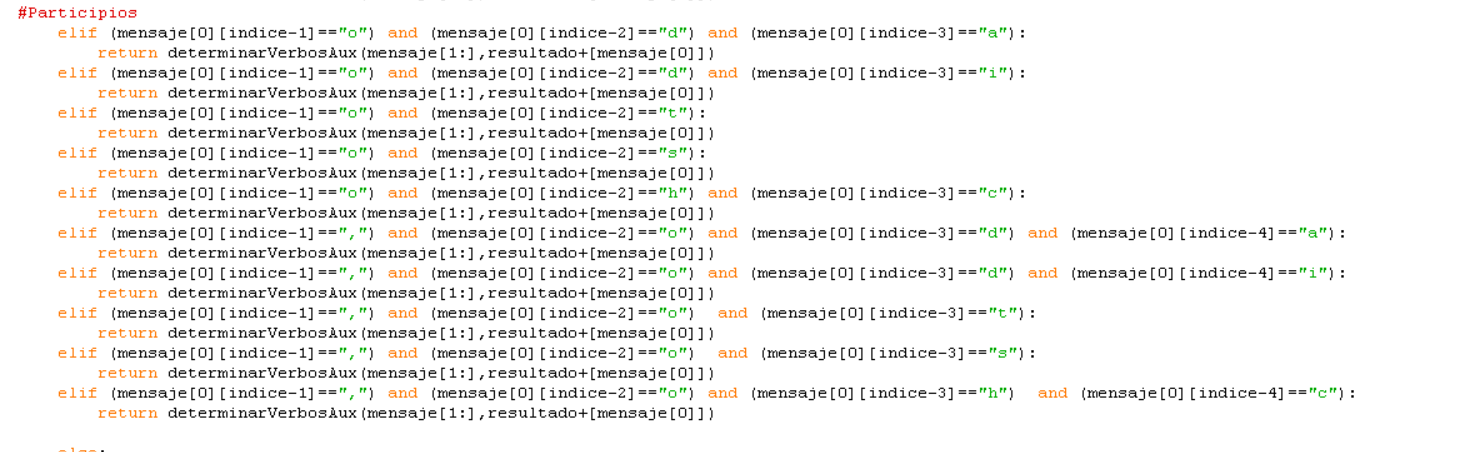
En la lengua española los verbos terminan en infinitivos, participios o gerundios. Esta función recursiva utiliza las características que tienen las 3 categorías de verbos para encontrarlos inicializando la variable índice que toma el valor de la función tamaño de la primera palabra del mensaje y el uso de condiciones If anidados.

Primero se evalúan los casos de si la palabra es un verbo infinitivo. Un verbo infinitivo es aquel que termina en ar, er, ir. Si la palabra cumple una de esas condiciones debe enviar una llamada recursiva a la misma función indicando que la palabra es un verbo.

*Figura 3.1*

El segundo caso que una palabra sea un verbo es ser un verbo gerundio. Un gerundio es aquel verbo cuya terminación es ando y iendo.

*Figura 3.2*

Y el último caso de que una palabra sea un verbo es ser un participio cuya terminación debe ser to so cho ado, ido.

*Figura 3.3*

En dado caso que no cumpla esas condiciones, debe de hacer una llamada recursiva a la misma función con el resto del mensaje y sin ningún cambio en la variable resultado.

Al enviar una palabra a esta función, por ejemplo la palabra comer.

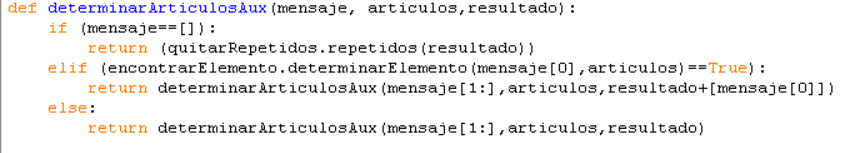
La variable índice tomaría el valor de 5, que es el tamaño de la palabra. Comenzaría el proceso de calificar las últimas letras de la palabra para comprobar si es un verbo.

La palabra comer cumple la condición de si su índice -1 es ‘r’ y su índice -2 es ‘e’, es un verbo infinitivo por lo que haría una llamada recursiva a la función con el resto del mensaje y en el resultado se añade la palabra comer. Así hasta que el mensaje sea vacío.

