UNIVERSIDAD TECMILENIO
Innovación que transforma vidas.

Reporte

Nombre: Rodrigo Ley Muñoz Angel Santiago Delgado Mendoza Kevin Adrián Pizaña Ayala Jessica Sanchez Tamez	**Matrícula:** 2845996 2959951 2953809 2730241
Nombre del curso: Proyecto Integrador de Software.	**Nombre del profesor:** Ana Carolina González d'Hyver de las Deses
Módulo: Fase 3	**Actividad:** Actividad 9 - Stop list.
Fecha: 08/Marzo/2024	
Bibliografía: (08/Marzo/2024). Fase 3. Weight tokens. Canvas. TecMilenio Recuperado de https://cursos.tecmilenio.mx/courses/157573/pages/mi-curso?module_item_id=587088 Jessica S. T. (09/Marzo/2024). Actividad 6. Sumar la misma palabra por archivos. Tecmilenio, Recuperado de file:///D:/UNI%20--%2012vo%20Semestre%20IDS%20-%20Ene-May%202024/Proyectos%20de%20Ingenier%C3%ADa%20de%20Software/Fase%202%20-%20PIdS/Actividad%206%20-%20PIdS/Actividad%206%20-%20PIdS.pdf	

Objetivo:

- Crear Identificar factores externos y calidad interna del sistema.
- Refinar el diccionario.

Requerimiento:

- Descargar el archivo stop list.

Procedimiento:
Resultados:

En esta actividad eliminarás las palabras del diccionario; estas palabras, conocidas como stop list, son las más frecuentes en el idioma inglés y decidirás si eliminarlas antes o después de crear el diccionario. Para esto, primero identificarás los factores externos y calidad interna del sistema, después refinarás el diccionario mediante la implementación de un stop list.

Identificar factores externos y calidad interna del sistema

1. Tú y tu equipo scrum identifiquen y justifiquen tres.
2. Creen las historias de usuario necesarias para el sprint en curso.

Refinar diccionario

3. Descarguen el archivo de *stop list* y realicen los cambios necesarios al programa para que el diccionario no contenga ninguna de esas palabras.

4. Eliminen, no procesen los siguientes tokens:
   a. Los que tengan frecuencias bajas, es decir, los que tengan repeticiones muy bajas de acuerdo a su criterio.
   b. Los que cuenten con longitud de uno, es decir, todas las palabras contenidas en el diccionario de una letra o un dígito deben ser eliminadas.

5. Agreguen un archivo log de salida con nombre `a9_matricula.txt`.

6. Cronometren el tiempo que tarda el programa en procesar cada archivo y todo el proceso.

7. Incluyan el archivo log de salida, tomen como base la siguiente figura.

```
541  c:\cs13309\files\001.html    0.01
542  c:\cs13309\files\002.html    0.01
543  c:\cs13309\files\003.html    0.05
544  ...
545  ...
546  ...
547  c:\cs13309\files\500.html    0.11
548  c:\cs13309\files\501.html    2.01
549
550  Tiempo total de ejecucion del programa: 45 segundos
```

