import numpy as np

# 1. Create arrays

a = np.array([1, 2, 3])

b = np.arange(0, 10, 2) # [0,2,4,6,8]

c = np.zeros((3,4))

d = np.ones((2,2), dtype=float)

# 2. Attributes

a.shape # dimensions

a.ndim # number of axes

a.size # total number of elements

a.dtype # data type

# 3. Indexing & slicing

A = np.arange(12).reshape(3,4)

row1 = A[1, :] # second row

col2 = A[:, 2] # third column

sub = A[0:2, 1:3] # slice

# Boolean mask filtering

mask = A > 5

A[mask] # elements > 5

# 4. Reshaping / flattening / stacking

v = np.arange(6)

v2 = v.reshape(2,3)

flat = v2.ravel()

stacked = np.vstack([v2, v2]) # vertical stacking

hstack = np.hstack([v2, v2]) # horizontal stacking

# 5. Broadcasting & arithmetic

# Suppose x is (5,3) and bias is (3,)

# broadcasting lets you add bias to each row of x

x + bias

# elementwise ops

x \* 2, x \*\* 2, np.exp(x), np.log(x)

# 6. Linear algebra & matrix ops

X = np.random.random((3,3))

y = np.random.random((3,1))

Xt = X.T

prod = X @ y # matrix multiply

dot = np.dot(X, y)

# inverse or pseudo-inverse (if square and invertible)

invX = np.linalg.inv(X)

pinv = np.linalg.pinv(X)

# 7. Random / sampling

rng = np.random.default\_rng(42)

r = rng.random((3,3)) # uniform random in [0,1)

ints = rng.integers(low=0, high=10, size=5)

perm = rng.permutation(10) # shuffle indices

# 8. Aggregations / axis

arr = np.arange(12).reshape(3,4)

arr.sum(axis=0) # sum each column

arr.mean(axis=1) # mean of each row

arr.min(), arr.max(), arr.std()

# 9. Copy vs view

v = np.arange(5)

v2 = v.view()

v3 = v.copy()

v2[0] = 100 # also changes v

v3[0] = 200 # does not change v