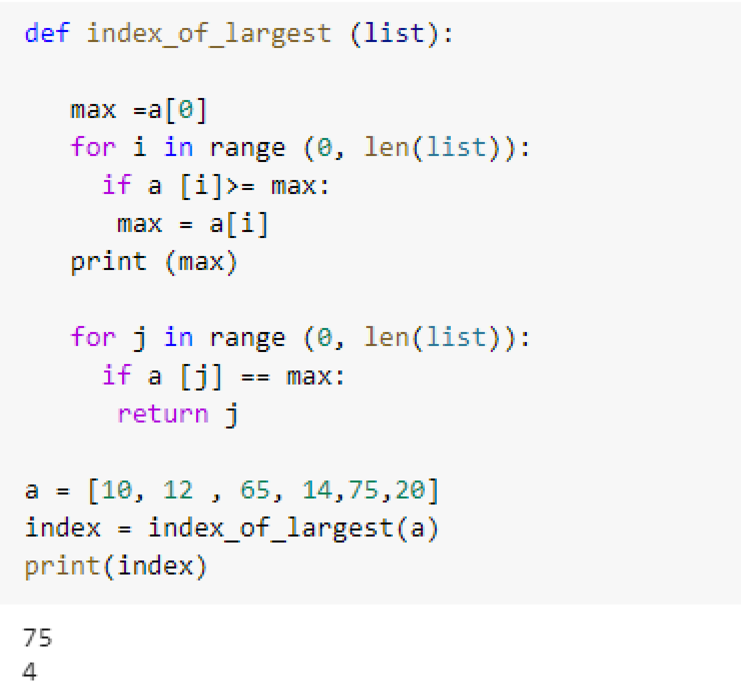
Q1) Create a function that returns the index of the largest element in a list. For example: **index_of_largest([100,24,99,211])** should return **3**.

Note: You are not allowed to use built-in functions or libraries.



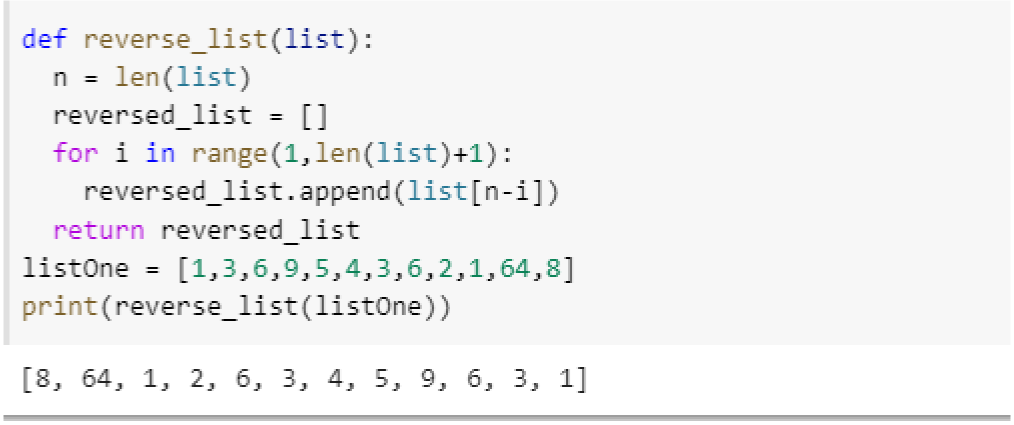
```
def index_of_largest (list):

   max =a[0]
   for i in range (0, len(list)):
     if a [i]>= max:
      max = a[i]
   print (max)

   for j in range (0, len(list)):
     if a [j] == max:
      return j

a = [10, 12 , 65, 14,75,20]
index = index_of_largest(a)
print(index)
```

```
75
4
```

Q2) Create a function that returns the reverse of a given list.

