	Bansilal Ramnath Agarwal Charitable Trust's Vishwakarma Institute of Information Technology **Department of Artificial Intelligence and Data Science**

Name: **Siddhesh Dilip Khairnar**		
Class: **TY**	Division: **B**	Roll No: **372028**
Semester: **6th**	Academic Year: **2023-24**	
Subject Name & Code: **Natural Language and Processing & ADUA32203**		
Title of Assignment: **Calculate minimum edit distance between two strings**		
Date of Performance: **27-01-2024**	Date of Submission: **08-02-2024**	

ASSIGNMENT NO: - 3

Aim: Calculate minimum edit distance between two strings

❖ **THEORY:**

- **Introduction of available word Similarity Measures:**

1. **Euclidean Distance**: It measures the straight-line distance between two points in Euclidean space. In the context of word similarity, it calculates the distance between word vectors in a high-dimensional space.

2. **Cosine Similarity**: This measure calculates the cosine of the angle between two vectors representing the word embeddings of the words. It is widely used in document retrieval and clustering tasks.

3. **Jaccard Similarity**: It calculates the similarity between two sets by measuring the intersection divided by the union of the sets. In the context of words, it considers the presence or absence of words in documents or contexts.

4. **Levenshtein Distance (Edit Distance):** This measures the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into another. It's useful for measuring the similarity between two strings or words.

5. **Fast Text Subword Embeddings:** FastText embeddings incorporate subword information, making them useful for handling out-of-vocabulary words and capturing morphological similarities between words.

6. **Word Movers Distance:** This measure calculates the minimum cumulative distance required to transport all the word embeddings of one text to another, effectively measuring the similarity between two texts based on their word embeddings.

7. **Embedding-based Measures:** With the advent of word embeddings (e.g., Word2Vec, GloVe, FastText), similarity measures often rely on vector representations of words. Cosine similarity, Euclidean distance, and others can be applied to these vector representations to measure word similarity.

```python
import nltk

# Download NLTK word list (if not already downloaded)
nltk.download('words')

from nltk.corpus import words

def min_edit_distance(source, target):
    m = len(source)
    n = len(target)

    # Initialize a matrix to store the edit distances
    dp = [[0] * (n + 1) for _ in range(m + 1)]

    # Initialize the first row and column
    for i in range(m + 1):
        dp[i][0] = i
    for j in range(n + 1):
        dp[0][j] = j

    # Fill in the matrix using dynamic programming
    for i in range(1, m + 1):
        for j in range(1, n + 1):
            cost = 0 if source[i - 1] == target[j - 1] else 1
            dp[i][j] = min(dp[i - 1][j] + 1,      # Deletion
                           dp[i][j - 1] + 1,      # Insertion
                           dp[i - 1][j - 1] + cost)  # Substitution

    return dp[m][n]

def spelling_checker(word, dictionary):
    # Find the closest match in the dictionary
    min_distance = float('inf')
    closest_match = None

    for candidate in dictionary:
        distance = min_edit_distance(word, candidate)
        if distance < min_distance:
            min_distance = distance
            closest_match = candidate

    return closest_match, min_distance

# Take user input
word_to_check = input("Enter a word: ")

# Use NLTK words list as the dictionary
dictionary = words.words()

closest_word, min_distance = spelling_checker(word_to_check, dictionary)
print(f"Suggested correction for '{word_to_check}': {closest_word}")
print(f"Minimum edit distance: {min_distance}")
```

OUTPUT:

```
[nltk_data] Downloading package words to
[nltk_data]     C:\Users\Dell\AppData\Roaming\nltk_data...
[nltk_data]   Package words is already up-to-date!
Enter a word: helllo
Suggested correction for 'helllo': hello
Minimum edit distance: 1

Enter a word: summmeer
Suggested correction for 'summmeer': summer
Minimum edit distance: 2

Enter a word: eeneergy
Suggested correction for 'eeneergy': energy
Minimum edit distance: 2

Enter a word: Happinneess
Suggested correction for 'Happinneess': happiness
Minimum edit distance: 3

Enter a word: intelligence
Suggested correction for 'intelligence': intelligence
Minimum edit distance: 0
```