Algoritmos y Estructuras de Datos II
Examen parcial de Programación Dinámica

Profesor: Jesús Sánchez Cuadrado

25 de abril de 2024

Nombre y apellidos: **Subgrupo**:

Una empresa de análisis de redes sociales tiene el encargo de analizar mensajes (posts, tweets, etc.) y extraer los términos más frecuentemente usados. El problema que se encuentran es que los *hashtags* no están separados por espacios pero necesitan recuperar las palabras que contiene el hashtag. Sin embargo, existe una dificultad adicional y es que es frecuente que para acortar los hashtags se escriban abreviaturas. Por ejemplo, en el hashtag #byeexmnlgoritmos se puede observar que *exmn* corresponde a la palabra *examen* del diccionario, pero con algunos cambios y *lgoritmos* sería *algoritmos*. La empresa dispone de un diccionario de abreviaturas junto con la distancia mínima de edición a la palabra correspondiente del diccionario. De esta forma, se puede recuperar la segmentación *bye, examen, algoritmos*.

Tu tarea es escribir un algoritmo que, dado un hashtag y la función que calcula la distancias, determine si el texto del hashtag se puede segmentar (minimizando el número de total cambios de las palabras seleccionadas) y en ese caso qué palabras forman esa segmentación. La función que calcula las distancias devuelve infinito (∞) si la palabra no está en el diccionario. Se usará el siguiente ejemplo concreto:

```
# Dada una palabra devuelve la palabra más cercana del diccionario
funcion palabra_cercana(palabra) : string
  bye       -> bye
  algo      -> algo
  lgo       -> algo
  ritmos    -> ritmos
  exmn      -> examen
  lgoritmos -> algoritmos
  -> en cualquier otro caso devuelve nulo

# Dada una palabra, devuelve la distancia a la palabra más cercana del diccionario
funcion distancia(palabra) : int
  bye       -> 0
  algo      -> 0
  lgo       -> 1
  ritmos    -> 0
  exmn      -> 2
  lgoritmos -> 1
  -> en cualquier otro caso devuelve infinito

Hashtag: byeexmnlgoritmos
  Distancia total: 3
  Palabras: bye examen algoritmos
```

Diseña un algoritmo de programación dinámica que resuelva este problema, siguiendo los siguientes apartados:

1. (2.5 puntos) Escribe la ecuación de recurrencia.

2. (2.5 puntos) Diseña la tabla de programación dinámica y establece el orden para rellenarla. Completa la tabla para el ejemplo mostrado arriba: segmentar el hashtag *byeexmnlgoritmos*.

3. (2.5 puntos) Implementa el algoritmo de programación dinámica. Utiliza pseudocódigo, C++ o una mezcla de ambos.

4. (2.5 puntos) Implementa el algoritmo para recuperar la solución