**MACHINE LEARNING (22AIE213)**

**ASSIGMENT-1**

**NAME: HARSHAVARDHAN REDDY**

**REG NO: AIE22169**

Q1: Consider the given list as [2,7,4,1,3,6].Write a program to count pairs of elements with sum equal to 10.2

**Pseudo Code:**

function count\_pairs\_with\_sum(lst, target\_sum):

pairs\_count = 0

seen\_numbers = set()

for num in lst:

complement = target\_sum - num

if complement in seen\_numbers:

pairs\_count += 1

seen\_numbers.add(num)

return pairs\_count

list\_a = [2, 7, 4, 1, 3, 6]

target\_sum\_A = 10

pairs\_count\_A = count\_pairs\_with\_sum(list\_a, target\_sum\_A)

print(f"Count of pairs with sum {target\_sum\_A} in the list: {pairs\_count\_A}")

**Explanation:**

function called count\_pairs\_with\_sum that takes a list (lst) and a target sum (target\_sum) as input. The function calculates and returns the count of pairs of numbers in the list that add up to the target sum.

The function uses a set (seen\_numbers) to keep track of numbers encountered while iterating through the list. For each number in the list, it calculates the complement (the difference between the target sum and the current number). If the complement is found in the set of seen numbers, it means a pair with the desired sum has been found, and the pairs\_count is incremented. The function then adds the current number to the set of seen numbers.

Q2: Write a program that takes a list of real numbers as input and returns the range(difference between minimum and maximum) of the list. Check for list being less than 3 elements in which case return an error message(Ex: “Range determination not possible”).Given a list[5,3,8,1,0,4],the range is 8(8-0).

**Pseudo Code:**

function calculate\_range(real\_numbers):

if length of real\_numbers < 3:

raise ValueError("Range determination not possible for lists with less than 3 elements.")

return maximum value in real\_numbers - minimum value in real\_numbers

real\_numbers\_a = [5, 3, 8, 1, 0, 4]

range\_a = calculate\_range(real\_numbers\_a)

print(f"Range of the list: {range\_a}")

**Explanation:**

The code defines a function called calculate\_range that takes a list of real numbers as input. It calculates the range of the numbers in the list, which is the difference between the maximum and minimum values.

Before performing the calculation, the function checks if the length of the list is less than 3. If it is, it raises a ValueError since determining the range is not possible with fewer than 3 elements.

Question 3: Write a program that accepts a square matrix A and a positive integer m as arguments and returns Am.

**Pseudo Code:**

function matrix\_power(matrix, power):

result = matrix

for \_ in range(power - 1):

result = [[sum(a \* b for a, b in zip(row, col)) for col in zip(\*matrix)] for row in result]

return result

function input\_matrix():

rows = input("Enter the number of rows: ")

cols = input("Enter the number of columns: ")

matrix = []

for i in range(rows):

row = []

for j in range(cols):

element = float(input(f"Enter element at position ({i + 1}, {j + 1}): "))

row.append(element)

matrix.append(row)

return matrix

function input\_power():

return int(input("Enter the power to which the matrix should be raised: "))

matrix\_a = input\_matrix()

power\_a = input\_power()

print(f"The resultant matrix after raising it to the power of {power\_a} is:\n{matrix\_power(matrix\_a, power\_a)}")

**Explanation:**

The code consists of three functions:

matrix\_power(matrix, power): This function takes a matrix and a power as input and calculates the result of raising the matrix to the given power using matrix multiplication. It uses list comprehensions to perform the multiplication.

input\_matrix(): This function prompts the user to input the number of rows and columns for a matrix, and then takes user input for each element of the matrix.

input\_power(): This function prompts the user to input the power to which the matrix should be raised.

Question 4: Write a program to count the highest occurring character & its occurrence count in an input string. Consider only alphabets. Ex: for “hippopotamus” as input string, the maximally occurring character is ‘p’& occurrence count is 3.

**Pseudo Code:**

function highest\_occurrence\_char(input\_string):

char\_count = {}

for char in input\_string:

if char.isalpha():

char\_count[char] = char\_count.get(char, 0) + 1

if not char\_count:

return None, 0

max\_char = character with maximum count in char\_count

max\_count = count of max\_char in char\_count

return max\_char, max\_count

input\_string = "hippopotamus"

max\_char\_A, max\_count\_A = highest\_occurrence\_char(input\_string)

print(f"Max occurring character: '{max\_char\_A}', Occurrence count: {max\_count\_A}")

**Explanation:**

The code defines a function highest\_occurrence\_char that takes a string as input and calculates the character with the highest occurrence count in the alphabetic characters of the string. It uses a dictionary (char\_count) to keep track of the counts.

The function iterates through the characters in the input string, updating the counts in the dictionary for each alphabetic character. If there are no alphabetic characters in the input, it returns None and count 0.

After counting, the function finds the character with the maximum count and returns both the character and its count.