Απόσταση Levenshtein

Έκδοση 1.0 (28/2/23) ΚΣ

Απόσταση Μεταξύ Λέξεων

Πώς ορίζεται;

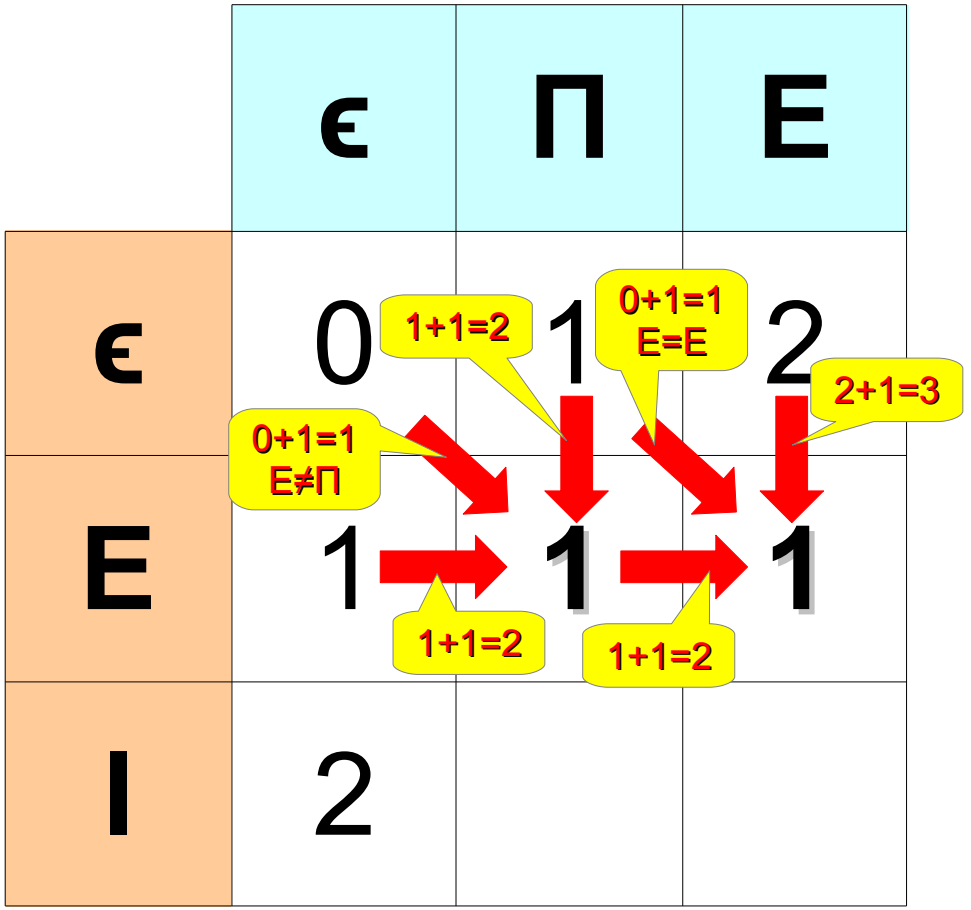
Από το πλήθος των διορθώσεων που χρειάζεται να γίνουν για να μετατρέψουμε τη μία λέξη στην άλλη.

(edit distances)

Απόσταση Levenshtein

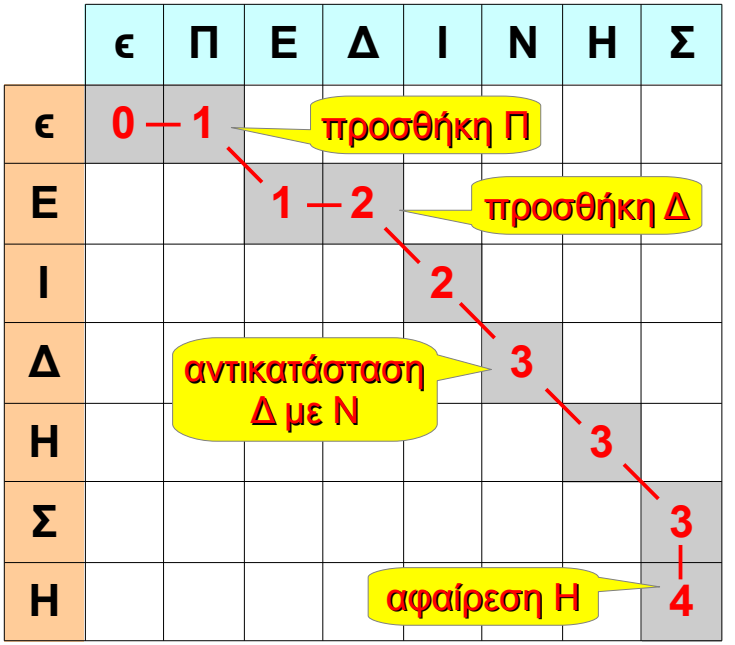
		ϵ	Π	Ε	Δ	Ι	Ν	Η	Σ
	ϵ	0	1	2	3	4	5	6	7
	Ε	1							
	Ι	2							
	Δ	3							
	Η	4							
	Σ	5							
	Η	6							

		ϵ	Π	Ε	Δ	Ι	Ν	Η	Σ
	ϵ	0	1	2	3	4	5	6	7
	Ε	1	1	1	2	3	4	5	6
	Ι	2	2	2	2	2	3	4	5
	Δ	3	3	3	2	3	3	4	5
	Η	4	4	4	3	3	4	3	4
	Σ	5	5	5	4	4	4	4	3
	Η	6	6	6	5	5	5	4	4