**Función Artículos**

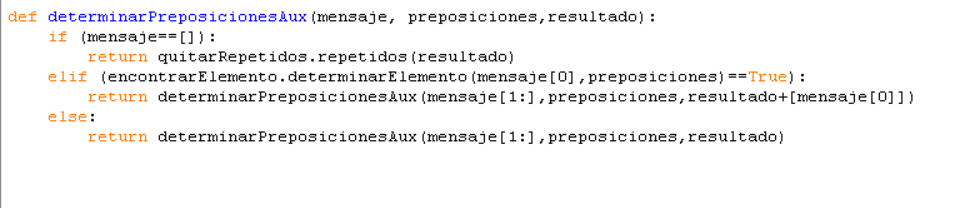
La responsabilidad de esta función es encontrar los artículos existentes dentro del mensaje. La función determinarArticulos(mensaje) realiza una llamada recursiva a una función auxiliar que recibe el mensaje, los artículos existentes en la lengua española y la variable donde se irán almacenando los artículos.

Para averiguar si una palabra mensaje [0] corresponde a un artículo de utiliza la función encontrarElemento *(Figura 3.10)* enviando los artículos y el mensaje [0], esta función comparará la palabra con cada uno de la cadena de artículos. Si la palabra corresponde a alguno de ellos retornará el valor booleano True. En la función principal, al saber el resultado True sumará el mensaje [0] a su variable resultado. En dado caso que no corresponda a un artículo enviará False, con lo que la variable resultado no sufre cambios.



*Figura 3.4*

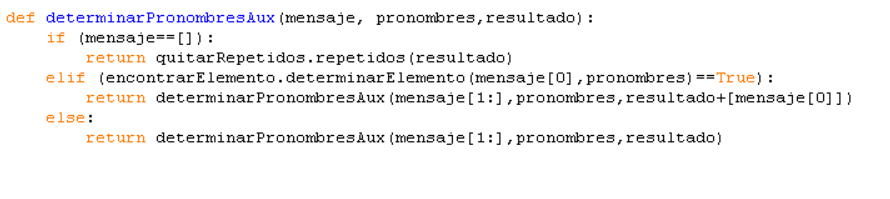
**Función Preposiciones**

Como en las anteriores se utiliza variable que almacena en lista todas las preposiciones. El procedimiento por realizar al comparar una palabra con una preposición es utilizar la función determinarElemento *(Figura 3.10)* utilizada anteriormente para determinar si la palabra se encuentra en la lista preposiciones. Si la función retorna True se suma esa palabra a la variable resultado, en caso que no lo cumpla se sigue haciendo una llamada recursiva a la misma función sin alterar la variable resultado.

*Figura 3.5*

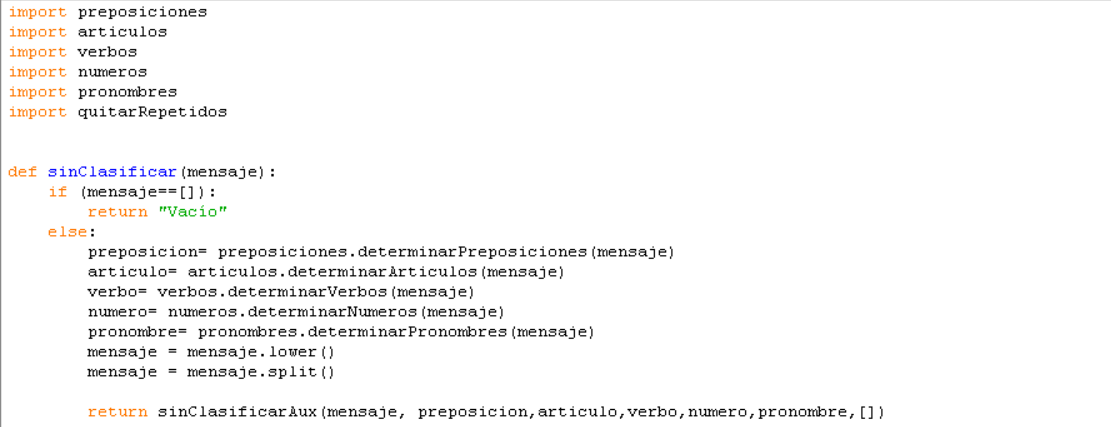
**Función Pronombre**

Se utiliza la misma técnica que las demás, una variable Pronombre que almacena una lista con todos los pronombres. La función recursiva tiene un If que hace un llamado a la función determinarElementos *(Figura 3.10)* y compara la primera palabra del mensaje con el diccionario. Si retorna True se agrega al resultado, sino, se hace una llamada a sí misma sin alterar el resultado.



