AMRITA
VISHWA VIDYAPEETHAM

Machine Learning

22AIE213

Team – 07

Assignment - 01

Submitted by :

BG.Shresta - AIE22106

Lakshmi Sindhu – AIE22126

D Nimraja – AIE22173

BG.Shresta – AIE22106

Q1. Write a program to count the number of vowels and consonants present in an input string.

Psuedo Code :

```
function Counting_vowels_and_consonants(string):
  vowels = ['a', 'e', 'i', 'o', 'u']
  vowels_count = 0
  consonants_count = 0
  for each character in string.lower():
    if character is an alphabetical character:
      if character is in vowels:
        increment vowels_count
      else:
        increment consonants_count
  return vowels_count, consonants_count

function main():
  input_string = get user input("Enter a string")
  vowels, consonants = Counting_vowels_and_consonants(input_string)
  display "Number of vowels:", vowels
  display "Number of consonants:", consonants

if __name__ is "__main__":
  call main()
```

Code Explanation :

This code counts the number of vowels and consonants in a given user input string. It defines a function 'Counting_vowels_and_consonants' which iterates through each character in the input, identifying alphabetical characters and distinguishing between vowels and consonants. The main function prompts the user for input, calls the counting function, and then displays the counts of vowels and consonants. The script efficiently employs character analysis and user interaction, demonstrating a simple and effective approach to processing and categorizing characters in a string.

Q2. Write a program that accepts two matrices A and Bas input and returns their product AB. Check if A & B are multipliable; if not, return error message.

Psuedo Code :

```
function MatrixMultiplication(A, B):
  product = [[0 for _ in range(len(B[0]))] for _ in range(len(A))]

  // Performing matrix multiplication
  for i from 0 to len(A) - 1:
    for j from 0 to len(B[0]) - 1:
      for k from 0 to len(B) - 1:
        product[i][j] += A[i][k] * B[k][j]
  return product

function main():
  rows_of_A = get user input("Enter the number of rows in matrix A: ")
  cols_of_A = get user input("Enter the number of columns in matrix A: ")
  rows_of_B = get user input("Enter the number of rows in matrix B: ")
  cols_of_B = get user input("Enter the number of columns in matrix B: ")

  if cols_of_A != rows_of_B:
    print("Error: Matrices are not multipliable.")
    return

  A = []
  print("Enter the elements of matrix A:")
  for i from 0 to rows_of_A - 1:
    row = []
    for j from 0 to cols_of_A - 1:
      element = get user input()
      row.append(element)
    A.append(row)
```

```
  B = []
  print("Enter the elements of matrix B:")
  for i from 0 to rows_of_B - 1:
    row = []
    for j from 0 to cols_of_B - 1:
      element = get user input()
      row.append(element)
    B.append(row)

  // Performing matrix multiplication
  product = MatrixMultiplication(A, B)

  print("Product of matrices A and B:")
  for row in product:
    print(row)

if __name__ is "__main__":
  call main()
```

Code Explanation :

This code performs matrix multiplication by defining a function 'MatrixMultiplication' which efficiently multiplies two matrices. The main function asks user input for the dimensions and elements of two matrices, checks their compatibility for multiplication, and then computes their product using the matrix multiplication function. The script showcases a structured approach to matrix operations, ensuring user-friendly interaction and delivering the product matrix as the final result. Overall, it demonstrates an effective methodology for handling matrix multiplication in a clear and concise manner.

Q3. Write a program to find the number of common elements between two lists. The lists contain integers.

Psuedo Code :

```
function main():
  display "Enter the elements of the first list (space-separated): "
  input_string_1 = get user input()
```

```
  list_1 = split input_string_1 by spaces
  list_1 = [convert each element to integer for each element in list_1]
  display "Enter the elements of the second list (space-separated): "
  input_string_2 = get user input()

  list_2 = split input_string_2 by spaces
  list_2 = [convert each element to integer for each element in list_2]

  // FindING the common elements between the two lists
  common_elements = set(list_1) intersect set(list_2)
  print "Number of common elements:", length of common_elements

if __name__ is "__main__":
  call main()
```

Code Explanation :

This code takes user input for two lists of integers, converts the input strings into integer lists, and then identifies and counts the common elements between the two lists using set operations. The sets of unique elements from each list are intersected to find the common elements. Then the code outputs the number of common elements found.