Código del Programa:

```
import os
import time
import string
from collections import defaultdict

# Ruta de la Carpeta de los Archivos HTML sin las etiquetas html.
input_folder_path = "C:\\Users\\100100050\\Desktop\\output"


#-------------------------
#    PARTE 1 - TOKENS    |
#-------------------------


# Ruta del Archivo log de salida para los Tokens.
log_file_path = "C:\\Users\\100100050\\Desktop\\txt9\\Token;Repeticiones;Num_Archivos.txt"

# Verificar si la carpeta de salida existe
if not os.path.exists(input_folder_path):
    print("...La carpeta de salida no existe...")
    exit()

# Iniciar Temporizador Total
total_timer_start = time.time()

# Diccionario para Almacenar los Tokens y sus Repeticiones en Todos los Archivos.
token_counts = defaultdict(int)

# Diccionario para Contar el Número de Archivos que Contienen cada Token.
file_counts = defaultdict(int)


# Iterar sobre Todos los Archivos HTML en la Carpeta de Salida.
for file_name in os.listdir(input_folder_path):
    if file_name.endswith(".html"):
        input_file_path = os.path.join(input_folder_path, file_name)

        # Leer el Contenido del Archivo HTML y Obtener Todas las Palabras.
        with open(input_file_path, "r") as input_file:
            words = input_file.read().split()

        # Contar las Repeticiones de Cada Palabra en el Archivo Actual.
        word_counts = defaultdict(int)
```

```
            for word in words:
                word_counts[word.lower()] += 1

        # Actualizar el Diccionario de Recuentos de Tokens.
        for word, count in word_counts.items():
            # Agregar Solo Tokens que NO sean de 1 letra o 1 dígito y Tengan Frecuencia Mayor a 20.
            if len(word) > 1 and not word.isdigit() and count >= 20:
                token_counts[word] += count

        # Actualizar el Diccionario de Recuentos de Archivos para Cada Token.
        for word in set(words):
            file_counts[word.lower()] += 1

# Crear el Archivo de log de salida "Token;Repeticiones;Num_Archivos.txt".
with open(log_file_path, "w") as log_file:
    # Escribir la Tabla de Tokens, Repeticiones y Archivos en el Archivo
"Token;Repeticiones;Num_Archivos.txt".
    log_file.write("Token;Repeticiones;Num_Archivos con el mismo Token\n")
    for token, count in sorted(token_counts.items()):
        num_files_with_token = file_counts[token]
        log_file.write(f"{token};{count};{num_files_with_token}\n")

    # Calcular y Escribir el Tiempo Total de Ejecución de la 1ra Parte del Programa en el Archivo
"Token;Repeticiones;Num_Archivos.txt".
    total_program_time = time.time() - total_timer_start
    log_file.write(f"\n\nTiempo Total de Ejecución del Programa: {total_program_time:.6f}
segundos\n")



#-----------------------------------------
#    PARTE 2 - DICCIONARIO DE PALABRAS    |
#-----------------------------------------

# Función para ELIMINAR Números y ELIMINAR Caracteres Especiales (Excepto la comilla simple (') ).
def clean_word(word):
    return ''.join(char for char in word if char.isalpha() or char == "'")


# Ruta del Archivo log de Salida para el Diccionario.
dictionary_file_path = "C:\\Users\\100100050\\Desktop\\txt9\\diccionario.txt"

# Ruta del Archivo log de salida para las Palabras Eliminadas
deleted_words_file_path = "C:\\Users\\100100050\\Desktop\\txt9\\palabras_eliminadas.txt"

# Ruta del Archivo log de salida para el Tiempo de Procesamiento.
processing_time_file_path = "C:\\Users\\100100050\\Desktop\\txt9\\a9_matricula.txt"


