Verificación de Diccionario Alienígena

Descripción del Proyecto

En este ejercicio abordamos el problema **"Verificación de Diccionario Alienígena"**, un reto común en entrevistas de empresas tecnológicas como Facebook, Microsoft y Google. El objetivo es determinar si una lista de palabras está ordenada según un alfabeto alienígena.

Planteamiento del Problema

Entradas:

- `palabras`: Lista de cadenas, por ejemplo, `["apple", "app"]`.
- `orden`: Cadena de 26 caracteres que define el alfabeto alienígena, por ejemplo, `"hlabcdefgijkmnopqrstuvwxyz"`.

Salida:

- Valor booleano (`True` o `False`):
    - `True` si `palabras` está ordenada lexicográficamente según `orden`.
    - `False` en caso contrario.

Cómo Resolverlo

1. **Construir el Mapa del Alfabeto Alienígena**
   Creamos un diccionario que asigne a cada letra su posición en `orden`:

   ```
   letter_order = {car: idx for idx, car in enumerate(orden)}
   ```

2. **Comparar Dos Palabras**
    - Recorremos carácter a carácter hasta encontrar la primera diferencia.
    - Si la letra de la primera palabra aparece antes en `letter_order`, esa palabra es menor.
    - Si aparecen todas iguales hasta el final de la palabra más corta, consideramos menor a la palabra de menor longitud.

3. **Verificar Toda la Lista**
   Recorremos la lista de palabras comparando cada par consecutivo:

   ```
   for i in range(1, len(palabras)):
       if not comparar(palabras[i-1], palabras[i]):
           return False
   return True
   ```

Ejemplo de Uso

```
if __name__ == "__main__":
    palabras = ["hola", "holaa", "holb"]
    orden = "hlabcdefgijkmnopqrstuvwxyz"
    resultado = is_alien_sorted(palabras, orden)
    print("¿Está ordenado?", resultado)  # → True
```

Análisis de Complejidad

- **Tiempo:** O(n · L)
    - n = número de palabras
    - L = longitud máxima de las palabras
- **Espacio extra:** O(1) (sin contar el almacenamiento de entrada)

Por Qué Importa Este Problema

- Evalúa la comprensión de órdenes lexicográficas y comparaciones personalizadas.
- Mide la habilidad para diseñar soluciones eficientes con estructuras de datos básicas.
- Es un ejercicio habitual en entrevistas para valorar tu capacidad de razonamiento y organización de código.

Consejos Prácticos

- Divide el problema en pasos claros y abórdalos uno a uno.
- Añade comentarios en el código para explicar la lógica.
- Prueba con distintos casos, incluyendo casos límite y prefijos.
- Planifica tu enfoque antes de escribir código para evitar errores.

¡Éxito en tu práctica! 🚀