Exercises

1. Import the numpy package under the name `np` (★☆☆)

In [1]:

```
import numpy as np
```

2. Print the numpy version and the configuration (★☆☆)

In [5]:

```
print(np.__version__)
np.show_config()
```

```
1.26.0
Build Dependencies:
  blas:
    detection method: pkgconfig
    found: true
    include directory: /opt/arm64-builds/include
    lib directory: /opt/arm64-builds/lib
    name: openblas64
    openblas configuration: USE_64BITINT=1 DYNAMIC_ARCH=1 DYNAMIC_OLDER= N
O_CBLAS=
      NO_LAPACK= NO_LAPACKE= NO_AFFINITY=1 USE_OPENMP= SANDYBRIDGE MAX_THR
EADS=3
    pc file directory: /usr/local/lib/pkgconfig
    version: 0.3.23.dev
  lapack:
    detection method: pkgconfig
    found: true
    include directory: /opt/arm64-builds/include
    lib directory: /opt/arm64-builds/lib
    name: openblas64
    openblas configuration: USE_64BITINT=1 DYNAMIC_ARCH=1 DYNAMIC_OLDER= N
O_CBLAS=
      NO_LAPACK= NO_LAPACKE= NO_AFFINITY=1 USE_OPENMP= SANDYBRIDGE MAX_THR
EADS=3
    pc file directory: /usr/local/lib/pkgconfig
    version: 0.3.23.dev
Compilers:
  c:
    commands: cc
    linker: ld64
    name: clang
    version: 14.0.0
  c++:
    commands: c++
    linker: ld64
    name: clang
    version: 14.0.0
  cython:
    commands: cython
    linker: cython
    name: cython
    version: 3.0.2
Machine Information:
  build:
    cpu: aarch64
    endian: little
    family: aarch64
    system: darwin
  host:
    cpu: aarch64
    endian: little
    family: aarch64
    system: darwin
Python Information:
  path: /private/var/folders/76/zy5ktkns50v6gt5g8r0sf6sc0000gn/T/cibw-run-
4sgjw1qw/cp311-macosx_arm64/build/venv/bin/python
  version: '3.11'
SIMD Extensions:
  baseline:
  - NEON
  - NEON_FP16
  - NEON_VFPV4
  - ASIMD
  found:
  - ASIMDHP
  not found:
  - ASIMDFHM
```

3. Create a null vector of size 10 (★☆☆)

In [6]:

```
Z = np.zeros(10)
print(Z)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

4. Create a null vector of size 10 but the fifth value which is 1 (★☆☆)

In [7]:

```
Z = np.zeros(10)
Z[4] = 1
print(Z)
```

```
[0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
```

5. Create a vector with values ranging from 10 to 49 (★☆☆)

In [8]:

```
Z = np.arange(10,50)
print(Z)
```

```
[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]
```

6. Reverse a vector (first element becomes last) (★☆☆)

In [9]:

```
Z = np.arange(50)
Z = Z[::-1]
print(Z)
```

```
[49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26
 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2
  1  0]
```

7. Create a 3x3 matrix with values ranging from 0 to 8 (★☆☆)

In [10]:

```
Z = np.arange(9).reshape(3, 3)
print(Z)
```

```
[[0 1 2]
 [3 4 5]
 [6 7 8]]
```

8. Find indices of non-zero elements from [1,2,0,0,4,0] (★☆☆)

In [11]:

```
nz = np.nonzero([1,2,0,0,4,0])
print(nz)
```

```
(array([0, 1, 4]),)
```

9. Create a 3x3 identity matrix (★☆☆)

In [13]:

```
Z = np.eye(3)
print(Z)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

In [12]:

```
X = np.identity(3)
print(X)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

10. Create a 3x3x3 array with random values (★☆☆)

In [14]:

```
Z = np.random.random((3,3,3))
print(Z)
```

```
[[[0.81534712 0.8953453  0.24423403]
  [0.62667507 0.75575832 0.43838019]
  [0.41391296 0.10782793 0.10117504]]

 [[0.11840819 0.17165397 0.21465952]
  [0.32099797 0.34694004 0.07579493]
  [0.05417501 0.51773321 0.05728572]]