# Verificar si la Carpeta de salida Existe.
if not os.path.exists(input_folder_path):
    print("...La carpeta de salida no existe...")
    exit()


# Iniciar Temporizador Total
total_timer_start = time.time()

# Diccionario para Almacenar Todas las Palabras en los Archivos HTML.
all_words = set()


# Iterar sobre Todos los Archivos HTML en la Carpeta de Salida.
for file_name in os.listdir(input_folder_path):
    if file_name.endswith(".html"):
```

```
        input_file_path = os.path.join(input_folder_path, file_name)

        # Iniciar Temporizador para el Archivo Actual.
        file_timer_start = time.time()

        # Leer el Contenido del Archivo HTML y Obtener Todas las Palabras.
        with open(input_file_path, "r") as input_file:
            text = input_file.read()
            # Eliminar Números y Caracteres Especiales, Luego Dividir por Espacios para Obtener
Palabras.
            words = text.translate(str.maketrans('', '', string.digits)).translate(str.maketrans('',
'', string.punctuation)).split()
            # Limpiar Cada Palabra y Agregarla al Conjunto de Palabras.
            all_words.update(clean_word(word.lower()) for word in words)

        # Calcular el Tiempo de Procesamiento del Archivo y Escribirlo en el archivo
"a9_matricula.txt".
        processing_time = time.time() - file_timer_start
        with open(processing_time_file_path, "a") as processing_time_file:
            processing_time_file.write(f"{file_name}: {processing_time:.6f} segundos\n")

# Escribir Todas las Palabras Limpias en Orden Alfabético en el Archivo "diccionario.txt".
with open(dictionary_file_path, "w") as dictionary_file:
    for word in sorted(all_words):
        dictionary_file.write(f"{word}\n")


# Tiempo Total de Ejecución del Programa.
total_program_time = time.time() - total_timer_start



# Función para ELIMINAR Palabras Específicas del Archivo "diccionario.txt"
def remove_words_from_dictionary(words_to_remove):
    # Leer el Archivo "diccionario.txt"
    with open(dictionary_file_path, "r") as dictionary_file:
        dictionary_words = set(dictionary_file.read().splitlines())

    # Eliminar Palabras Especificadas.
    remaining_words = dictionary_words.difference(words_to_remove)

    # Guardar las Palabras Eliminadas en el Archivo "palabras_eliminadas.txt".
    removed_words = dictionary_words.difference(remaining_words)
    with open(deleted_words_file_path, "w") as deleted_words_file:
        for word in removed_words:
            deleted_words_file.write(f"{word}\n")

    # Guardar las Palabras Restantes en Archivo "diccionario.txt".
    with open(dictionary_file_path, "w") as dictionary_file:
        for word in sorted(remaining_words):
            dictionary_file.write(f"{word}\n")



# Función para ELIMINAR las palabras especificas del rchivo "diccionario.txt".
words_to_remove = {"a", "about", "above", "according", "across", "actually", "adj", "after",
"afterwards", "again",
                   "against", "all", "almost", "alone", "along", "already", "also", "although",
"always", "among",
                   "amongst", "an", "and", "another", "any", "anybody", "anyhow", "anyone",
"anything", "anywhere",
                   "are", "area", "areas", "aren't", "around", "as", "ask", "asked", "asking",
"asks", "at", "away",
                   "b", "back", "backed", "backing", "backs", "be", "became", "because", "become",
"becomes",
```