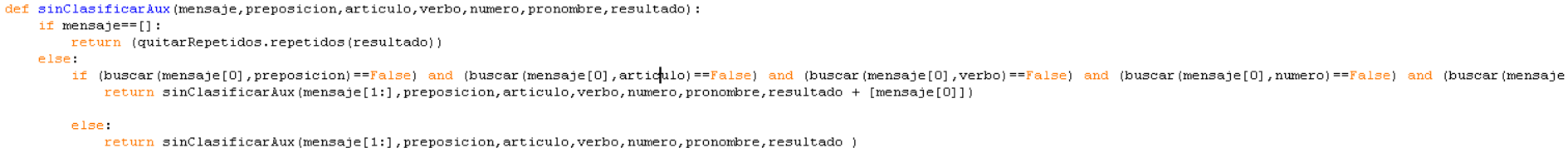
*Figura 3.6*

**Función sin Clasificar**

Esta función funciona un poco diferente a las demás. La función realiza una importación de las demás funciones (artículos, verbos, pronombres, preposiciones. Números) y sus resultados los ingresa dentro de distintas variables. Con ello realiza una llamada recursiva a una función auxiliar.

*Figura 3.7*

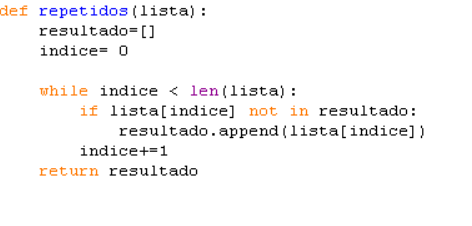
El proceso de la función auxiliar es realizar una comparación If de la primera palabra del mensaje y los demás resultados de las funciones por medio del uso de la función determinarElemento *(Figura 3.10)*, enviando la primera palabra del mensaje como elemento y el resultado de las funciones como lista comparativa.



*Figura 3.8*

**Función Quitar Repetidos**

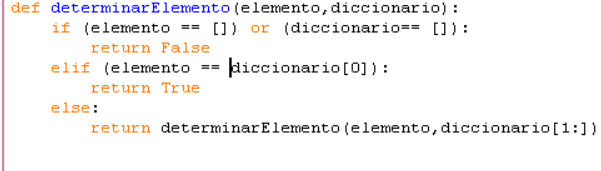
El propósito de esta función iterativa es quitar los elementos repetidos de los resultados de todas las funciones (artículos, verbos, preposiciones. números, pronombres, sin Clasificar) mediante el uso de la variable índice que nos permite indicarle a la lista el elemento a comparar y la variable resultado que irá acumulando los elementos sin repetir. Además se utilizó una función integrada en el If: not in. Esta función permite comparar un elemento con una lista y verificar si el elemento a comparar está dentro de esta lista o no. Al comparar el primer elemento de lista con la variable resultado que se encuentra vacía, este se irá sumando y los demás elementos que no se encuentren anteriormente en la lista resultado también se irán incorporando. En dado caso que no cumpla la condición el índice se aumenta normalmente y sigue el procedimiento sin agregar ese elemento a la variable resultado.