For example: **reverse_list([1,2,5,8,9,4])** should return **[4,9,8,5,2,1]**

Note: You are not allowed to use any built-in function or libraries

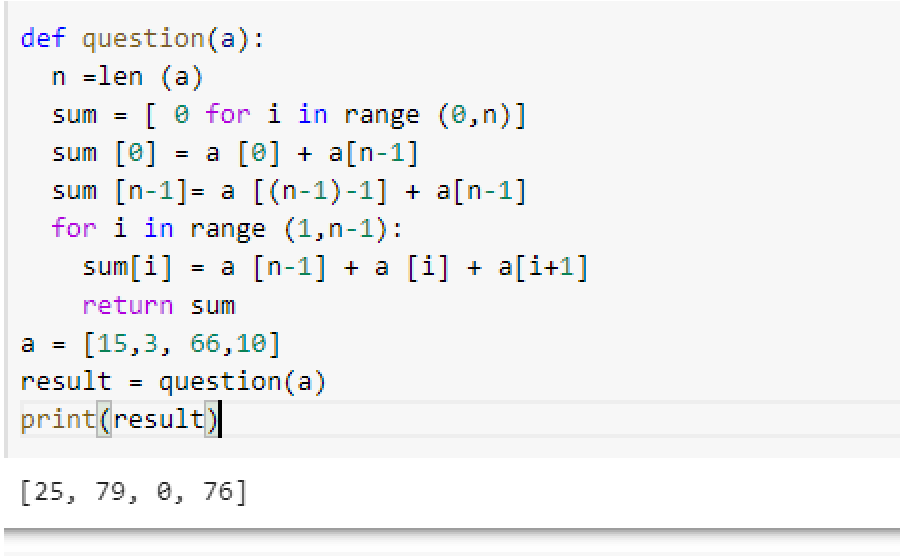


```
def reverse_list(list):
  n = len(list)
  reversed_list = []
  for i in range(1,len(list)+1):
    reversed_list.append(list[n-i])
  return reversed_list
listOne = [1,3,6,9,5,4,3,6,2,1,64,8]
print(reverse_list(listOne))
```

```
[8, 64, 1, 2, 6, 3, 4, 5, 9, 6, 3, 1]
```

Q3) Create a function that takes a list as input and returns a list containing the sum of a number and its adjacent elements.

For example: **sum_of_neighbors([5,10,12,14,24])** should returns**[15,27,36,50,38]**



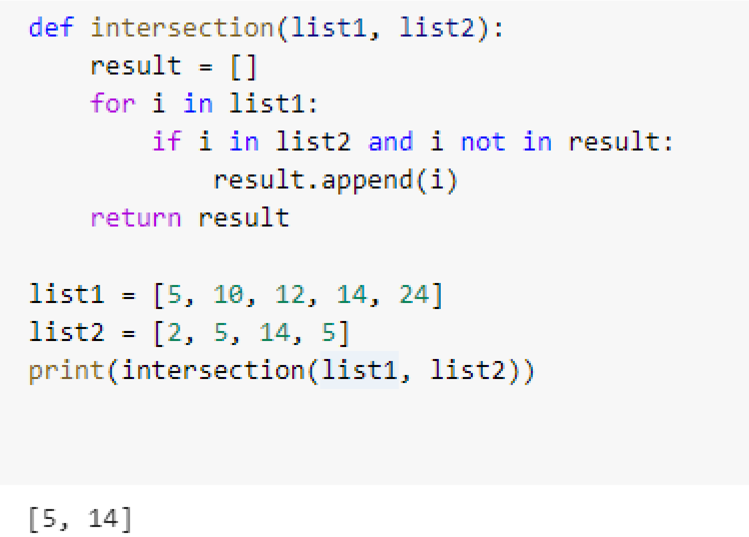
```
def question(a):
  n =len (a)
  sum = [ 0 for i in range (0,n)]
  sum [0] = a [0] + a[n-1]
  sum [n-1]= a [(n-1)-1] + a[n-1]
  for i in range (1,n-1):
    sum[i] = a [n-1] + a [i] + a[i+1]
    return sum
a = [15,3, 66,10]
result = question(a)
print(result)
```

```
[25, 79, 0, 76]
```

Q4) Create a function that takes two lists as argument and returns intersection of the lists.

