**Algoritmo de distancia de edición**

1. **Inicio**

**-**Se crea una matriz llamada DP donde su tamaño es

Donde*m* es la longitud X (m=X) y donde *n* es la longitud Y(n=Y)

-DP representa la distancia mínima para transformar las primeras *i* en caracteres de X en los primeros j en caracteres de Y

1. **Llenado de tabla**

**-**se recorre la tabla de arriba abajo y de izquierda a derecha o dicho mas fácilmente recorre primero columnas y después filas

-Si los caracteres son iguales : dp[i][j] = dp[i-1][j-1] no necesita cambiar nada o mostrar nada ya que no se necesita la operación o proceso

-Si los caracteres no coinciden:  
-dp[i][j] = 1 + min(

dp[i-1][j] → Eliminación

dp[i][j-1] → Inserción

dp[i-1][j-1] → Sustitución  
así usando la solución mas simple y sencilla

**3. verificación de letras agregadas**

Cuando ya tenemos la tabla dp lista, empezamos desde la esquina final dp[m][n] (abajo a la derecha) y vamos retrocediendo hasta llegar al inicio:

-Si las letras son iguales → vamos en diagonal hacia arriba a la izquierda (dp[i-1][j-1]). (No hay cambio, así que seguimos con la letra anterior de cada palabra).

1. Si venimos desde la izquierda (dp[i][j-1] + 1) → eso significa que se agregó una letra de la segunda palabra (Y).  
   (La guardamos en la lista agregadas y retrocedemos una columna.)
2. Si venimos desde arriba (dp[i-1][j] + 1) → significa que se eliminó una letra de la primera palabra (X).  
   (No la guardamos porque no es una inserción, solo subimos una fila.)
3. Si venimos en diagonal con +1 (dp[i-1][j-1] + 1) → significa que se cambió una letra por otra.  
   (Avanzamos en diagonal, pero no agregamos nada a la lista.)
4. Cuando terminamos, invertimos la lista agregada para que las letras aparezcan en el orden en que se agregan al transformar la primera palabra en la segunda.