Q1. Write a python prog. for Row-wise element addition in tuple matrix.

```python
# Define the tuple matrix
tuple_matrix = [
    (1, 2, 3),
    (4, 5, 6),
    (7, 8, 9) ]

# Initialize an empty list to store the row sums
row_sums = []

# Iterate through each row in the tuple matrix
for row in tuple_matrix:
    # Calculate the sum of the current row and append it to the row_sums list
    row_sums.append(sum(row))

# Print the row-wise sums
print("Row-wise sums:", row_sums)
```

Output:

Row-wise sums: [6,15,24]

Q2. Write python prog. to count the number of character (character frequency) in a string “google.com” .

```python
# Define the string
string = "google.com"

# Initialize an empty dictionary to store the frequency count
freq_count = {}

# Iterate through each character in the string
for char in string:
    # If the character is already in the dictionary, increment its count
    if char in freq_count:
        freq_count[char] += 1
    # Otherwise, add the character to the dictionary with a count of 1
    else:
        freq_count[char] = 1

# Print the frequency count of each character
print("Character frequencies:", freq_count)
```

Output: Character frequencies: {'g': 2, 'o': 3, 'l': 1, 'e': 1, '.': 1, 'c': 1, 'm': 1}

Q3. Input a string “Information Technology” and write python code to do the following-

a) Find the letter 'n' form the given string.

```
# Input string
input_string = "Information technology"

letter_n = 'n'
found_n = letter_n in input_string
print(f"Letter '{letter_n}' found in the string: {found_n}")
```

Output: Letter 'n' found in the string: True

b) Split the string at every occurrence of the letter 'o'.

```
split_string = input_string.split('o')
print(f"String split at every 'o': {split_string}")
```

Output: String split at every 'o': ['Inf', 'rmati', 'n techn', 'l', 'gy']

c) Check if the string is palindrome or not.

```
is_palindrome = input_string == input_string[::-1]
print(f"Is the string a palindrome? {is_palindrome}")
```

Output: Is the string a palindrome? False

d) Count the occurrence to letter 'n' in the string.

```
count_n = input_string.count('n')
print(f"Occurrences of letter 'n': {count_n}")
```

Output: Occurrences of letter 'n': 3

e) Form a new string by appending the string 'Platform' at the end of given string.

```
new_string = input_string + " Platform"
print(f"New string: {new_string}")
```

Output: New string: Information technology Platform

Q4. Write a python prog. to check the first and last char. are same from a given list of strings.

```
strings = ["level", "radar", "hello", "world", "python", "refer"]
print("The word have first and last char is Same-",end="")
for word in strings:
    if word[0]==word[-1]:
        print(word, end=", ")
```

Output: The word has first and last char is Same-level, radar, refer,

Q5. Write a python prog. to count the even and odd number in a list using lambda function.

```
# List of numbers
numbers = [1, 5, 14, 23, 56, 78, 91]

# Count the even numbers using lambda
even_count = len(list(filter(lambda x: x % 2 == 0, numbers)))

# Count the odd numbers using lambda
odd_count = len(list(filter(lambda x: x % 2 != 0, numbers)))

# Print the counts
print(f"Even numbers count: {even_count}")
print(f"Odd numbers count: {odd_count}")
```

Output: Even numbers count: 3
Odd numbers count: 4

Q6. Write a python prog. to concatenate two 3-D NumPy array on axis 1.

```python
import numpy as np

# Creating two sample 3-D arrays
array1 = np.array([[[1, 2, 3], [4, 5, 6]],
                   [[7, 8, 9], [10, 11, 12]]])

array2 = np.array([[[13, 14, 15], [16, 17, 18]],
                   [[19, 20, 21], [22, 23, 24]]])

# Concatenating the two arrays along axis 1
result = np.concatenate((array1, array2), axis=1)

# Printing the resulting array
print("Array 1:\n", array1)
print("\nArray 2:\n", array2)
print("\nConcatenated Array:\n", result)
```

Output: Concatenated Array:

```
[[[ 1  2  3]
  [ 4  5  6]
  [13 14 15]
  [16 17 18]]

 [[ 7  8  9]
  [10 11 12]
  [19 20 21]
  [22 23 24]]]
```