For example: **intersection([5,10,12,14,24],[2,5,14,5])** should return **[5,14]**

Note: You are not allowed to use any built-in function or libraries



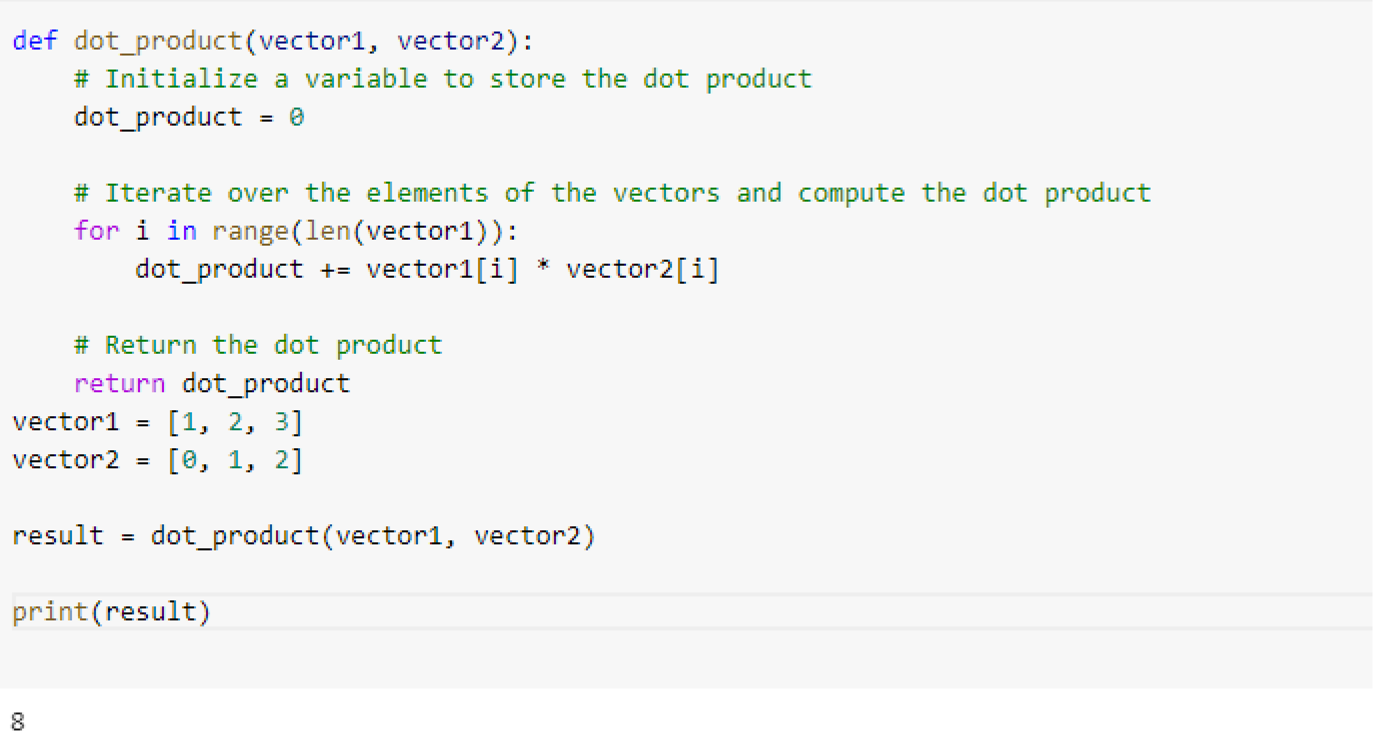
```
def intersection(list1, list2):
    result = []
    for i in list1:
        if i in list2 and i not in result:
            result.append(i)
    return result

list1 = [5, 10, 12, 14, 24]
list2 = [2, 5, 14, 5]
print(intersection(list1, list2))
```

```
[5, 14]
```

Q5) Create a function to calculate the dot product of two vectors of same size.

