Numpy

Collection Data Types

- List: my_list = [3, 8, 1, 6, 0, 8, 4]
- Tuple: thistuple = ("apple", "banana", "cherry")
- Sets : thisset = {"apple", "banana", "cherry"}
- Dictionary: thisdict = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
  }

Numpy

- import numpy as np
- cvalues = [20.1, 20.8, 21.9, 22.5, 22.7, 22.3, 21.8, 21.2, 20.9, 20.1]
- C = np.array(cvalues)
- print(C)
- print(C * 9 / 5 + 32)
- Type(C)

Arange

- import numpy as np

  a = np.arange(1, 10)

  print(a)
- x = np.arange(0.5, 10.4, 0.8)

  print(x)
- x = np.arange(0.5, 10.4, 0.8, int)

  print(x)

linspace

- import numpy as np

  # 50 values between 1 and 10:
- print(np.linspace(1, 10))

  # 7 values between 1 and 10:
- print(np.linspace(1, 10, 7))

Zero-dimensional Arrays in Numpy

- import numpy as np

  x = np.array(42)

  print("x: ", x)

  print("The type of x: ", type(x))

  print("The dimension of x:", np.ndim(x))

One-dimensional Arrays

- F = np.array([1, 1, 2, 3, 5, 8, 13, 21])
- V = np.array([3.4, 6.9, 99.8, 12.8])
- print("F: ", F)
- print("V: ", V)
- print("Type of F: ", F.dtype)
- print("Type of V: ", V.dtype)
- print("Dimension of F: ", np.ndim(F))
- print("Dimension of V: ", np.ndim(V))

Two- and Multidimensional Arrays

- A = np.array([[3.4, 8.7, 9.9],
  [1.1, -7.8, -0.7],
  [4.1, 12.3, 4.8]])
- print(A)
- print(A.ndim)

- B = np.array([[[111, 112], [121, 122]],
  [[211, 212], [221, 222]],
  [[311, 312], [321, 322]]])
- print(B)
- print(B.ndim)

Shape of an Array

- x = np.array([[67, 63, 87],
  [77, 69, 59],
  [85, 87, 99],
  [79, 72, 71],
  [63, 89, 93],
  [68, 92, 78]])
- print(np.shape(x))

- x.shape = (3, 6)

  print(x)
- x.shape = (2, 9)

  print(x)

- B = np.array([[[111, 112, 113], [121, 122, 123]],
  [[211, 212, 213], [221, 222, 223]],
  [[311, 312, 313], [321, 322, 323]],
  [[411, 412, 413], [421, 422, 423]]])
- print(B.shape)

Indexing and Slicing

- F = np.array([1, 1, 2, 3, 5, 8, 13, 21])

  # print the first element of F

  print(F[0])

  # print the last element of F

  print(F[-1])

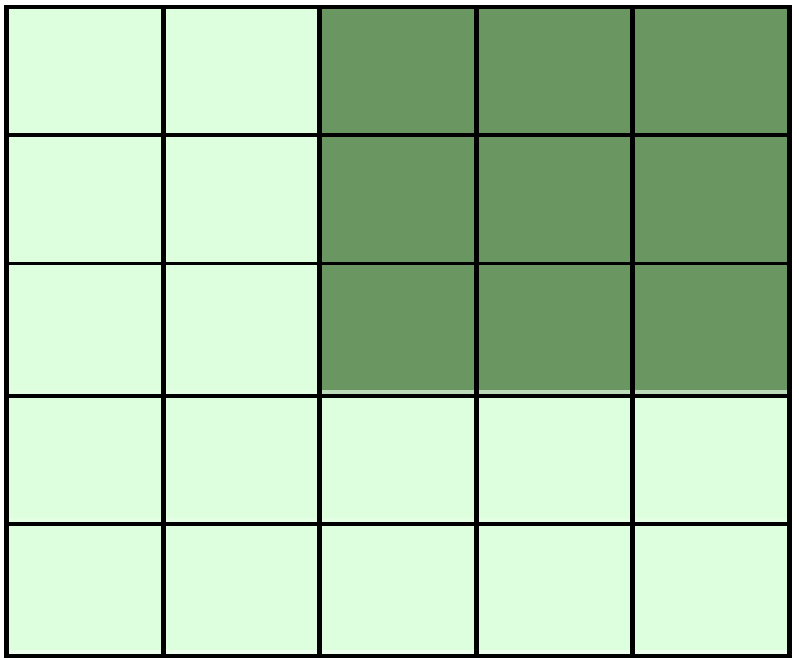
- A = np.array([[3.4, 8.7, 9.9],
  [1.1, -7.8, -0.7],
  [4.1, 12.3, 4.8]])