Βέλτιστο Μονοπάτι

	ϵ	Π	Ε	Δ	Ι	Ν	Η	Σ
ϵ	0	1	2	3	4	5	6	7
Ε	1	1	1	2	3	4	5	6
Ι	2	2	2	2	2	3	4	5
Δ	3	3	3	2	3	3	4	5
Η	4	4	4	3	3	4	3	4
Σ	5	5	5	4	4	4	4	3
Η	6	6	6	5	5	5	4	4

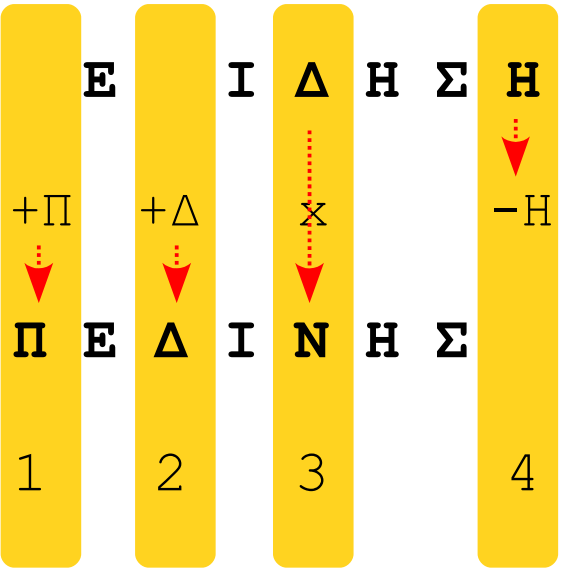


Ξεκινά πάντα από το πάνω αριστερό τετράγωνο δημιουργώντας μια αύξουσα ακολουθία τιμών μέχρι το κάτω δεξί τετράγωνο.

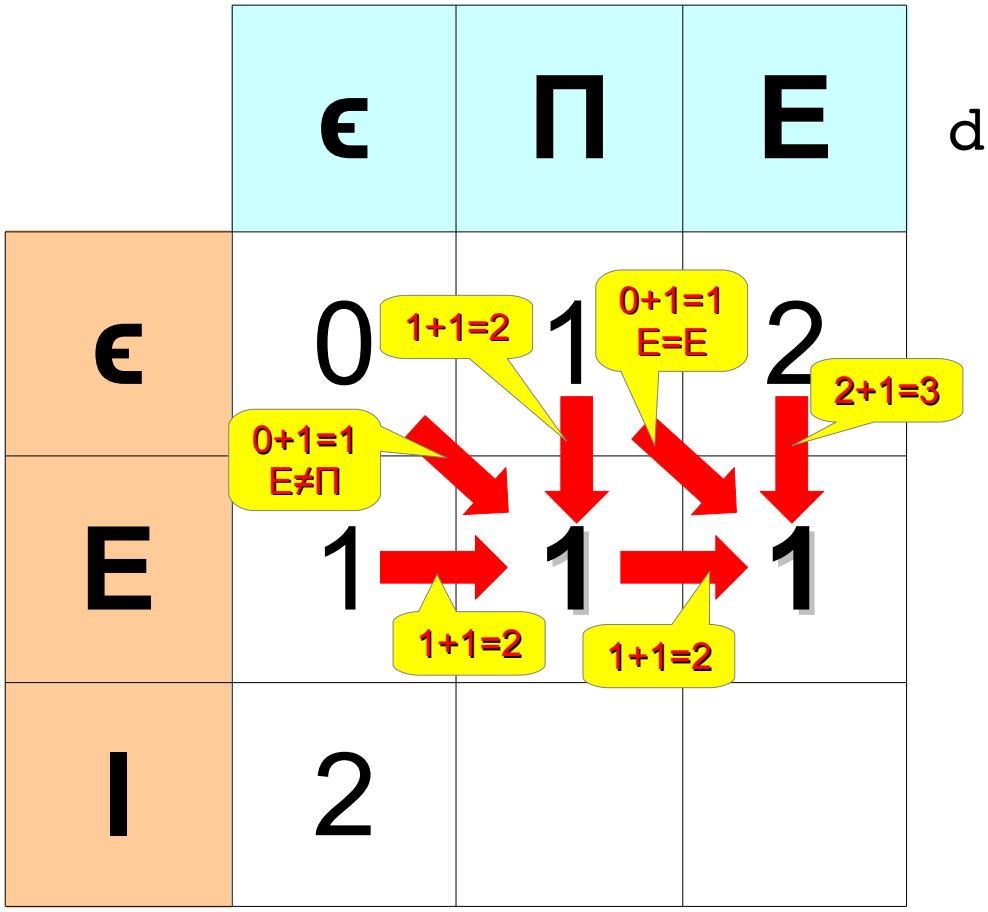
Για κάθε οριζόντια ή κατακόρυφη μετακίνηση η αύξηση γίνεται κατά 1.
Για κάθε διαγώνια μετακίνηση η αύξηση μπορεί να είναι 0 ή 1.

Κάθε φορά που αυξάνεται η τιμή στο μονοπάτι έχουμε και μια αλλαγή:

> οριζόντια μετακίνηση = προσθήκη του χαρακτήρα της στήλης
> κατακόρυφη μετακίνηση = αφαίρεση του χαρακτήρα γραμμής
> διαγώνια μετακίνηση = αντικατάσταση του συμβόλου της γραμμής με το σύμβολο της στήλης



Υλοποίηση




```
def Lev(S,T):

    if S=="": return len(T)
    if T=="": return len(S)

    p1=Lev(S,T[:-1])+1
    p2=Lev(S[:-1],T)+1

    p3=Lev(S[:-1],T[:-1])
    if S[-1]!=T[-1]: p3+=1

    return min(p1,p2,p3)
```