Ejercicio obligatorio semana 2

Juegos de palabras

Introducción

Actualmente existen distintos juegos de palabras, tanto en formato de juego de mesa, como en juegos de video. En muchos de ellos, el juego consiste en encontrar palabras a partir de letras que pueden, o bien estar todas en un tablero, o bien ser asignadas a cada jugador.
Para verificar que la palabra formada por un jugador sea correcta existen distintos métodos, y uno de los mas simples consiste en tener un lista de palabras correctas y el programa debe chequear que la palabra formada esté en esa lista.
Cómo armamos la lista? Existen distintas formas pero, en general, las listas que uno puede obtener no son “ideales” para los requerimientos del juego. Por ejemplo, el idioma español utiliza acentos pero, normalmente, es preferible ignorarlos en los juegos.

Tarea

Has sido contratado por una empresa de video juegos que están trabajando en un juego basado en el Scrabble™ . Ya tienen una lista de palabras y te encargan que los ayudes a limpiar la lista, como para que la puedan utilizar en el juego.

Los problemas principales que tiene la lista que te dan son:
1) Las palabras están acentuadas y desean que se remuevan los acentos
2) Desean mantener algunas de las reglas del juego original con respecto a qué letras se pueden incluir en el alfabeto en español. En particular, te piden que elimines de la lista todas las palabras que contengan letras que, aún cuando las usamos, no forman parte del alfabeto del idioma (ver más información acá). No tenés que preocuparte por letras como la ‘rr’, ‘ch’ o ‘ll’; esas se consideran como dos letras separadas.
3) La lista contiene palabras con mayúsculas y minúsculas, por lo que te piden que entregues una lista con todas las palabras en minúscula.
4) Finalmente, te piden que la lista no contenga palabras duplicadas y que la entregues en orden alfabético.

Especificación

Junto con estas instrucciones hay un archivo de Python llamado Helpers.py . El archivo ya contiene funciones, que son las que deberás completar de acuerdo a los siguientes requerimientos:

1)

```
AMinusculas(palabra) : string
```

La función debe tomar una palabra como argumento y retornar dicha palabra con todos sus caracteres en minúsculas. Si la palabra pasada es 'ElEfanTE', por ejemplo, la función deberá retornar 'elefante'.

2)

```
ConvertirAcentos(palabra, letras) : string
```

La función debe tomar como argumento una palabra y una estructura de datos a su elección que le sirva para determinar si la palabra contiene una letra acentuada y reemplazarla por la variante no acentuada si es necesario. Si la palabra pasada es 'Función', por ejemplo, deber retornar 'Funcion'.

3)

```
EliminarDuplicados(palabras) : list
```

La función debe tomar como argumento una estructura de datos que contenga todas las palabras y debe retornar una lista (list) que contenga todas las palabras pero sin repeticiones. Si la estructura que se pasa como argumento contiene las palabras: 'hola', 'gato', 'perro', 'gato', 'hola', la función debe retornar ['hola', 'gato', 'perro']. El orden puede variar, pero es importante que el formato sea una lista, que no falten palabras y que no haya palabras repetidas

4)

```
Ordenar(palabras) : list
```

La función debe tomar como argumento una estructura de datos que contenga todas las palabras y debe retornar una lista (list) que contenga todas las palabras en orden alfabético ascendente. Si la estructura que se pasa como argumento contiene las palabras: 'hola', 'gato', 'perro', 'elefante', la función debe retornar ['elefante', 'gato', 'hola', 'perro']

5)

```
RemoverPalabrasInvalidas(palabras, letras) : list
```

La función debe tomar como argumento una estructura de datos que contenga todas las palabras y otra estructura de datos que contenga las letras que les ayuden a solamente seleccionar las que corresponden al español. Pueden decidir pasar las letras que pertenecen a nuestro alfabeto, o las que no pertenecen. Pueden contar con que no va a

haber en la lista letras “extrañas”, como ‘č’, ‘ž’, ‘ç’,’ã’, etc., así como tampoco caracteres que pertenezcan a otros sistemas alfabéticos (cirílico, kanji, etc.).
Si la estructura de datos con palabras contiene: ‘hola’, ‘washington’, ‘perro’, la función debe retornar [‘hola’, ‘perro’]

6)

```
ProcesarLista(palabras, acentos, letras) : list
```

Esta función es la que procesa la lista original y debe retornar la lista que cumpla todos los requisitos. Toma como argumentos adicionales las estructuras que ustedes elijan para facilitar el reemplazo de acentos y la remoción de palabras con letras no válidas.
Tienen libertad de llamar las 5 funciones anteriores en el orden que mejor les parezca, mientras que el resultado final sea el correcto.
Si bien el programa podría dar el resultado correcto sin necesidad de llamar a las 5 funciones anteriores, es requisito que todas las funciones provean resultados correctos.

7)

```
main()
```

El código incluye la presencia de main(), y una lista de palabras ya definida como para que puedan probar el funcionamiento de su código. Tienen total libertad de organizar el código como quieran dentro de main().
Podrán observar que la llamada a main() se encuentra dentro de la guarda

```
if __name__ == "__main__":
```

Recuerden que esto se pone para evitar que el código de main() corra si uno importa el archivo como módulo.
Es requerimiento de la tarea que NO alteren esa línea de código. Tampoco pueden poner código que se ejecute fuera de las funciones que ya existen. Si desean crear sus propias funciones extras lo pueden hacer, pero recuerden llamarlas desde dentro de alguna de las funciones arriba mencionadas.
Si desean importar alguna librería de Python, lo pueden hacer. Sin embargo, deberían poder resolver el ejercicio sin necesidad de hacerlo

Formato

Una vez que estén satisfechos con el resultado de su programa, deben subirlo a Moodle luego de cambiar el nombre de acuerdo al formato siguiente:

NombreApellido_Helpers.py

Es decir, si tu nombre es Juan Pérez, el archivo que subas se tiene que llamar: JuanPérez_Helpers.py

Recursos y tips

Página de Wikipedia donde pueden leer las letras aceptadas en Scrabble en español: Wiki

Documentación de Python donde pueden encontrar recursos con respecto a manejar distintas estructuras de datos: Docs (arriba a la izquierda pueden cambiar el idioma, esta sí está en castellano).

Dado que les doy libertad para pasar argumentos del tipo que les resulte mas fácil, pero deben retornar sólo string o list, dependiendo del caso, les recuerdo que Python es muy flexible con respecto a generar distintos tipos de estructuras.
Por ejemplo, si tienen una lista llamada miLista y la quieren convertir en set, simplemente pueden hacer:

```
set(miLista)
```

Lo mismo, si tienen un set llamado miSet y lo quieren convertir en lista, simplemente hacen:

```
list(miSet)
```