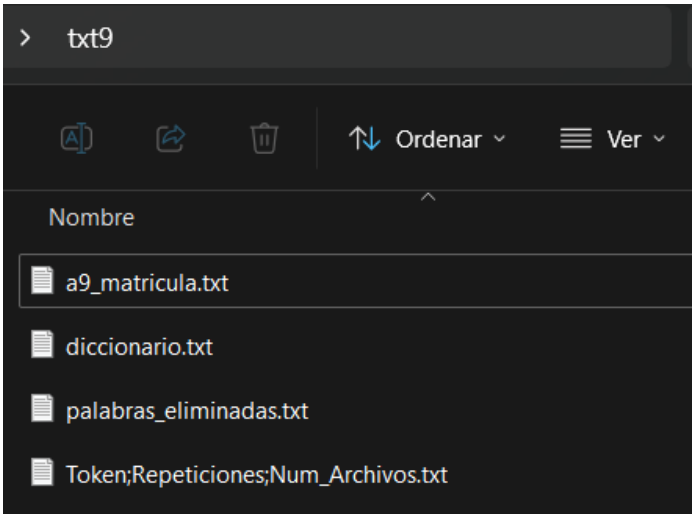
```
"becoming", "been", "before", "beforehand", "began", "begin", "beginning",
"behind", "being",
                    "beings", "below", "beside", "besides", "best", "better", "between", "beyond",
"big", "billion",
                    "both", "but", "by", "c", "came", "can", "can't", "cannot", "caption", "case",
"cases", "certain",
                    "certainly", "clear", "clearly", "co", "come", "could", "couldn't", "d", "did",
"didn't", "differ",
                    "different", "differently", "do", "does", "doesn't", "don't", "done", "down",
"downed", "downing",
                    "downs", "during", "e", "each", "early", "eg", "eight", "eighty", "either",
"else", "elsewhere", "end",
                    "ended", "ending", "ends", "enough", "etc", "even", "evenly", "ever", "every",
"everybody", "everyone",
                    "everything", "everywhere", "except", "f", "face", "faces","fact", "facts",
"far", "felt", "few", "fifty",
                    "find", "finds", "first", "five", "for", "former", "formerly", "forty", "found",
"four", "from", "further",
                    "furthered", "furthering", "furthers", "g", "gave", "general", "generally",
"get", "gets", "give","given",
                    "gives", "go", "going", "good", "goods", "got", "great", "greater", "greatest",
"group", "grouped",
                    "grouping", "groups", "h", "had", "has", "hasn't ", "have", "haven't", "having",
"he", "he'd", "he'll",
                    "he's", "hence", "her", "here","here's", "hereafter", "hereby", "herein",
"hereupon", "hers", "herself",
                    "high", "higher", "highest", "him", "himself", "his", "how", "however",
"hundred", "i", "i'd", "i'll", "i'm",
                    "i've", "ie", "if", "important", "in", "inc", "indeed", "instead","", "interest",
"interested", "interesting",
                    "interests", "into", "is", "isn't", "it", "it's", "its", "itself", "j", "just",
"kl", "large", "largely", "last",
                    "later", "latest", "latter", "latterly", "least", "less", "let", "let's", "lets",
"like", "likely", "long",
                    "longer", "longest", "ltd", "m", "made", "make", "makes", "making", "man",
"many", "may", "maybe", "me","meantime",
                    "meanwhile", "member", "members", "men", "might", "million", "miss", "more",
"moreover", "most", "mostly", "mr",
                    "mrs", "much", "must", "my", "myself", "n", "namely", "necessary", "need",
"needed", "needing", "needs", "neither",
                    "never", "nevertheless","new", "newer", "newest", "next", "nine", "ninety", "no",
"nobody", "non", "none",
                    "nonetheless", "noone", "nor", "not", "nothing", "now", "nowhere", "number",
"numbers", "o", "of", "off", "often",
                    "old", "older", "oldest", "on", "once","one", "one's", "only", "onto", "open",
"opened", "opens", "or", "order",
                    "ordered", "ordering", "orders", "other", "others", "otherwise", "our", "ours",
"ourselves", "out", "over", "overall",
                    "own", "p", "part", "parted", "parting", "parts", "per", "perhaps", "place",
"places", "point", "pointed", "pointing",
                    "points", "possible", "present", "presented", "presenting", "presents",
"problem", "problems", "put","puts", "q",
                    "quite", "r", "rather", "really", "recent", "recently", "right", "room", "rooms",
"s", "said", "same", "saw", "say",
                    "says", "second", "seconds", "see", "seem", "seemed", "seeming", "seems",
"seven", "seventy", "several", "she","she'd",
                    "she'll", "she's", "should", "shouldn't", "show", "showed", "showing", "shows",
"sides", "since", "six", "sixty",
                    "small", "smaller", "smallest", "so", "some", "somebody", "somehow", "someone",
"something", "sometime", "sometimes",
                    "somewhere", "state", "states", "still","stop", "such", "sure", "t", "take",
"taken", "taking", "ten", "than", "that",
                    "that'll", "that's", "that've", "the", "their", "them", "themselves", "then",
"thence", "there", "there'd", "there'll",
                    "there're", "there's", "there've", "thereafter", "thereby", "therefore",
"therein", "thereupon", "these", "they",
```