For example: **dot_product([1,2,3],[0,1,2])** should return **8**

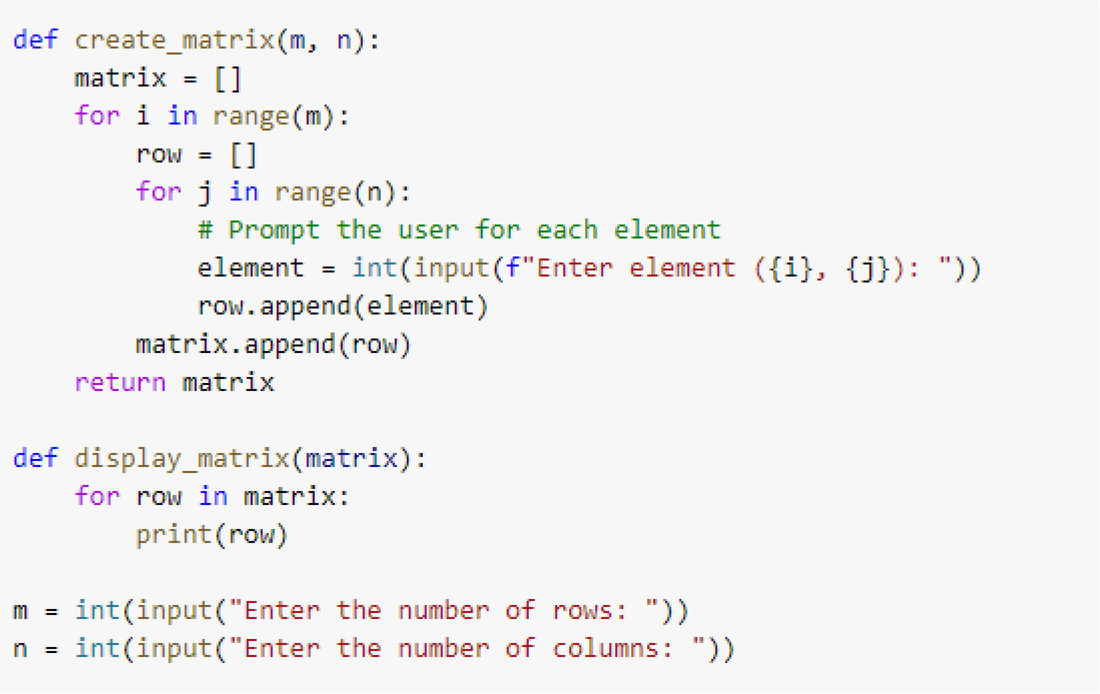
Note: You are not allowed to use any built-in function or libraries



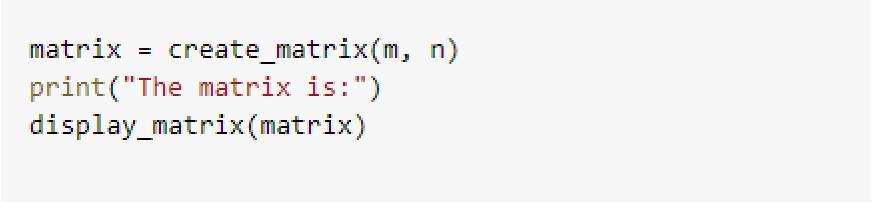
```
def dot_product(vector1, vector2):
    # Initialize a variable to store the dot product
    dot_product = 0

    # Iterate over the elements of the vectors and compute the dot product
    for i in range(len(vector1)):
        dot_product += vector1[i] * vector2[i]

    # Return the dot product
    return dot_product
vector1 = [1, 2, 3]
vector2 = [0, 1, 2]

result = dot_product(vector1, vector2)

print(result)
```

```
8
```

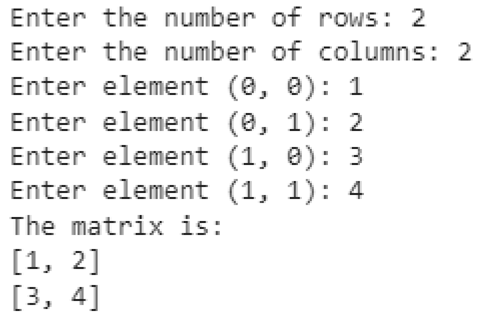
Q6) Write a program to create a matrix of dimensions m x n without using any additional libraries and create a function to display the values.



```python
def create_matrix(m, n):
    matrix = []
    for i in range(m):
        row = []
        for j in range(n):
            # Prompt the user for each element
            element = int(input(f"Enter element ({i}, {j}): "))
            row.append(element)
        matrix.append(row)
    return matrix

def display_matrix(matrix):
    for row in matrix:
        print(row)

m = int(input("Enter the number of rows: "))
n = int(input("Enter the number of columns: "))
```

