EDIT DISTANCE

THE EDIT DISTANCE IS A WAY TO QUANTIFY HOW DISSIMILAR TWO STRINGS ARE, BY COUNTING THE MINIMUM NUMBER OF OPERATIONS REQUIRED TO TRANSFORM ONE STRING INTO THE OTHER.

THE POSSIBLE OPERATIONS ARE: INSERTION, DELETION, REPLACEMENT AND TRANSPOSITION.
(OPTIONAL ↑)

THEY ARE TYPICALLY DONE AT CHAR-LEVEL

EXAMPLES

ED (CAT , ACT) = 2 USING 2 REPLACES

NOTE = ACTUALLY WE CAN ACHIEVE 1, IF WE ARE ABLE TO TRANSPOSE

ED (CAT , DOG) = 3 USING 3 REPLACES

GENERALLY IMPLEMENTED WITH **DYNAMIC PROGRAMMING**

```
ED (S1, S2) =
    i = 0        Rows
    j = 0        columns
    m [i][j] = 0

    for i = 1, ..., |S1| - 1 :
        m [i][0] = i

    for j = 1, ..., |S2| - 1 :
        m [0][j] = j


    for i = 1, ..., |S1| - 1 :
        for j = 1, ..., |S2| - 1 :
            if S1[i] == S2[j] :
                m[i][j] = m[i-1][j-1]
            else:
                m[i][j] = 1 + min { m[i, j-1],
                                    m[i-1, j],
                                    m[i-1, j-1] }

    return m[|S1|-1][|S2|-1]
```

(i-1,j-1)	(i-1,j)
(i,j-1)	(i,j)

THE COST IS $O(|S1| \cdot |S2|)$, MAKING THIS APPROACH QUITE EXPENSIVE FOR OUR NEEDS.

OUR PROBLEM

IN ORDER TO DO ISOLATED WORD CORRECTION, WE ARE GIVEN A LEXICON (DIZIONARIO) AND A CHAR SEQUENCE Q, AND WE WOULD LIKE TO RETURN TO THE USER THE WORDS IN THE LEXICON CLOSEST TO Q

E.D. APPROACH

SO, GIVEN A QUERY Q, WE WOULD WANT TO ENUMERATE ALL CHARS SEQUENCE WITHIN A PRESET EDIT DISTANCE AND INTERSECT THIS SET WITH THE LIST OF "CORRECT" WORDS THAT WE OWN

BRUTE FORCE!!
THE LEXICON IS USUALLY HUGE

A POSSIBLE WAY TO REDUCE THE NUMBER OF COMPUTATIONS IS THE WEIGHTED EDIT DISTANCE.

↓

LIKE NORMAL EDIT DISTANCE, BUT THE OPERATIONS ARE WEIGHTED DEPENDING ON THE CHARS INVOLVED

EXAMPLE

IT IS MORE COMMON TO MISS-TYPE A m TO A n THEN TO A q, THEREFORE IT IS A PREFERRED OPERATIONS

However

- Now it is required a weigthed matrix as input (MORE SPACE?)
- We have to modify the classic Dynamic Programming algo To handle weigths (Todious!)