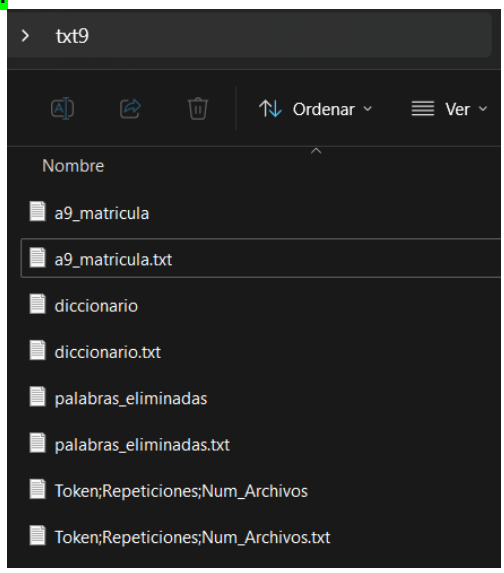
```
                    "they'd", "they'll", "they're", "they've", "thing", "things", "think", "thinks",
"thirty", "this", "those","though",
                    "thought", "thoughts", "thousand", "three", "through", "throughout", "thru",
"thus", "to", "today", "together", "too",
                    "took", "toward", "towards", "trillion", "turn", "turned", "turning", "turns",
"twenty", "two", "u", "under", "unless",
                    "unlike", "unlikely","until", "up", "upon", "us", "use", "used", "uses", "using",
"v", "very", "via", "w", "want",
                    "wanted", "wanting", "wants", "was", "wasn't", "way", "ways", "we", "we'd",
"we'll", "we're", "we've", "well", "wells",
                    "were","weren't", "what", "what'll", "what's", "what've", "whatever", "when",
"whence", "whenever", "where", "where's",
                    "whereafter", "whereas", "whereby", "wherein", "whereupon", "wherever",
"whether", "which", "while", "whither", "who",
                    "who'd", "who'll", "who's", "whoever", "whole", "whom", "whomever", "whose",
"why", "will", "with", "within", "without",
                    "won't", "work", "worked", "working", "works", "would", "wouldn't", "xy","year",
"years", "yes", "yet", "you", "you'd",
                    "you'll", "you're", "you've", "young", "younger", "youngest", "your", "yours",
"yourself", "yourselves", "z"}
remove_words_from_dictionary(words_to_remove)



# Tiempo Total de Ejecución del Programa en el archivo "a9_matricula.txt"
with open(processing_time_file_path, "a") as processing_time_file:
    # Tiempo Total de Ejecución del Programa.
    total_program_time = time.time() - total_timer_start
    processing_time_file.write(f"\nTiempo Total de Ejecucion del Programa: {total_program_time:.6f}
segundos\n")


print("\n\n ------------------------------------------------------------------------------------------
---")
print(" |  Proceso Completado... Revisa la carpeta 'txt9' para observar los resultados del tiempo
|")
print(" ------------------------------------------------------------------------------------------
")
```