 [[0.92868687 0.61879969 0.76077216]
  [0.7971539  0.3578063  0.6896685 ]
  [0.88089514 0.08022359 0.70686741]]]
```

11. Create a 10x10 array with random values and find the minimum and maximum values (★☆☆)

In [15]:

```
Z = np.random.random((10,10))
```

```
Zmin, Zmax = Z.min(), Z.max()
print(Zmin, Zmax)
```

```
0.0007156363747757855 0.989063718252267
```

12. Create a random vector of size 30 and find the mean value (★☆☆)

In [16]:

```
Z = np.random.random(30)
m = Z.mean()
print(m)
```

```
0.5306539455700996
```

13. Create a 2d array with 1 on the border and 0 inside (★☆☆)

In [21]:

```
Z = np.ones((5,5))
Z[1:-1,1:-1] = 0
print(Z)
```

```
[[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.]]
```

14.Create a 2d array with 0 on the border and 1 inside (★☆☆)

In [19]:

```
Z = np.zeros((5,5))
Z[1:-1,1:-1] = 1
print(Z)
```

```
[[0. 0. 0. 0. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 0. 0. 0. 0.]]
```

In [20]:

```
Z = np.ones((5,5))
# Using fancy indexing
Z[:, [0, -1]] = 0
Z[[0, -1], :] = 0
print(Z)
```

```
[[0. 0. 0. 0. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 0. 0. 0. 0.]]
```

15. Create a 8x8 matrix and fill it with a checkerboard pattern (★☆☆)

In [25]:

```
Z = np.zeros((8,8),dtype=int)
Z[1::2,::2] = 1
Z[::2,1::2] = 1
print(Z)
```

```
[[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]]
```

16. Normalize a 5x5 random matrix (★☆☆)

- หมายถึงการปรับสเกลของค่าทั้งหมดในเมทริกซ์สุ่มขนาด 5x5 ให้มีคุณสมบัติตามที่เรากำหนดไว้
- โดยในที่นี้ให้ทำ Z-Score Normalization:เปลี่ยนค่าในเมทริกซ์ให้มีค่าเฉลี่ยเป็น 0 และส่วนเบี่ยงเบนมาตรฐานเป็น 1 โดยใช้สูตร x_norm = x−μ/σ

In [29]:

```
Z = np.random.random((5,5))
Z = (Z - np.mean (Z)) / (np.std (Z))
print(Z)
```

```
[[ 0.29098369 -0.56077788 -1.6518406  -1.51411043  0.08672838]
 [ 0.42007465 -0.91430687  1.26479929 -1.36568926 -0.38634809]
 [ 1.29001516 -0.26933732 -1.85204751 -0.22000504  0.80819425]
 [ 1.7685984  -1.35945832  0.52641039 -0.19224262  0.51844786]
 [ 1.60808689  0.10715933  0.67158348  0.317427    0.6076552 ]]
```

17. Multiply a 5x3 matrix by a 3x2 matrix (real matrix product) (★☆☆)

In [32]:

```
print(np.ones((5,3)))
print(np.ones((3,2)))
Z = np.dot(np.ones((5,3)), np.ones((3,2)))
print(Z)
```

```
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
[[1. 1.]
 [1. 1.]
 [1. 1.]]
[[3. 3.]
 [3. 3.]
 [3. 3.]
 [3. 3.]
 [3. 3.]]
```

18. Create a random vector of size 10 and sort it (★★☆)

In [52]:

```
Z = np.random.random(10)
Z.sort()
print(Z)
```

```
[0.07962669 0.12920713 0.19556228 0.36152301 0.47562931 0.51225313
 0.60582025 0.69161861 0.83433671 0.92145505]
```

19. Create random vector of size 10 and replace the maximum value by 0 (★★☆)

```
In [51]: Z = np.random.random(10)
         Z[Z.argmax()] = 0
         print(Z)
```

```
[0.82330768 0.79521354 0.40718333 0.         0.49809601 0.69250456
 0.15006329 0.37306895 0.43055947 0.79349812]
```

20. Create a 5x5 matrix with row values ranging from 0 to 4 (★★☆)

```
In [45]: Z = np.zeros((5,5))
         Z += np.arange(5)
         print(Z)
```

```
[[0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]
 [0. 1. 2. 3. 4.]]
```