Q4. Write a program that accepts a matrix as input and returns its transpose

Psuedo Code :

```
function Transpose_of_matrix(matrix):
  rows = length of matrix
  cols = length of matrix[0]
  transposed_matrix = create matrix of dimensions [cols][rows]

  for i from 0 to rows - 1:
    for j from 0 to cols - 1:
      transposed_matrix[j][i] = matrix[i][j]

  return transposed_matrix
```

```
function main():
    display "Enter the number of rows: "
    rows = get user input as integer
    display "Enter the number of columns: "
    cols = get user input as integer

    matrix = create empty matrix of dimensions [rows][cols]

    for i from 0 to rows - 1:
        for j from 0 to cols - 1:
            display "Enter element at position (" + (i + 1) + ", " + (j + 1) + "): "
            element = get user input as integer
            matrix[i][j] = element

    display "Original Matrix:"
    for each row in matrix:
        print row

    transposed_matrix = Transpose_of_matrix(matrix)

    display "Transposed Matrix:"
    for each row in transposed_matrix:
        print row

if __name__ is "__main__":
    call main()
```

Code Explanation :

This code performs matrix transposition, where the user inputs the dimensions of a matrix and its elements. The code first creates the original matrix and displays it. Then, it calculates the transposed matrix by swapping rows and columns and displays the result. The transposition is achieved by iterating through each element of the original matrix and assigning it to the corresponding position in the transposed matrix. The script provides a user-friendly interface for matrix manipulation.

Lakshmi Sindhu – AIE22126

Q1. Write a program to count the number of vowels and consonants present in an input string.

Psuedo Code :

Function Vowels_and_Consonants(input_string):

Vowels = "AEIOUaeiou" // Initialize vowels to upper and lower case

vowel_count = 0 // Initial vowel count

consonant_count = 0 // Initial count of consonants

For each character in the input string: // For loop to check each character in the input string

If character is an alphabet: // Check if the given input contains alphabets

If char is in Vowels: // If the character is one of the initialized vowels

(Vowels_count += 1)

Increase the number of vowels by 1 // Increase the number of vowels by 1

Other:

Increase the number of consonants by 1 // Increase the number of consonants by 1

(Consonant_count += 1)

Print "Number of Vowels in the string are: ", vowel_count // Display the number of vowels

Print "The number of consonants in the string is: ", consonant_count // Display the number of consonants

// Initial function call

user_input = Input("Enter string: ")

Vowels_and_Consonants(user_input) // Fixed function calls

Code Explanation :

This code snippet allows us to recognise the number of vowels and consonants in a string given by the user. A function is defined inorder to perform this task. At first Vowels are Initialized (upper and lower cases). “for” loop is used for iterating throughout the string. If the letter belongs to the initialised vowel set it gets stored in the vowel_count and the value of the vowel count is incremented by the value 1 and similar procedure applies for consonants as well. User input section allows us to give any string as an input. This string is then checked to see if it contains alphabets.

As a result we get to see the numerical count of the no.of vowels and consonants.

Q2. Write a program that accepts two matrices A and Bas input and returns their product AB. Check if A & B are multipliable; if not, return error message.

Psuedo Code :

```
Functionmultiply_matrices(matrix_A,matrix_B):
rows_A,cols_A=size of matrix_A
rows_B,cols_B=size of matrix_B
//Check if matrix is multiplier
If cols_A then not is equal to row_B:< br> Return "Error: matrices are not equal"

// Initialize result matrix with zero
result_matrix = zero matrix of length(rows_A, column_B )
< br> // Use embedding Do matrix multiplication in a loop
For each i from 0 to rows_A:
For each j from 0 to cols_B:
For each k from 0 to cols_A:
result_matrix[ i] [j] +=matrix_A[i][k] *matrix_B[k][j]

Return result_matrix

// User input matrix A
rows_A = Input ("Enter the number of rows of matrix A:")
cols_A = Output("Enter the number of rows of matrix A:")
matrix_A = []< br>
Print("Enter the number of rows of matrix A) matrix A content - wise :")
For me from 0 to rows_A:
row = list of numbers obtained from user input< br> matrix_A.append(row)

// of matrix B User input
>rows_B = Output("Enter the number of rows in matrix B:")
cols_B = input( "Enter the number of columns in matrix B:")
matrix_B = []<br< b="" style="margin: 0px; padding:
0px;"></br<> >< br >Print ( "Enter the element of matrix B row by row:")
For me from 0 to row_B :
row = list of numbers obtained from user input
matrix_B.append(row)< br> < br >result = multiple_matrices(matrix_A,matrix_B)

Try this as a result:
Print result
Another:
Print "\nMatrix A:"
For matrix_A for each row: <br< b="" style="margin: 0px; padding: 0px;"></br<>> print row

print "\nMatrixB:"
for each row of matrix_B:
print row

print " \nProduct AB: "< br> for each row in the result :
Print line
```