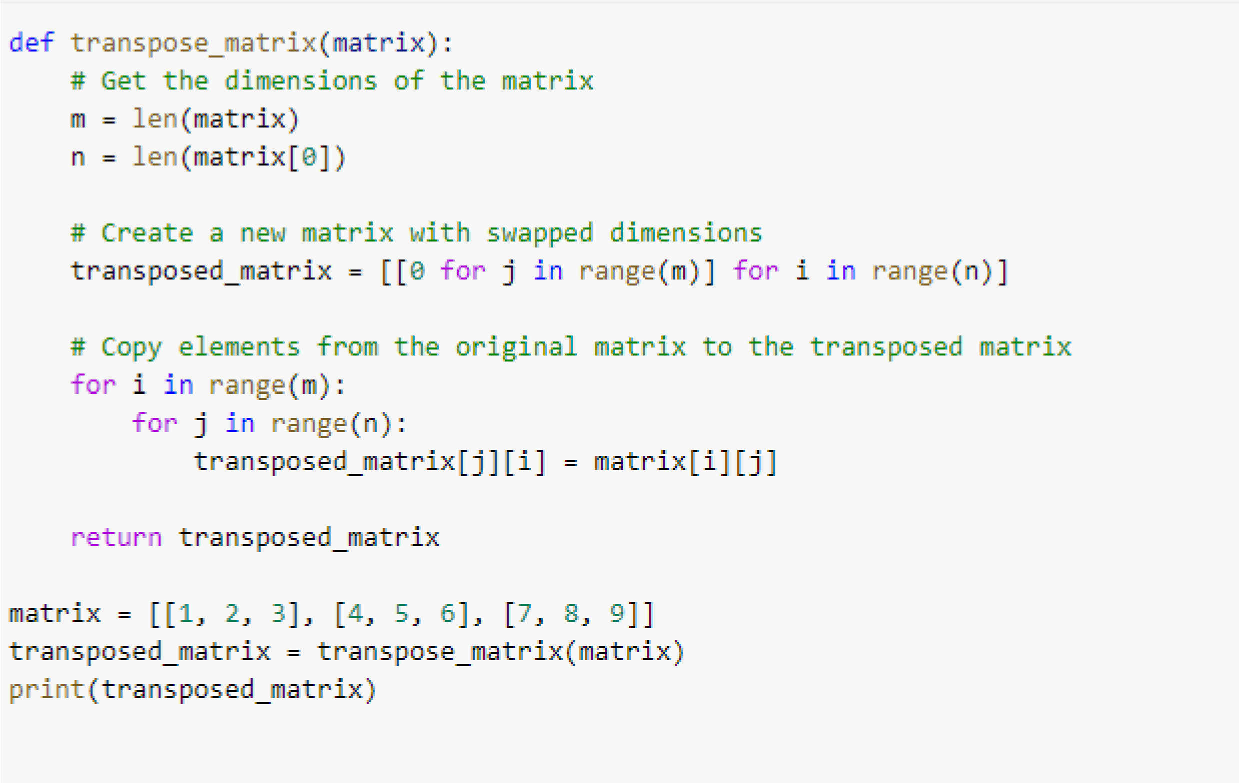


```python
matrix = create_matrix(m, n)
print("The matrix is:")
display_matrix(matrix)
```

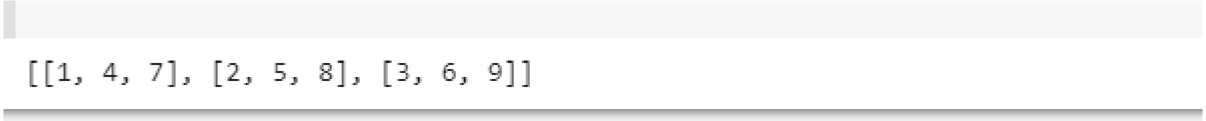


```
Enter the number of rows: 2
Enter the number of columns: 2
Enter element (0, 0): 1
Enter element (0, 1): 2
Enter element (1, 0): 3
Enter element (1, 1): 4
The matrix is:
[1, 2]
[3, 4]
```

Q7) Create a function that calculates the transpose of a given matrix.

