**Code Demo 2: Array Operations**

Write a Python function called `array\_operations` that takes in two parameters: `n` (an integer) and `operation` (a string). The function should perform the following operations:

1. If the `operation` is "square", create a NumPy array of shape `(n, n)` containing the squares of the numbers from 1 to `n^2`.

2. If the `operation` is "multiply", create a NumPy array of shape `(n, n)` containing the multiplication table from 1 to `n`. Each element at index `(i, j)` should be the product of `i+1` and `j+1`.

3. If the `operation` is "identity", create a NumPy array of shape `(n, n)` representing an identity matrix.

The function should return the resulting NumPy array based on the specified operation.

Example usage:

| import numpy as np  def array\_operations(n, operation):  if operation == "square":  *# Create array of squares*  pass  elif operation == "multiply":  *# Create multiplication table*  pass  elif operation == "identity":  *# Create identity matrix*  pass  else:  raise ValueError("Invalid operation!")   return array  *# Test cases* print(array\_operations(4, "square")) print(array\_operations(5, "multiply")) print(array\_operations(3, "identity")) |
| --- |

Output:

| [[ 1 4 9 16]  [25 36 49 64]  [81 100 121 144]  [169 196 225 256]]   [[ 1 2 3 4 5]  [ 2 4 6 8 10]  [ 3 6 9 12 15]  [ 4 8 12 16 20]  [ 5 10 15 20 25]]   [[1. 0. 0.]  [0. 1. 0.]  [0. 0. 1.]] |
| --- |

This exercise allows you to practice creating NumPy arrays and performing different operations based on the input parameters.