Code Explanation :

This code snippet allows us to add two matrices A and B respectively. At first it checks the two entered matrices by the user are multipliable based their lengths. If not, it returns a statement saying “matrices are not equal”. The result matrix is initialized to zero and later appended with the attained values after performing the multiplication using for loop to iterate row by row until the end of the length of both matrices. As a result we get the desired product matrix.

Q3. Write a program to find the number of common elements between two lists. The lists contain integers.

Psuedo Code :

Function Common_Elements_Count(list1, list2):
common_elements = empty list # Create an empty list to store common elements
For each element in list1: # For each element in the first list(“for” loops recursively checks for common elements in list 2 based on list 1)

If element is in list2 and the element is not in common_elements:
Append the element to common_elements (the initialized empty list)
Return common_elements Length # Return number of elements in count
# User_input
list1 = input(“Enter the elements of list 1(separated with comma)
list2 = input(“Enter the elements of list 2(separated with comma)
result = Common_Elements_Count(list1, list2)

< Print "No . Common Elements:", #Print the quantity element of the common element

Code Explanation :

This code snippet is all about finding the common elements among two lists which contain data in the form of int values separated with comma. This data is user input data. A common_element empty list is created which will get appended by common elements among the two lists. “for” loop is used for performing iteration in the lists to find the common values based on list 1. Once all are found the count of common elements is displayed as a final result.

Q4. Write a program that accepts a matrix as input and returns its transpose

Psuedo Code :

# Define the transpose function of the matrix
def Transpose_Matrix(matrix):
# Now initialize the size of the matrix
row = len(matrix)
cols = len(matrix)
# Initialize empty transpose matrix

```
transpose_matrix = [[0 for _ in range(rows)] for _ in range(cols)]
# Using a loop, iterate over the elements in the matrix from
to i in range(rows):
to j in range(columns):
transpose_matrix[j][i] = matrix[ i][j# After matrix iteration, the transposed elements are returned to the
initialized transpose_matrix
# User input of the matrix
rows = int(input("input Number of rows: ") ))
cols = int(input("Enter the number of columns: "))
matrix = []
print ("Enter matrix element row by row:")< br >for i (row) in array:
row = list(map(int, input( ).split()))
matrix.append (row)
result = Transpose_Matrix( matrix )< br>
print("\nOriginal matrix:")
Row in matrix:< br> print(row)
print("\nTranspose matrix:" )
```

Code Explanation :

This code snippet allows us to enter a matrix and attain a transpose of the same matrix. Similarly

The transpose matrix is initialised with zero later append with the values. A simple logic is applied here with exchanges the rows to columns and vice versa ([i][j]->[j][i]) this switch of change is performed by iteration using for loop. As a result we are displayed with the original and transposed matrix.

D Nimraja – AIE22173

Q1. Consider the given list as [2,7,4,1,3,6].Write a program to count pairs of element with sum equal to 10.

Psuedo Code :

```
Function count(Input_list):
pairs = 0
dictionary = {}
For each num in input_list: complement = 10 - num
If complement exists in dictionary:
pairs += dictionary[complement]
Update dictionary: Increment count for num
Return pairs
Function main():
input_list = [2, 7, 4, 1, 3, 6]
pairs = count(input_list)
Print "Number of pairs with sum equal to 10:", pairs
```

If script is executed directly:
Call main()

Code Explanation :

The main objective of the question is to find pairs that give us a specified target sum. In this case we used dictionary to keep track to how many times each number appears in the given list. While going in the list we calculate the complement foe each number.If this complement has been seen before,we know there's a pair that sums up to 10.Atlast it shows how many pairs are there in the string that's show the output 10.