Q7. Write a python prog. to find key with max. unique value in a dictionary.

```python
test_dict = {
    "A": [4, 5, 6, 3],
    "B": [7, 5, 9, 0],
    "C": [1, 5, 9, 2]
}

max_unique_count = 0
key_with_max_unique = None

for key, values in test_dict.items():
    unique_values = len(set(values))
    if unique_values > max_unique_count:
        max_unique_count = unique_values
        key_with_max_unique = key

print(f"The key with the maximum unique values is: {key_with_max_unique}")
print(f"Maximum unique values count: {max_unique_count}")
```

Output: The key with the maximum unique values is: A
Maximum unique values count: 4

Q8. Write a python prog. to calculate sum of all even numbers (first 100 even numbers in a list).

```python
# Generate a list of the first 100 even numbers
even_numbers = [i for i in range(2, 202, 2)]

# Calculate the sum of all even numbers in the list
sum_even_numbers = sum(even_numbers)

print(f"The sum of the first 100 even numbers is: {sum_even_numbers}")
```

Output: The sum of the first 100 even numbers is: 10100

Q9. Write a python prog. to calculate sum of all odd numbers (first 100 odd numbers in a list).

```
# Initialize the list to store odd numbers
odd_numbers = []

# Generate the first 100 odd numbers
for i in range(1, 201, 2):  # Start at 1, end at 201 (exclusive), step by 2 to get odd numbers
    odd_numbers.append(i)

# Calculate the sum of the list of odd numbers
sum_of_odd_numbers = sum(odd_numbers)

# Print the sum
print("Sum of the first 100 odd numbers:", sum_of_odd_numbers)
```

Output: Sum of the first 100 odd numbers: 10000

Q10. Write a python prog. to find area of circle and rectangle, take input from the user.

```
r=int(input())
l=int(input())
b=int(input())

# Area of Circle
area_circ=3.14*r*r
print("The area of circle is = ",area_circ)

#Area of Rectangle
area_rect=l*b
print("The area of rectangle is = ",area_rect)
```

Q11. Write a python prog. to input marks and name of 10 student and find out max. and min. marks among them and display their name with marks.

```
# Predefined dictionary of student names and their corresponding marks
students = {"Alice": 85,"Bob": 92,"Charlie": 78,"David": 88,"Eve": 95,
            "Frank": 77,"Grace": 84,"Hannah": 91,"Isaac": 89,"Judy": 76}

# Extract names and marks from the dictionary
names = list(students.keys())
marks = list(students.values())

# Find the maximum and minimum marks
max_marks = max(marks)
min_marks = min(marks)

# Find the names of students with maximum and minimum marks
max_marks_name = names[marks.index(max_marks)]
min_marks_name = names[marks.index(min_marks)]

# Display the results
print("Student with the highest marks:")
print(f"Name: {max_marks_name}, Marks: {max_marks}")

print("\nStudent with the lowest marks:")
print(f"Name: {min_marks_name}, Marks: {min_marks}")
```

Output:

Student with the highest marks- Name: Eve, Marks: 95

Student with the lowest marks- Name: Judy, Marks: 76

Q12. Write a python prog. to count positive and negative number in a list.

```python
# Predefined list of numbers
numbers = [12, -7, 5, -3, 0, 8, -2, 15, -6, 0]

# Initialize counters for positive and negative numbers
positive_count = 0
negative_count = 0

# Iterate through the list and count positive and negative numbers
for num in numbers:
    if num > 0:
        positive_count += 1
    elif num < 0:
        negative_count += 1

# Display the results
print(f"Number of positive numbers: {positive_count}")
print(f"Number of negative numbers: {negative_count}")
```

Output:

Number of positive numbers: 5

Number of negative numbers: 4

Q13. Find the currency notes

(simple currency notes are: 2000,500,200,100,50,20,10,5) against a given amount.

```python
amount = 2755
# List of currency denominations in descending order
denominations = [2000, 500, 200, 100, 50, 20, 10, 5]

# Dictionary to store the count of each denomination
note_count = {}

# Iterate through each denomination
for denomination in denominations:
    # Calculate the number of notes for the current denomination
    count = amount // denomination
    # Assign the count to the corresponding denomination in the dictionary
    note_count[denomination] = count
    # Reduce the amount by the total value of notes for the current denomination
    amount %= denomination

# Print the results
for denom, count in note_count.items():
    if count > 0:
        print(f"{denom} : {count}")
```

Output:

2000 : 1

500 : 1

200 : 1

50 : 1

5 : 1