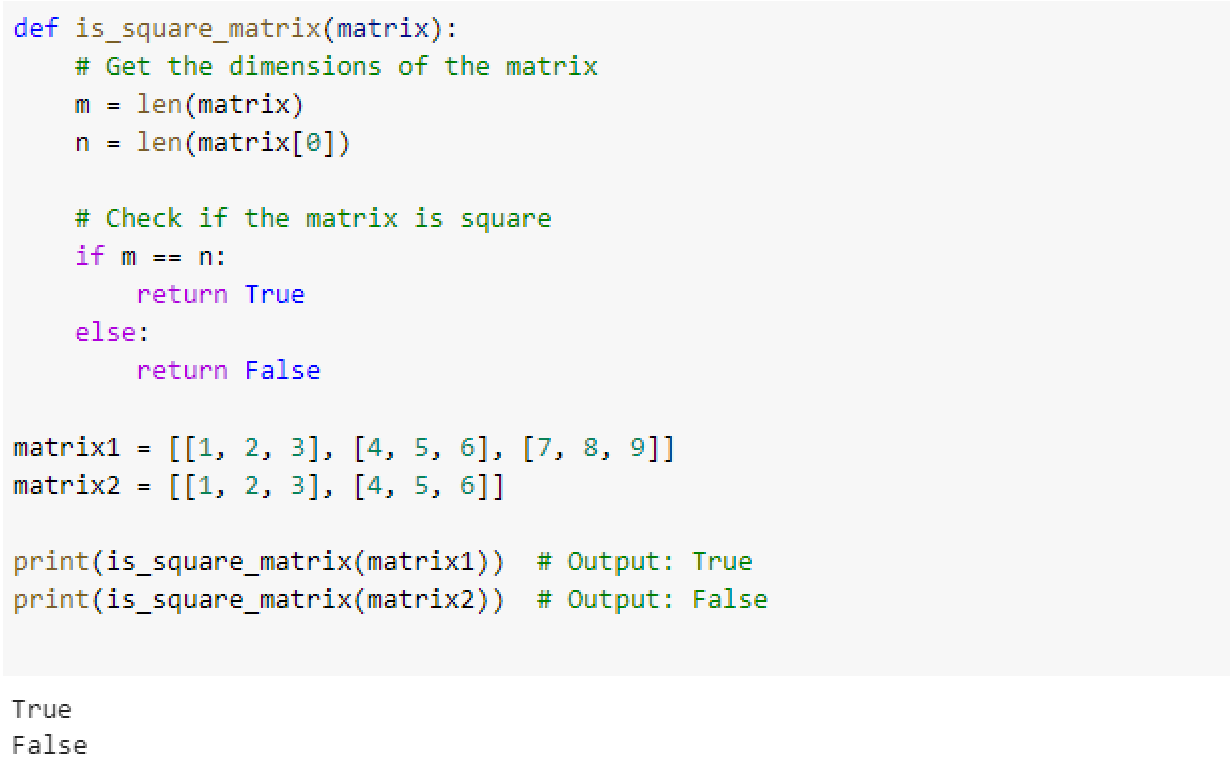


```python
def transpose_matrix(matrix):
    # Get the dimensions of the matrix
    m = len(matrix)
    n = len(matrix[0])

    # Create a new matrix with swapped dimensions
    transposed_matrix = [[0 for j in range(m)] for i in range(n)]

    # Copy elements from the original matrix to the transposed matrix
    for i in range(m):
        for j in range(n):
            transposed_matrix[j][i] = matrix[i][j]

    return transposed_matrix

matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
transposed_matrix = transpose_matrix(matrix)
print(transposed_matrix)
```



```
[[1, 4, 7], [2, 5, 8], [3, 6, 9]]
```

