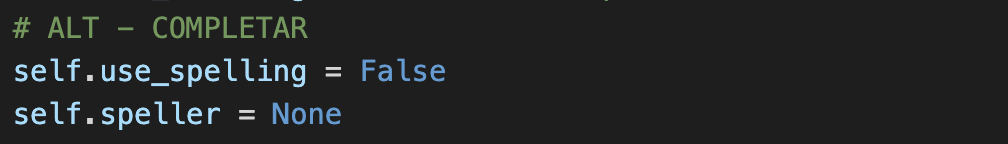
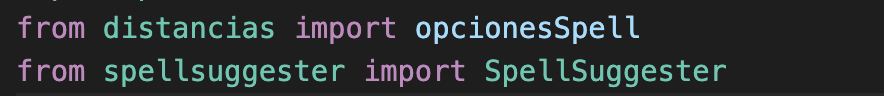
**4. Integración del corrector en el proyecto SAR**

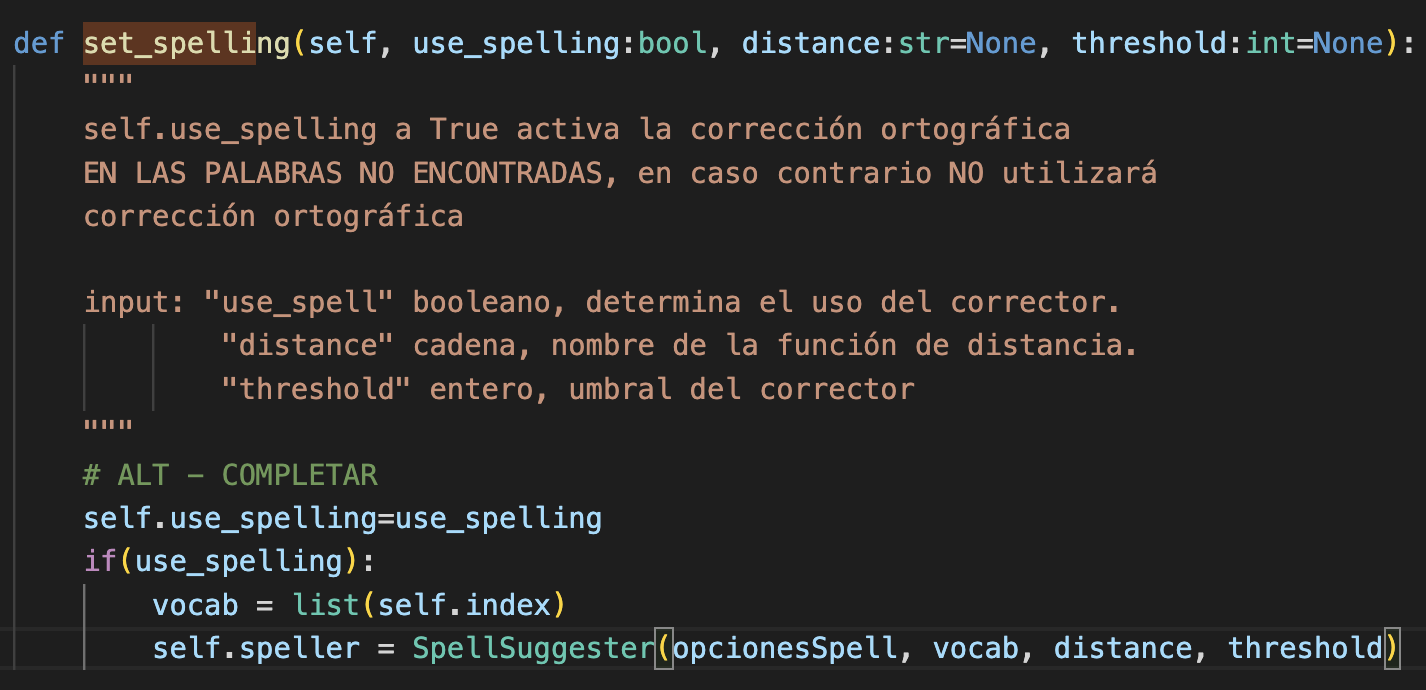
Para implementar la integración, se ha modificado el fichero SAR\_lib.py:

* Añadiendo los atributos ‘use\_spelling’ y ‘speller’ a la clase SAR\_Indexer.



* Se ha añadido un nuevo método ‘set\_spelling’, que comprueba si ‘set\_spelling’ está a True, en caso afirmativo asigna al atributo ‘speller’ el objeto SpellSuggester, pasándole a su constructor el vocabulario (claves del atributo ‘index’), distance y threshold (que se pasan como argumentos de la función) y el diccionario de tipos de distancia, previamente importado





* Se ha modificado también el método ‘get\_posting’ para poder realizar la búsqueda con tolerancia, comprobando, primeramente, si ‘use\_spelling’ es True y r1 está vacía (porque el término no está en el diccionario ‘index’), para poder usar el speller.

Recorriendo la lista de las palabras sugeridas, se busca una a una con el get con una lista auxiliar y seguidamente se hace el or\_posting de la auxiliar (r2) con r1 para así ir obteniendo el posting list juntándolo todo.

Texto

Descripción generada automáticamente

**AMPLIACIONES**

**3. Ampliación Damerau-Levenshtein intermedio (reducción coste espacial)**

* Para implementar la versión intermedia se han añadido dos nuevos casos en la ecuación de recurrencia, que son:

acb <-> ba

ab <-> bca

También ha hecho falta usar 3 vectores para poder tener una visión de los resultados intermedios con una ventana de 3.

Texto

Descripción generada automáticamente

**4. Ampliación Damerau-Levenshtein intermedio (recuperación secuencia de edición)**

* Para la recuperación de la secuencia de edición no se puede hacer uso de la versión con reducción de coste espacial, ya que se necesita toda la matriz de resultados intermedio para ir recorriéndola inversamente. Se ha hecho uso del algoritmo que se usa en ‘damerau\_intermediate\_matriz’ para obtener la matriz resuelta.

Se ha recorrido D comprobando en todo momento cuál de los 6 casos ha sido el que se ha realizado (porque se cumplen las condiciones del caso en concreto y la distancia es la mínima) y decrementando ‘i’ o/y ‘j’ como sea oportuno en cada caso.

Es decir, si se sabe que se ha hecho una transposición ab <-> ba, se ha de retroceder D[i-2][j-2], que será el nuevo punto de partida del bucle de recuperación del camino.

Cada vez que se encuentra una operación de edición se hace el append a la lista ‘l’, indicando los símbolos de ‘x’ e ‘y' involucrados en la operación.

Una vez se sale del bucle, puede pasar que un queden símbolos de ‘x’ o de ‘y’, se hace la comprobación y se hace el append de las operaciones de edición que faltan (solo pueden ser o borrado o inserción).

Finalmente, se hace el ‘l.reverse’ para dar la secuencia ordenada de izquierda a derecha.

Texto

Descripción generada automáticamente