*Figura 3.9*

**Función determinar Elementos**

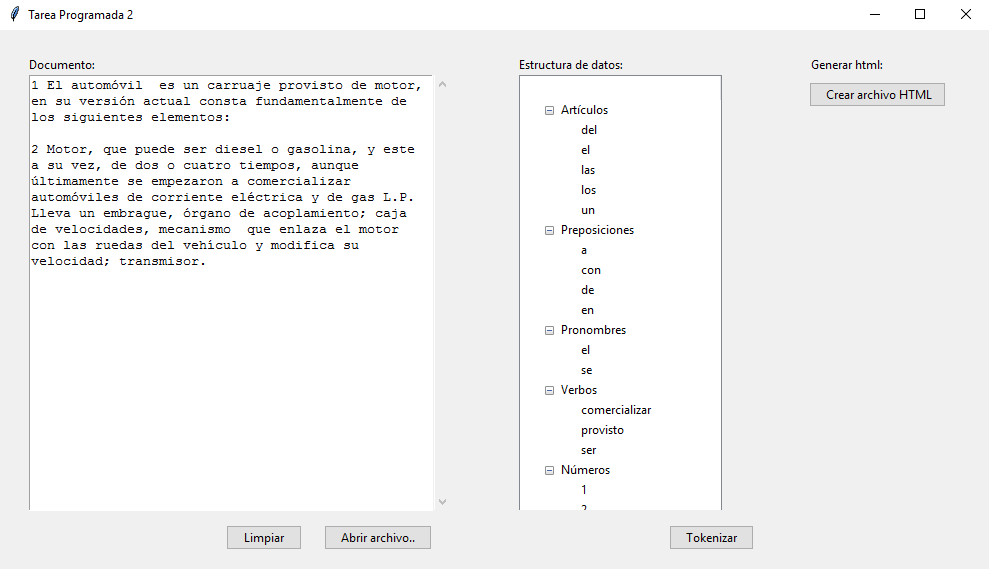
Esta función está presente en la mayoría de las funciones. Tiene como propósito comparar la variable elemento, que sería el mensaje [0] de cada una de las funciones, con la variable diccionario que es la variable donde se envían las listas de elementos de las funciones que se utilizan como diccionarios almacenando los resultados correctos. El procedimiento que sigue esta función es comparar el elemento con el primero de la variable diccionario. Si se cumple la igualdad se retorna el valor booleano True, en caso que se dé lo contrario, se realiza una llamada recursiva a la misma función con el elemento y el resto del diccionario quitando su primera palabra [1:]. En caso que el elemento diccionario se quede sin palabras, se retorna el elemento False como respuesta a que no se encontró el elemento en la lista diccionario.



