PIYUSH KUMAR MISHRA

230957212

WEEK2:

1. Write a Python program to find the longest increasing subsequence from a given list of numbers.

```
def longest_increasing_subsequence(arr):

    n = len(arr)

    lis = [1] * n  # Initialize LIS values for all indexes as 1


    # Compute optimized LIS values in a bottom-up manner

    for i in range(1, n):

        for j in range(0, i):

            if arr[i] > arr[j] and lis[i] < lis[j] + 1:

                lis[i] = lis[j] + 1


    # Find the maximum value in lis[]

    maximum = max(lis)


    # Reconstruct the longest increasing subsequence

    lis_sequence = []

    current_length = maximum

    for i in range(n - 1, -1, -1):

        if lis[i] == current_length:

            lis_sequence.append(arr[i])
            current_length -= 1

    lis_sequence.reverse()  # The sequence is constructed in reverse order

    return lis_sequence

# Example usage
arr = [10, 22, 9, 33, 21, 50, 41, 60]
print("Longest Increasing Subsequence is:", longest_increasing_subsequence(arr))
```

OUTPUT:

```
Longest Increasing Subsequence is: [10, 22, 33, 41, 60]
```

2. Create a Python script to generate a list that contains 25 elements and display the frequency of each item in a list

```
# To generate a list and find the frequency of each item in the list

def CountFrequency(my_list):

    freq = {}
    for item in my_list:
        if (item in freq):
            freq[item] +=1
        else:
            freq[item] = 1

    for key,value in freq.items():
        print("%d : %d " % (key,value))
```

```
n = int(input("Enter the no of items in the list: \n"))

lis = []

i = 0

print("Enter the items of the list:")

while i < n:

    lis_item = lis.append(int(input()))

    i +=1


CountFrequency(lis)
```

OUTPUT:

```
Enter the no of items in the list:

25

Enter the items of the list:

1

2

3

4

5

6

7

8

9

1

2
```

```
3
4
5
6
7
8
9
1
2
3
4
5
6
7
1 : 3
2 : 3
3 : 3
4 : 3
5 : 3
6 : 3
7 : 3
8 : 2
9 : 2
```

<u>**3. Develop a Python program that constructs a list of 15 strings. It should then determine the count of strings in this list that have a minimum length of two characters and also start and end with identical characters. You can choose any specific list of strings for this task.**</u>

```
def count_strings_with_identical_ends(strings):
    count = 0

    for s in strings:

        if len(s) >= 2 and s[0].lower() == s[-1].lower():  # Added .lower() to handle case sensitivity

            count += 1

    return count


strings = [

    'Ava',

    'Ben',

    'Civic',

    'David',

    'Eve',

    'Felicity',

    'Gog',

    'Hannah',

    'I',

    'Jill',

    'Kayak',

    'Liam',

    'Madam',

    'Nina',

    'Otto'

]


count = count_strings_with_identical_ends(strings)

print("Count of strings with min length of 2 and identical start and end characters:", count)
```

OUTPUT:

```
Count of strings with min length of 2 and identical start and end characters: 9
```

4. Develop a Python script that generates a list with 15 items and then eliminates any duplicates from that list.

```
# To create a random list of 15 elements and delete the duplicate elements from that list


import random


def generate_random_list(size, lower_bound, upper_bound):
    return [random.randint(lower_bound, upper_bound) for _ in range(size)]


def remove_duplicates(input_list):
    return list(set(input_list))



list_size = 15
lower_bound = 1
upper_bound = 100



random_list = generate_random_list(list_size, lower_bound, upper_bound)
print("Original list with possible duplicates:")
print(random_list)



unique_list = remove_duplicates(random_list)
print("\nList after removing duplicates:")
print(unique_list)
```

OUTPUT:

```
Original list with possible duplicates:

[48, 70, 24, 52, 67, 34, 68, 30, 24, 19, 83, 39, 54, 91, 86]


List after removing duplicates:

[34, 67, 68, 70, 39, 48, 19, 52, 83, 54, 86, 24, 91, 30]
```

5. Develop a Python script that builds a list with 15 elements. This script will reposition the items in the list by doing a circular right shift. The number of positions shifted will be based on a user-specified value.

```
# To  reposition the items in the list by doing a circular right shift. The number of positions shifted will be based on a user-specified value.


import random




def circular_right_shift(lst, positions):

   n = len(lst)

   positions = positions % n

   return lst[-positions:] + lst[:-positions]


def main():


   lis = []

   n =  15

   i = 0

   while i < n :
        lis.append(int(random.randint(0, 100)))

        i += 1

    print("Original list:", lis)



    while True:

        try:

            positions = int(input("Enter the number of positions to shift (integer): "))

            if positions < 0:

                print("Please enter a non-negative integer.")

            else:

                break

        except ValueError:

            print("Invalid input. Please enter a valid integer.")



    shifted_list = circular_right_shift(lis, positions)



    print("List after shifting:", shifted_list)


if __name__ == "__main__":

    main()
```

OUTPUT:

```
Original list: [42, 6, 5, 91, 11, 49, 64, 65, 22, 60, 2, 99, 4, 3, 89]

Enter the number of positions to shift (integer): 2

List after shifting: [3, 89, 42, 6, 5, 91, 11, 49, 64, 65, 22, 60, 2, 99, 4]
```