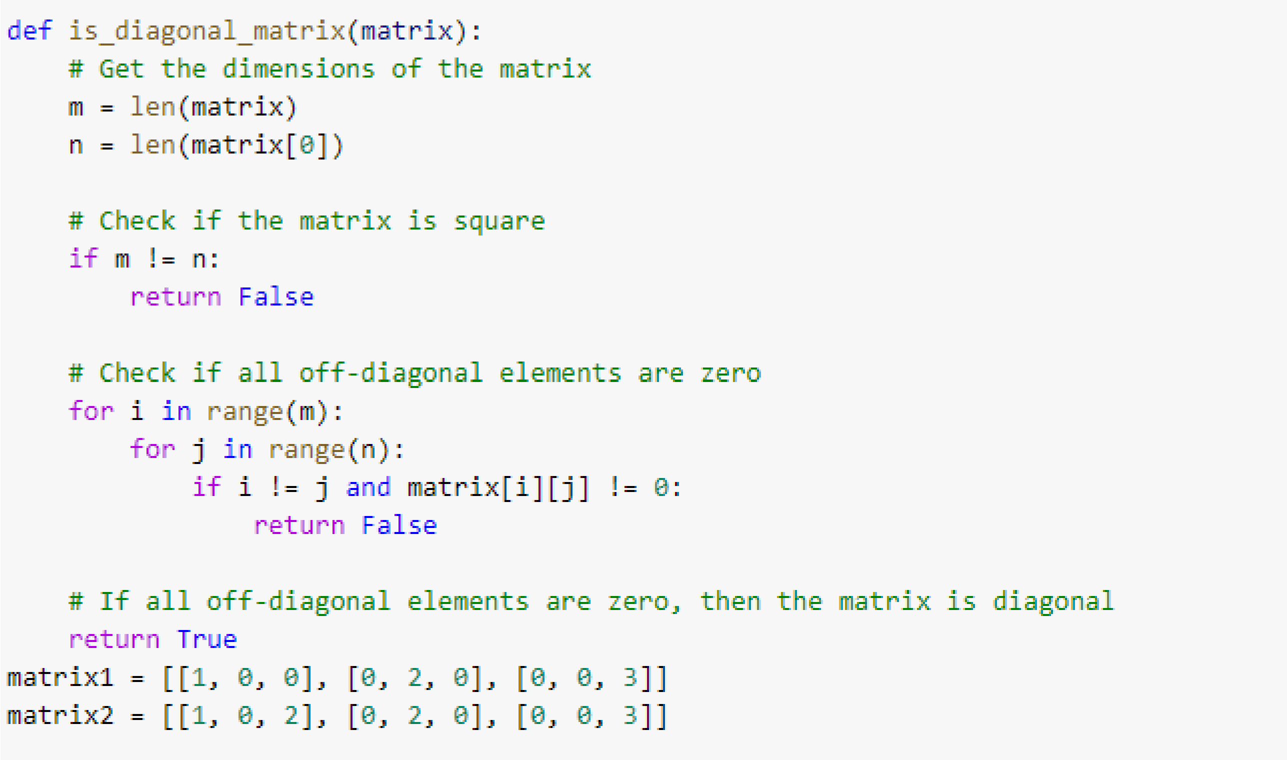
Q8) Create a function that can determine if a given matrix is a square matrix or not.



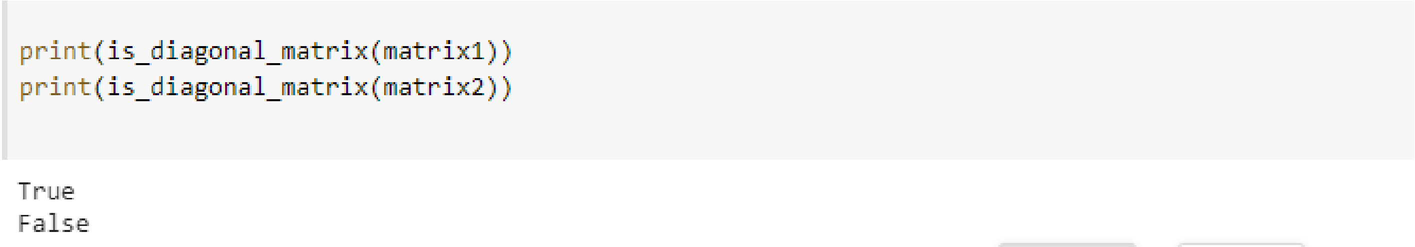
```
def is_square_matrix(matrix):
    # Get the dimensions of the matrix
    m = len(matrix)
    n = len(matrix[0])

    # Check if the matrix is square
    if m == n:
        return True
    else:
        return False

matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
matrix2 = [[1, 2, 3], [4, 5, 6]]

print(is_square_matrix(matrix1))  # Output: True
print(is_square_matrix(matrix2))  # Output: False
```

```
True
False
```

Q9) Write a program to identify the given matrix is diagonal matrix or not.



