OUTPUT

```
Problem 1: Array Creation
1. Empty array (2x2):
 [[2.5e-323 1.5e-323]
 [9.9e-324 4.9e-324]]
2. All ones array (4x2):
 [[1. 1.]
 [1. 1.]
 [1. 1.]
 [1. 1.]]
3. Full array (3x3) with 7:
 [[7 7 7]
 [7 7 7]
 [7 7 7]]
4. Zeros like ref_array:
 [[0 0]
 [0 0]]
5. Ones like ref_array:
 [[1 1]
 [1 1]]
6. Converted list to array:
 [1 2 3 4]
```

```
Problem 2: Array Manipulation
1. Array from 10 to 49:
 [10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49]
2. 3x3 matrix from 0 to 8:
 [[0 1 2]
 [3 4 5]
 [6 7 8]]
3. Identity matrix (3x3):
 [[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
4. Random array (size 30):
 [0.26143264 0.37414919 0.28403579 0.98896243 0.81537497 0.57485355
 0.17155103 0.46464771 0.50093071 0.60639015 0.60202635 0.79516521
 0.20043409 0.85479629 0.83560969 0.073964   0.90757913 0.88649434
 0.30337049 0.640167   0.17289244 0.77021465 0.50821354 0.18054954
 0.10841842 0.51365624 0.2503808  0.89720993 0.17102178 0.53608359]
Mean of array: 0.5083525228924983
5. Random 10x10 array:
 [[0.41414259 0.37968643 0.57799274 0.5939736  0.43969467 0.73054279
  0.1986308  0.93063442 0.45350796 0.38915608]
 [0.32841124 0.60341673 0.0729663  0.54658411 0.30051515 0.6815966
  0.6217933  0.06640462 0.93242504 0.82344994]
 [0.08970277 0.95890203 0.8028182  0.76512042 0.65888964 0.44245319
  0.59387265 0.60802971 0.57708848 0.59361043]
 [0.3706881  0.10730926 0.1326251  0.1983359  0.67078475 0.87328426
  0.96040642 0.83645439 0.95532712 0.33046255]
 [0.56662009 0.8602525  0.19098692 0.16075554 0.66812174 0.93782593
  0.00136552 0.97585648 0.33714239 0.02694117]
 [0.25618101 0.05152563 0.56579182 0.73782935 0.54410473 0.78599352
  0.53605122 0.50221952 0.13082864 0.3194914 ]
 [0.06883221 0.54883015 0.36851949 0.87977382 0.25928503 0.87285589
  0.06730373 0.08855948 0.17002503 0.4508455 ]
 [0.67068587 0.42541452 0.69867435 0.7176803  0.69558982 0.72993375
  0.82414492 0.05124133 0.30654402 0.40550521]
 [0.68651935 0.98038875 0.18371201 0.12005386 0.69497899 0.1438833
  0.48355111 0.81309239 0.53385808 0.52135352]
 [0.66658986 0.5666248  0.40829084 0.15855425 0.02450377 0.48256525
  0.90667474 0.50856881 0.18561126 0.16125856]]
Min: 0.0013655229400930669 Max: 0.9803887529602017
6. Zero array with 5th element = 1:
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
7. Reversed array:
 [0, 4, 0, 0, 2, 1]
8. 2D array with 1 on border and 0 inside:
 [[1. 1. 1. 1. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 0. 0. 0. 1.]
 [1. 1. 1. 1. 1.]]
9. 8x8 checkerboard pattern:
 [[0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]]


Problem 3: Array Operations
x: [[1 2]
 [3 5]]
y: [[5 6]
 [7 8]]
v: [ 9 10]
w: [11 12]
1. x + y: [[ 6  8]
 [10 13]]
2. x - y: [[-4 -4]
 [-4 -3]]
3. x * 3: [[ 3  6]
 [ 9 15]]
4. Square of x: [[ 1  4]
 [ 9 25]]
5. Dot products:
v . w = 219
x . v =
 [29 77]
```

```
x . y =
 [[19 22]
 [50 58]]
6. Concatenate x and y along rows:
 [[1 2]
 [3 5]
 [5 6]
 [7 8]]
Concatenate v and w along columns:
 [[ 9 10]
 [11 12]]
7. Concatenate x and v:
Error: all the input arrays must have same number of dimensions, but the
array at index 0 has 2 dimension(s) and the array at index 1 has 1
dimension(s)


Problem 4: Matrix Operations
1. A * A_inv: [[1.00000000e+00 0.00000000e+00]
 [1.77635684e-15 1.00000000e+00]]
2. AB: [[23 13]
 [51 29]]
BA: [[36 44]
 [13 16]]
AB == BA? False
3. (AB)^T: [[23 51]
 [13 29]]
B^T * A^T: [[23 51]
 [13 29]]
(AB)^T == B^T * A^T? True


Experiment: How Fast is NumPy?

1. Element-wise Addition
NumPy Array Addition Time: 0.003609180450439453 seconds

2. Element-wise Multiplication
NumPy Array Multiplication Time: 0.0029222965240478516 seconds

3. Dot Product
NumPy Array Dot Product Time: 0.002206563949584961 seconds

4. Matrix Multiplication
NumPy Matrix Multiplication Time: 2.3309781551361084 seconds
```