  print(A[1][0])

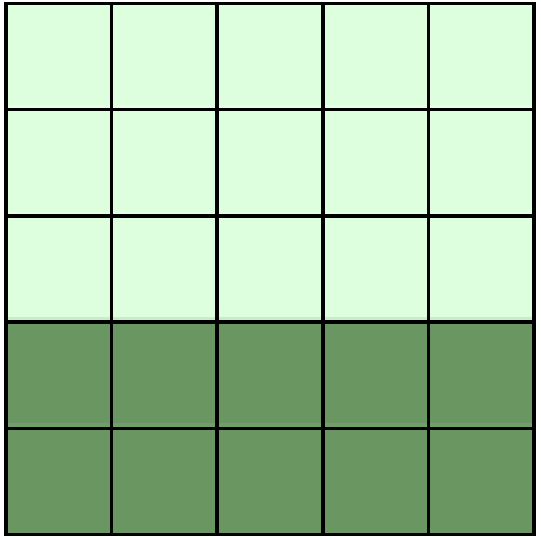
slicing

- S = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
- print(S[2:5])
- print(S[:4])
- print(S[6:])
- print(S[:])

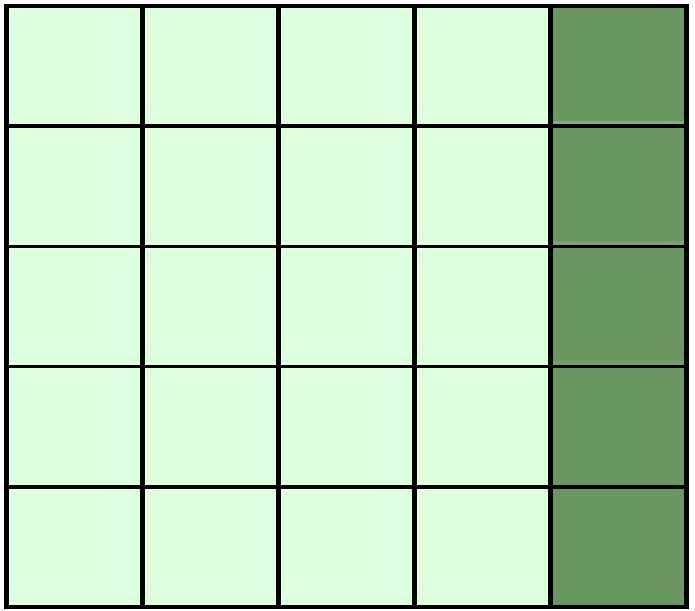
- A = np.array([
  [11, 12, 13, 14, 15],
  [21, 22, 23, 24, 25],
  [31, 32, 33, 34, 35],
  [41, 42, 43, 44, 45],
  [51, 52, 53, 54, 55]])
  print(A[:3, 2:])



- print(A[3:, :])

