NumPy exercises

In [ ]:

```
# Import the numpy package under the name np
import numpy as np

# Print the numpy version and the configuration
print(np.__version__)
```

Array creation

Create a numpy array of size 10, filled with zeros.

In [ ]:

```
# your code goes here
```

In [ ]:

```
#np.array([0] * 10)
np.zeros(10)
```

Create a numpy array with values ranging from 10 to 49

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.arange(10,50)
```

Create a numpy matrix of 2*2 integers, filled with ones.

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.ones([2,2], dtype=np.int)
```

Create a numpy matrix of 3*2 float numbers, filled with ones.

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.ones([3,2], dtype=np.float)
```

Given the X numpy array, create a new numpy array with the same shape and type as X, filled with ones.

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.arange(4, dtype=np.int)

np.ones_like(X)
```

Given the X numpy matrix, create a new numpy matrix with the same shape and type as X, filled with zeros.

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([[1,2,3], [4,5,6]], dtype=np.int)

np.zeros_like(X)
```

Create a numpy matrix of 4*4 integers, filled with fives.

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.ones([4,4], dtype=np.int) * 5
```

Given the X numpy matrix, create a new numpy matrix with the same shape and type as X, filled with sevens.

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([[2,3], [6,2]], dtype=np.int)

np.ones_like(X) * 7
```

Create a 3*3 identity numpy matrix with ones on the diagonal and zeros elsewhere.

In [ ]:

```
# your code goes here
```

In [ ]:

```
#np.eye(3)
np.identity(3)
```

Create a numpy array, filled with 3 random integer values between 1 and 10.

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.random.randint(10, size=3)
```

Create a 3*3*3 numpy matrix, filled with random float values.

In [ ]:

```
# your code goes here
```

In [ ]:

```
#np.random.random((3,3,3))
np.random.randn(3,3,3) # 0 to 1 floats
```

Given the X python list convert it to an Y numpy array

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = [1, 2, 3]
print(X, type(X))

Y = np.array(X)
print(Y, type(Y)) # different type
```

Given the X numpy array, make a copy and store it on Y.

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([5,2,3], dtype=np.int)
print(X, id(X))

Y = np.copy(X)
print(Y, id(Y)) # different id
```

Create a numpy array with numbers from 1 to 10

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.arange(1, 11)
```

Create a numpy array with the odd numbers between 1 to 10

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.arange(1, 11, 2)
```

Create a numpy array with numbers from 1 to 10, in descending order.

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.arange(1, 11)[::-1]
```

Create a 3*3 numpy matrix, filled with values ranging from 0 to 8

In [ ]:

```
# your code goes here
```

In [ ]:

```
np.arange(9).reshape(3,3)
```

Show the memory size of the given Z numpy matrix

In [ ]:

```
# your code goes here
```

In [ ]:

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

print("%d bytes" % (Z.size * Z.itemsize))
```

Array indexation

Given the X numpy array, show it's first element

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array(['A','B','C','D','E'])

X[0]
```

Given the X numpy array, show it's last element

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array(['A','B','C','D','E'])

#X[len(X)-1]
X[-1]
```

Given the X numpy array, show it's first three elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array(['A','B','C','D','E'])

X[0:3] # remember! elements start at zero index
```

Given the X numpy array, show all middle elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array(['A','B','C','D','E'])

X[1:-1]
```

Given the X numpy array, show the elements in reverse position

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array(['A','B','C','D','E'])

X[::-1]
```

Given the X numpy array, show the elements in an odd position

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array(['A','B','C','D','E'])

#X[[0, 2, -1]]
X[::2]
```

Given the X numpy matrix, show the first row elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,  2,  3,  4],
    [5,  6,  7,  8],
    [9, 10, 11, 12],
    [13, 14, 15, 16]
])

X[0]
```

Given the X numpy matrix, show the last row elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

X[-1]
```

Given the X numpy matrix, show the first element on first row

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

#X[0][0]
X[0, 0]
```

Given the X numpy matrix, show the last element on last row

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

#X[-1][-1]
X[-1, -1]
```

Given the X numpy matrix, show the middle row elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

#X[1:-1][1:-1] wrong!
X[1:-1, 1:-1]
```

Given the X numpy matrix, show the first two elements on the first two rows

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

#X[:2][:2] wrong!
#X[0:2, 0:2]
X[:2, :2]
```

Given the X numpy matrix, show the last two elements on the last two rows

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

X[2:, 2:]
```

Array manipulation

Convert the given integer numpy array to float

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = [-5, -3, 0, 10, 40]

np.array(X, np.float)
```

Reverse the given numpy array (first element becomes last)

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = [-5, -3, 0, 10, 40]

X[::-1]
```

Order (sort) the given numpy array

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = [0, 10, -5, 40, -3]

X.sort()
X
```

Given the X numpy array, set the fifth element equal to 1

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.zeros(10)

X[4] = 1
X
```

Given the X numpy array, change the 50 with a 40

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([10, 20, 30, 50])

X[3] = 40
X
```

Given the X numpy matrix, change the last row with all 1

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

X[-1] = np.array([1, 1, 1, 1])
X
```

Given the X numpy matrix, change the last item on the last row with a 0

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

X[-1, -1] = 0
X
```

Given the X numpy matrix, add 5 to every element

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

X + 5
```

Boolean arrays *(also called masks)*

Given the X numpy array, make a mask showing negative elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([-1,2,0,-4,5,6,0,0,-9,10])

mask = X <= 0
mask
```

Given the X numpy array, get the negative elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([-1, 2, 0, -4, 5, 6, 0, 0, -9, 10])

mask = X <= 0
X[mask]
```

Given the X numpy array, get numbers higher than 5

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([-1, 2, 0, -4, 5, 6, 0, 0, -9, 10])

mask = X > 5
X[mask]
```

Given the X numpy array, get numbers higher than the elements mean

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([-1, 2, 0, -4, 5, 6, 0, 0, -9, 10])

mask = X > X.mean()
X[mask]
```

Given the X numpy array, get numbers equal to 2 or 10

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([-1, 2, 0, -4, 5, 6, 0, 0, -9, 10])

mask = (X == 2) | (X == 10)
X[mask]
```

Logic functions

Given the X numpy array, return True if none of its elements is zero

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([-1, 2, 0, -4, 5, 6, 0, 0, -9, 10])

X.all()
```

Given the X numpy array, return True if any of its elements is zero

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([-1, 2, 0, -4, 5, 6, 0, 0, -9, 10])

X.any()
```

Summary statistics

Given the X numpy array, show the sum of its elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([3, 5, 6, 7, 2, 3, 4, 9, 4])

#np.sum(X)
X.sum()
```

Given the X numpy array, show the mean value of its elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([1, 2, 0, 4, 5, 6, 0, 0, 9, 10])

#np.mean(X)
X.mean()
```

Given the X numpy matrix, show the sum of its columns

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

X.sum(axis=0) # remember: axis=0 columns; axis=1 rows
```

Given the X numpy matrix, show the mean value of its rows

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([
    [1,   2,  3,  4],
    [5,   6,  7,  8],
    [9,  10, 11, 12],
    [13, 14, 15, 16]
])

X.mean(axis=1) # remember: axis=0 columns; axis=1 rows
```

Given the X numpy array, show the max value of its elements

In [ ]:

```
# your code goes here
```

In [ ]:

```
X = np.array([1, 2, 0, 4, 5, 6, 0, 0, 9, 10])

#np.max(X)
X.max()
```