# Lists, tuples and dictionary:

# finding the maximum, minimum, mean;

# linear search on list/tuple of numbers, and counting the

# frequency of elements in a list using a dictionary.

# Introduce the notion of accessing elements in a collection

# using numbers and names.

# Strings: compare, concat, substring;

# notion of states and transitions using state transition

def findMaxMin():

    list1 = [62,45,37,56,48]

    tuple1 = (23,65,76,4,64)

    dict1 = {'age1':44, 'age2':14, 'age3':31, 'age4':54}

    print(f"\nOriginal list: {list1}, Maximum: {max(list1)}, and Minimum: {min(list1)}")

    print(f"Original tuple: {tuple1}, Maximum: {max(tuple1)}, and Minimum: {min(tuple1)}")

    print(f"Original dictionary: {dict1}, Maximum: {max(dict1.values())}, and Minimum: {min(dict1.values())}")

    print()

def linearSearch(search):

    list2 = [62,45,37,56,48]

    tuple2 = (23,65,76,4,64)

    for i in list2:

        if search == i:

            print(f'\n{i} is present in list2')

            break

    else:

        print(f'\n{search} not found in list2')

    for k in range(len(tuple2)):

        if search == k:

            print(f'\n{k} is present in tuple2')

            break

    else:

        print(f'\n{search} not found in tuple2')

def frequencyElement(list3):

    freq = {}

    print(f'Original: {list3}')

    for items in list3:

        freq[items] = list3.count(items)

    for k in freq.items():

        print(k)

def stringMan(str1, str2):

    print('\nOriginal Strings...')

    print(f'str1: {str1}')

    print(f'str2: {str2}')

    print('\nCacatenating two strings...')

    print(f'{str1} and {str2} as: {str1} {str2}')

    print('\nComparing two strings lexigographically...')

    if str1 > str2:

        print(f'{str1} is lexigraphically greater than {str2}')

    else:

        print(f'{str2} is lexigraphically greater than {str1}' )

if \_\_name\_\_ == "\_\_main\_\_":

    # findMaxMin()

    # linearSearch(3)

    # frequencyElement(['a', 's', 'a', 's', 'c', 'c', 'c','b'])

    stringMan('Hello', 'Ezra')

OUTPUTS