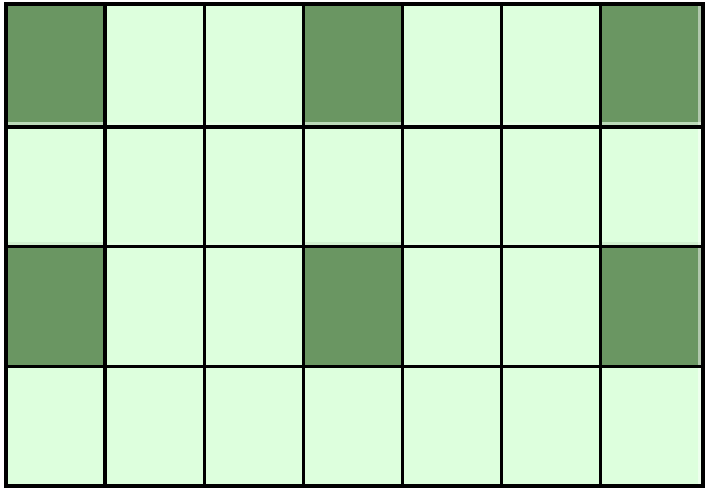


- print(A[:, 4:])
-

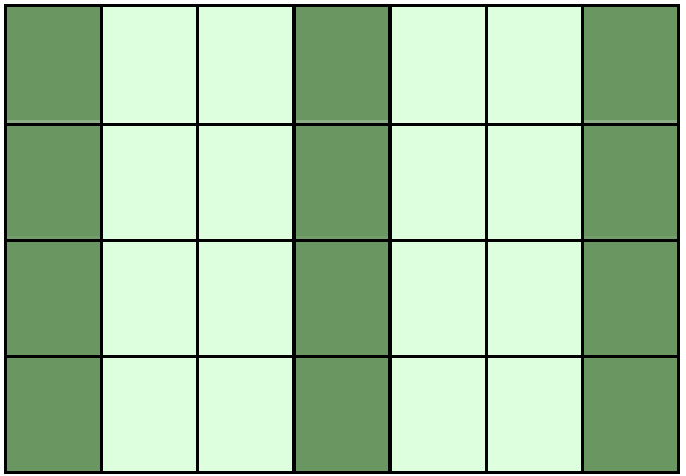


- X = np.arange(28).reshape(4, 7)

  print(X)
- print(X[::2, ::3])



- print(X[::, ::3])



The identity Function

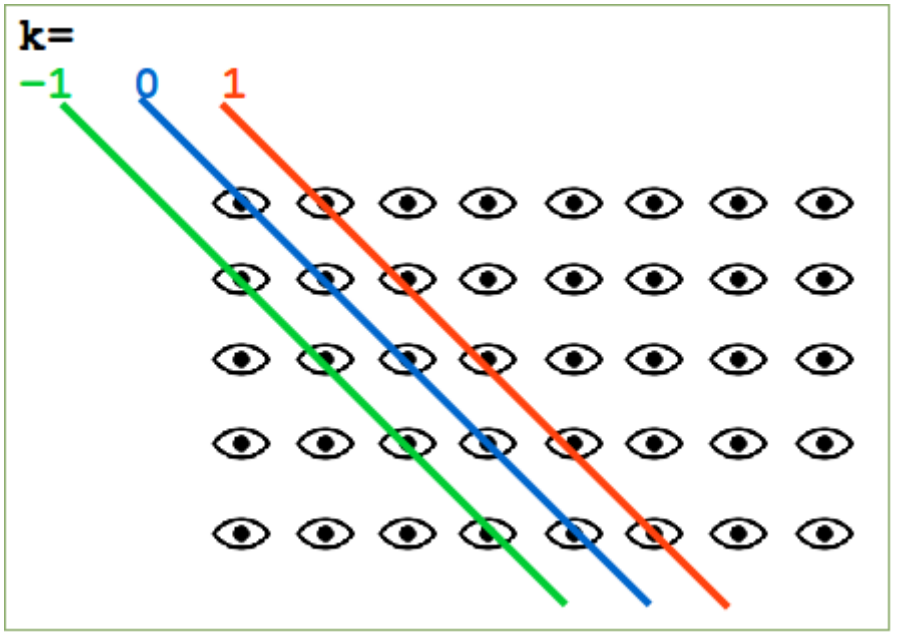
- import numpy as np

  np.identity(4)
- np.identity(4, dtype=int)

The eye Function

- import numpy as np

  np.eye(5, 8, k=1, dtype=int)