Q2. Write a program that takes a list of real numbers as input and returns the range of the list.Check for list being less than 3 elements in which case return an error message.

Psuedo Code :

function find_range(real_numbers):
if length of real_numbers < 3:
return "Range determination is not possible"
minimum_value = find_minimum(real_numbers) maximum_value = find_maximum(real_numbers)
range_value = maximum_value - minimum_value
return range_value
function find_minimum(numbers):
return minimum value in numbers
function find_maximum(numbers):
return maximum value in numbers
if __name__ == "__main__":
input_list = [5, 3, 8, 1, 0, 4] # given input
minimum = find_minimum(input_list)
maximum = find_maximum(input_list)
range_result = find_range(input_list)
print("Range is from (", maximum, "-", minimum, ")")
print("Difference:", range_result)

Code Explanation :

The main objective of the question is to find range based on maximum and minimum values.'find_range',which calculates the range of the given list of real number. It first checks if the length of the input list is less than three, and if so, returns a message indicating that range determination is not possible. Atlast find the minimum and maximum values within the list using functions (find_minimum and find_maximum)

Q3. Write a program at accepts a square matrix A and a positive integer m as arguments and returns A^m.

Psuedo Code :

function matrix_power(matrix, power):
result = matrix for _ in range(power - 1):
result = multiply_matrices(result, matrix)
return result
function multiply_matrices(matrix_a, matrix_b):
result = []

```
for each row in matrix_a:
new_row = []
for each column in transpose(matrix_b):
dot_product = calculate_dot_product(row, column)
new_row.append(dot_product)
result.append(new_row)
return result
function transpose(matrix):
transposed_matrix = []
for each column in matrix:
transposed_matrix.append(new_row_with_elements_from_column(column))
return transposed_matrix
function calculate_dot_product(vector_a, vector_b):
return sum(product_of_corresponding_elements(vector_a, vector_b))
function product_of_corresponding_elements(vector_a, vector_b):
return [element_a * element_b for element_a, element_b in zip(vector_a, vector_b)]function main():
matrix_rows = get_user_input("Enter the number of rows in the matrix")
matrix_cols = get_user_input("Enter the number of columns in the matrix")
matrix_a = create_matrix(matrix_rows, matrix_cols)
power_a = get_user_input("Enter the power to raise the matrix to")
result_matrix_a = matrix_power(matrix_a, power_a)
print_matrix("Original matrix", matrix_a)
print_matrix(f"Resultant matrix after raising to power {power_a}", result_matrix_a)
function get_user_input(message):
print(message)
return parse_user_input(input())
function create_matrix(rows, cols):
matrix = []
for i in range(rows):
row = get_user_input(f"Enter values for row {i + 1} (separated by space)")
matrix.append(row)
return matrix
function print_matrix(label, matrix):
print(label)
for row in matrix:
print(row)main()
```

Code Explanation :

The main objective of the question is to find the mulitiplcation of the given square matrix. The matrix_power function is designed to raise a given matrix to a specified power.The multiply_matrices function calculates the product of two matrices by iterating through rows and columns, employing a helper function, transpose, to obtain the transpose of a matrix. The calculate_dot_product computes the dot product of two vectors, and product_of_corresponding_elements calculates the element-wise product of vectors. The main function orchestrates user input for matrix dimensions and values, as well as the desired power.

Q4. Write a program to count the highest occurring character & its occurrence count in an input string.

Psuedo Code :

```
function count(input_string):
```

```
if input_string is empty:
return None, 0
convert input_string to uppercase
initialize an empty dictionary char_count
for each character char in input_string:
if char is an alphabetical character:
increment char_count[char] by 1
if char_count is empty:
return None, 0
find the character with the maximum count (max_char) in char_count
set max_count to the count of max_char in char_count
return max_char, max_countfunction main():
set input_string to "HIPPOPOTAMUS"
call count(input_string) and store the results in highest_char and occurrence_count
if highest_char is not None:
print "Highest occurring character is:", highest_char
print "Occurrence count:", occurrence_count
else:
print "No alphabetic characters found in the input string."
if script is executed as the main program:
call main()
```

Code Explanation :

The main objective of the question is to find highest occurring character & its occurrence count in an input string.The 'count' function counts the occurrences of alphabetical characters in the input string. The main function, calls the count function, and prints the highest occurring character and its count in the input string ; or, it says no alphabetic characters were found.