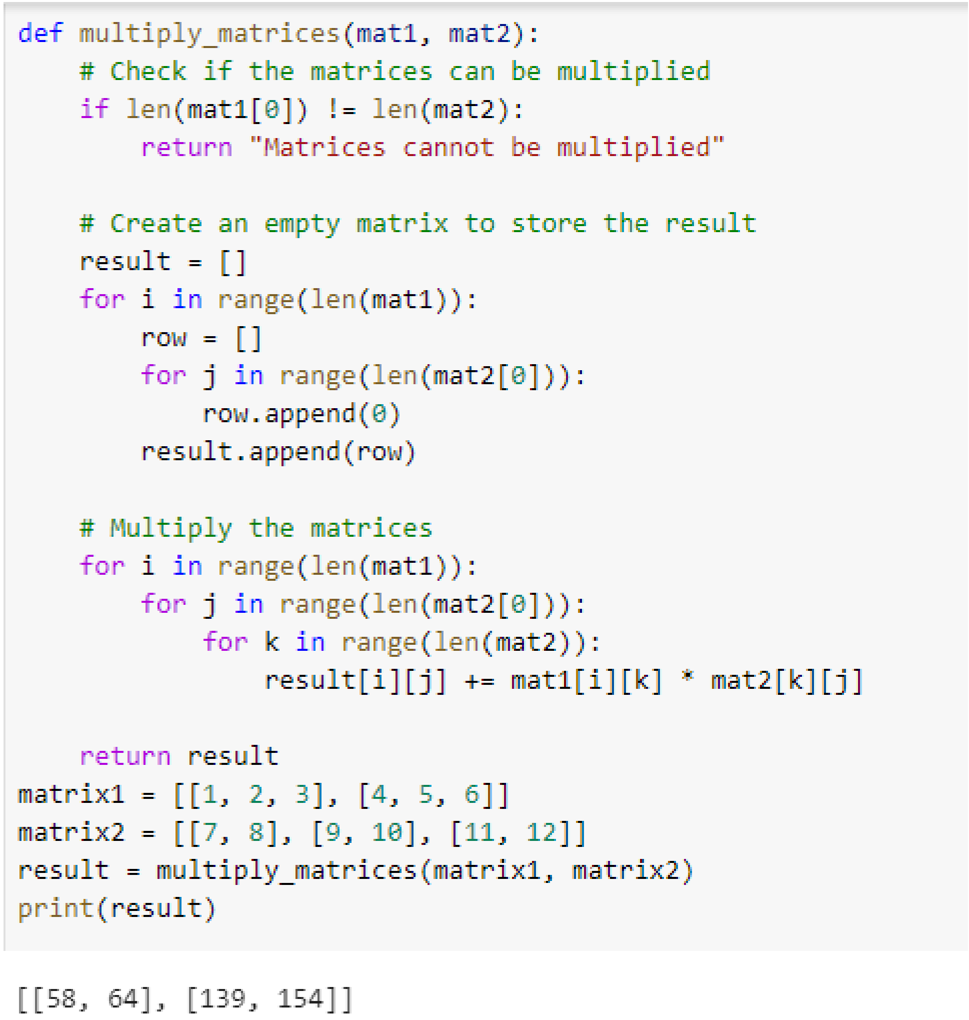
```python
def is_diagonal_matrix(matrix):
    # Get the dimensions of the matrix
    m = len(matrix)
    n = len(matrix[0])

    # Check if the matrix is square
    if m != n:
        return False

    # Check if all off-diagonal elements are zero
    for i in range(m):
        for j in range(n):
            if i != j and matrix[i][j] != 0:
                return False

    # If all off-diagonal elements are zero, then the matrix is diagonal
    return True
matrix1 = [[1, 0, 0], [0, 2, 0], [0, 0, 3]]
matrix2 = [[1, 0, 2], [0, 2, 0], [0, 0, 3]]
```



```python
print(is_diagonal_matrix(matrix1))
print(is_diagonal_matrix(matrix2))
```

```
True
False
```

Q9) Create a function that performs multiplication of two matrices.



```
def multiply_matrices(mat1, mat2):
    # Check if the matrices can be multiplied
    if len(mat1[0]) != len(mat2):
        return "Matrices cannot be multiplied"

    # Create an empty matrix to store the result
    result = []
    for i in range(len(mat1)):
        row = []
        for j in range(len(mat2[0])):
            row.append(0)
        result.append(row)

    # Multiply the matrices
    for i in range(len(mat1)):
        for j in range(len(mat2[0])):
            for k in range(len(mat2)):
                result[i][j] += mat1[i][k] * mat2[k][j]

    return result
matrix1 = [[1, 2, 3], [4, 5, 6]]
matrix2 = [[7, 8], [9, 10], [11, 12]]
result = multiply_matrices(matrix1, matrix2)
print(result)
```

```
[[58, 64], [139, 154]]
```