Compilación del Programa - Impresión de Consola.

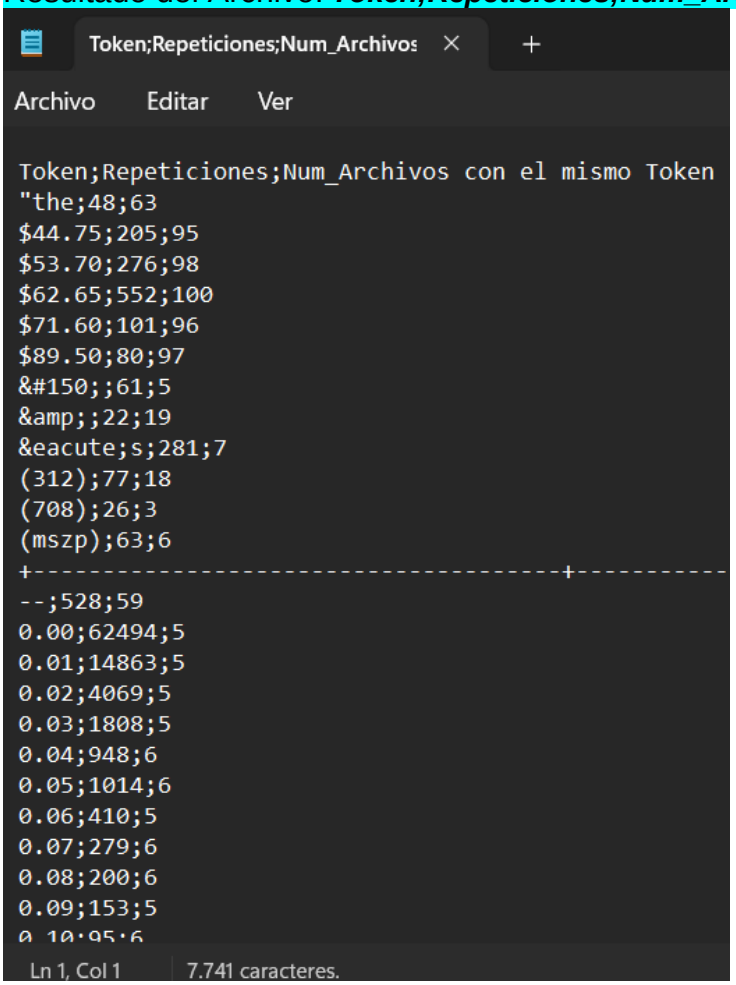
```
C:\Users\100100050\AppData\
This version of python seems to be incorrectly compiled
(internal generated filenames are not absolute).
This may make the debugger miss breakpoints.
Related bug: http://bugs.python.org/issue1666807


 ------------------------------------------------------------------------------------------
 |  Proceso Completado... Revisa la carpeta 'txt9' para observar los resultados del tiempo |
 ------------------------------------------------------------------------------------------
Press any key to continue . . .
```

Resultado de la Carpeta "**txt9**"- *(antes y despues)*:





Resultado del Archivo: ***Token;Repeticiones;Num_Archivos.txt*** :



Resultado del Archivo: ***diccionario.txt*** :

diccionario

Archivo Editar Ver

```
aacuteldozattal
aacuteldozunk
aacuteldva
aacutelellenzeacutekiek
aacutell
aacutellaacutespontja
aacutellaacutespontjukat
aacutellaacutespontot
aacutellam
aacutellamalapiacutetaacutesa
aacutellamhataacuterait
aacutellami
aacutellamisaacuteg
aacutellamnyelv
aacutellamok
aacutellamokboacutel
aacutellamotismeri
aacutellampaacuterti
aacutellampolgaacuterainak
aacutellampolgaacuteri
aacutellampolgaacuterok
aacutellamrend
aacutellamrendet
aacutellamtitkaacuter
aacutellamtitkaacutera
aacutellamtitkaacuterokat
```

Ln 1, Col 1 1.240.486 caracteres.

Resultado del Archivo: ***palabras_eliminadas.txt*** :

palabras_eliminadas

Archivo Editar Ver

```
downs
v
enough
nonetheless
her
done
if
formerly
showing
problems
saw
necessary
f
an
above
mrs
much
pointed
meantime
me
there
say
alone
fact
d
way
interested
```

Ln 106, Col 8 2.985 caracteres.