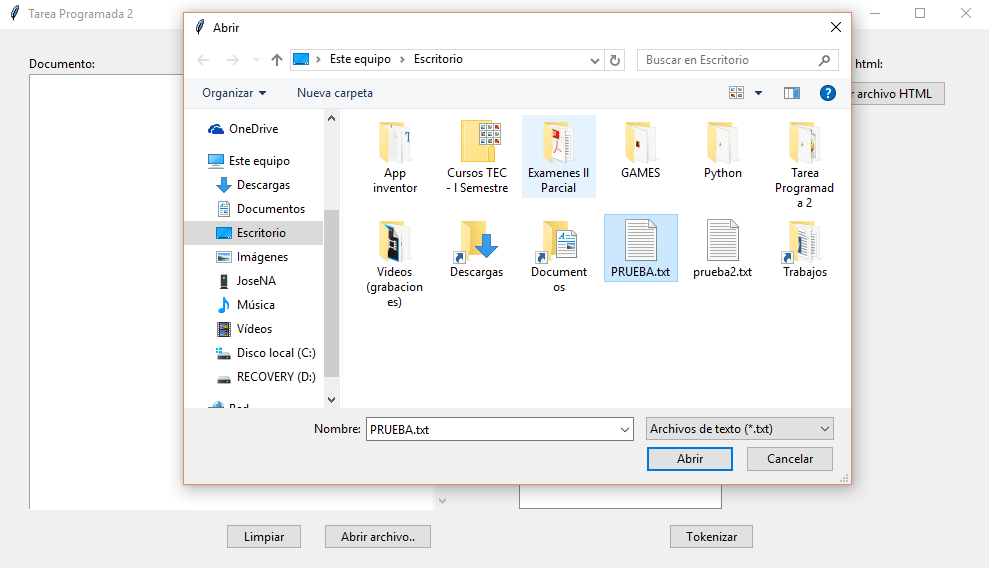
*Figura 3.10*

**4. Pruebas de funcionalidad**

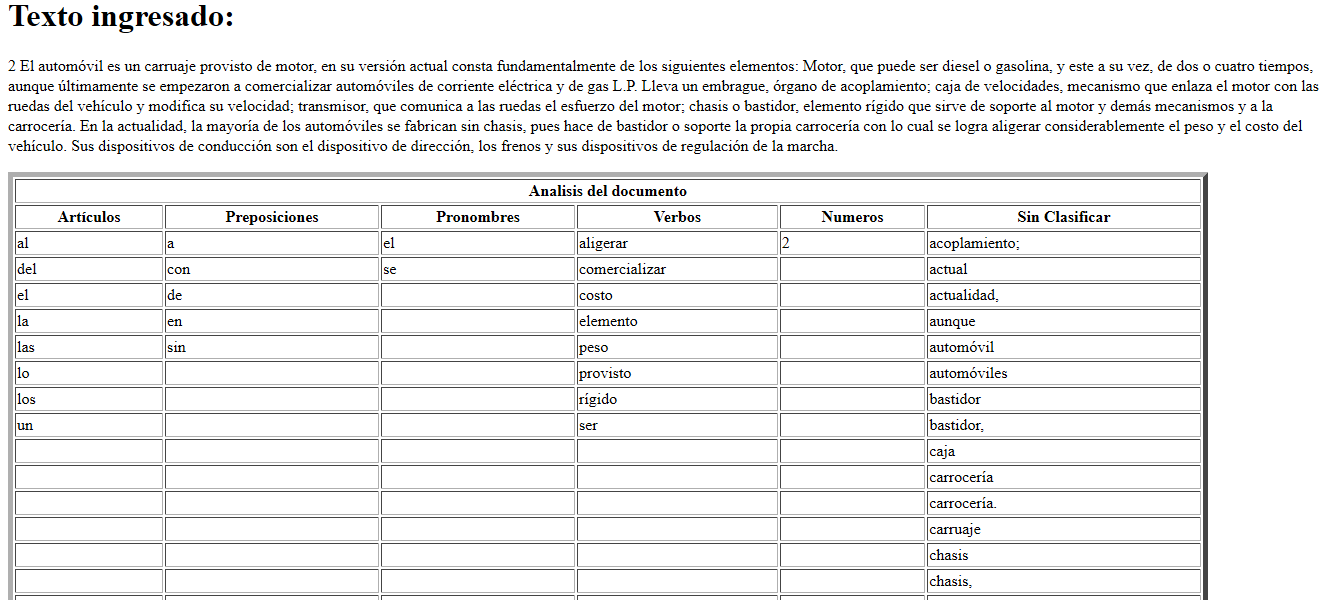
-Tokenización:



-Botón: “Abrir archivo..”:



-Generar Html:



**5. Lecciones aprendidas**

-El uso de la interfaz gráfica Tkinter en Python (Botones, etiquetas, cuadros de texto, treeView, entre otros).

-El uso de los comandos “for” e “in” para desplegar los resultados de forma vertical (Nos fue muy útil para imprimir los resultados en el treeView).

-La apertura de archivos de textos para la tokenización (“Abrir archivo”).

-El manejo de listas de diferentes maneras (Por ejemplo: el manejo de variables dentro de funciones distintas, para luego obtener el resultado en forma de lista).

-La creación de archivos (html) en base a los resultados obtenidos en la tokenización.

-La creación de estructuras html, basándose en el resultado obtenido en la tokenización.

**6.**  **Estrategias utilizadas para apartarse de los olores de software**

-Utilizamos nombres significativos en todas nuestras funciones, así evitamos la opacidad. Por ejemplo:

olores2olores3olores

-Comentamos gran parte de nuestro código, lo cual facilita aún más el entendimiento de las funciones.

comentarcomentar2

-Creamos varias funciones, que posteriormente utilizamos mediante el método “import” en la mayoría de las funciones, lo cual evitamos la repetición de código.

import

Luego:

usarImport

-En el código de la interfaz gráfica, realizamos cada algoritmo de manera clara y siguiendo un orden lógico de cada función y procedimiento.

-Aprendimos a utilizar los parámetros “is not” como una forma de iterar en una lista determinar más fácilmente.

# **7. Bibliografía**

* Python Software Foundation. 25. Graphical User Interfaces with Tk. <https://docs.python.org/3/library/tkinter.ttk.html>
* Tutorials Point. Python GUI Programming (Tkinter)

<http://www.tutorialspoint.com/python/python_gui_programming.htm>

* TkDocs. Tutorial. Tree.

<http://www.tkdocs.com/tutorial/tree.html>

* Libros Web. Python

<http://librosweb.es/libro/python/>