Resultado del Archivo: ***a9_matricula.txt*** :

a9_matricula

Archivo Editar Ver

```
483.html: 0.007021 segundos
484.html: 0.017979 segundos
485.html: 0.007997 segundos
486.html: 0.012970 segundos
487.html: 0.008024 segundos
488.html: 0.000979 segundos
489.html: 0.003002 segundos
490.html: 0.002960 segundos
491.html: 0.004021 segundos
492.html: 0.083762 segundos
493.html: 0.003000 segundos
494.html: 0.010077 segundos
495.html: 0.004999 segundos
496.html: 0.001992 segundos
497.html: 0.003994 segundos
498.html: 0.089018 segundos
499.html: 0.012000 segundos
500.html: 0.019995 segundos
501.html: 0.023988 segundos
502.html: 0.017017 segundos
503.html: 0.012984 segundos
hard.html: 0.000999 segundos
medium.html: 0.000989 segundos
simple.html: 0.002019 segundos

Tiempo Total de Ejecucion del Programa: 21.155014 segundos
```

Explicación del Funcionamiento del Programa:

El programa se divide en 2 partes.

Parte 1 - Tokens:

En esta parte, el programa usa como base el código de la actividad 6 *(el cual procesa los archivos HTML en la carpeta de salida, contabiliza los tokens y sus repeticiones, y genera un archivo de registro con los resultados y el tiempo total de ejecución).* Y se implementaron nuevas funsiones para que se cuente y elimine una determinada frecuencia de aparición de los tokens (palabras) en los archivos HTML. Ademas, para crear un diccionario con las palabaras contenidas en los archivos HTML y eliminar una serie de palabras.

Paso 1 - Lectura de Archivos HTML:
El programa itera sobre todos los archivos en la carpeta de salida y lee el contenido de los archivos HTML.

Paso 2 - Conteo de Tokens:
Despues de que se ha leído el contenido de un archivo, el programa cuenta la frecuencia de cada palabra (token) en ese archivo, utilizando un diccionario `*word_counts*`.

Paso 3 - Actualización de Diccionarios:
El recuento de palabras se agregan al diccionario principal `*token_counts*`, que almacena la frecuencia total de cada token en todos los archivos. Además, se actualiza el diccionario `*file_counts*`, que cuenta cuántos archivos contienen ese token, elimina los tokens con frecuencias igual o menor a 20 y elimina tokens de una letra o un digito.

Paso 4 - Escritura en Archivo de Log:
Después de procesar todos los archivos, el programa escribe los resultados en el archivo de log *"Token;Repeticiones;Num_Archivos.txt"*, que incluye el token, su frecuencia total y el número de archivos que contienen ese token.

Parte 2 - Diccionario de Palabras

En esta parte del programa, se crea un diccionario de palabras limpias a partir de los archivos HTML en la carpeta de salida. Aquí está el proceso detallado:

Paso 1 - Limpieza de Palabras:
Primero se eliminan los números y caracteres especiales (excepto la comilla simple) de cada palabra en los archivos HTML.

Paso 2 - Creación del Diccionario:
Las palabras limpias se agregan a un conjunto `*all_words*`. Y este conjunto se utilizará para crear el diccionario de palabras en el próximo paso.

Paso 3 - Escritura en el Diccionario:
Las palabras limpias se ordenan alfabéticamente y se escriben en un archivo de texto llamado *"diccionario.txt".*

Paso 4 - Eliminación de Palabras Específicas:

⇒ Del artchivo "diccionario.txt", se eliminan las palabras especificadas en el pdf *“Actividad9_stoplist”* proporcionado en las intrucciones de la actividad.

⇒ Las palabras eliminadas se escriben y guardan en el archivo *"palabras_eliminadas.txt".*

Paso 5 - Tiempo de Ejecución:

El tiempo total de ejecución de esta parte del programa se calcula y se escribe en el archivo *"a9_matricula.txt".*

Resultado Final:

El resultado final del programa incluye un archivo de log de tokens que muestra la frecuencia de cada palabra en todos los archivos HTML en *“Token;Repeticiones;Num_Archivos.txt”*, un diccionario de palabras limpias en *"diccionario.txt"*, una lista de palabras eliminadas en *"palabras_eliminadas.txt"* y el tiempo de ejecución en *"a9_matricula.txt"*.