Numerical Operations

- import numpy as np

  ```
  lst = [2,3, 7.9, 3.3, 6.9, 0.11, 10.3, 12.9]
  v = np.array(lst)
  v = v + 2
  print(v)
  ```

- import numpy as np

  A = np.array([[11, 12, 13], [21, 22, 23], [31, 32, 33]])

  B = np.ones((3,3))

  print("Adding to arrays: ")

  print(A + B)

Broadcasting

- import numpy as np

  A = np.array([[11, 12, 13], [21, 22, 23], [31, 32, 33]])

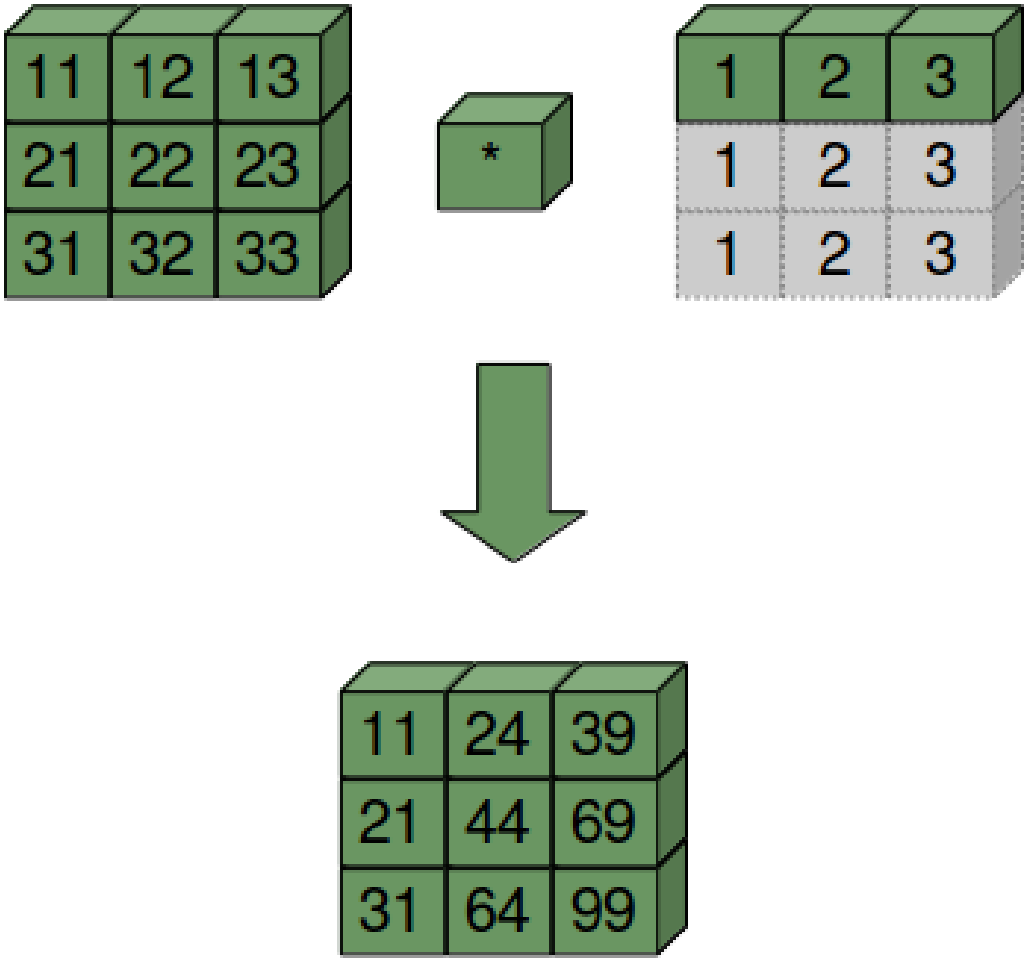
  B = np.array([1, 2, 3])

  print("Multiplication with broadcasting: ")

  print(A * B)

  print("... and now addition with broadcasting: ")

  print(A + B)

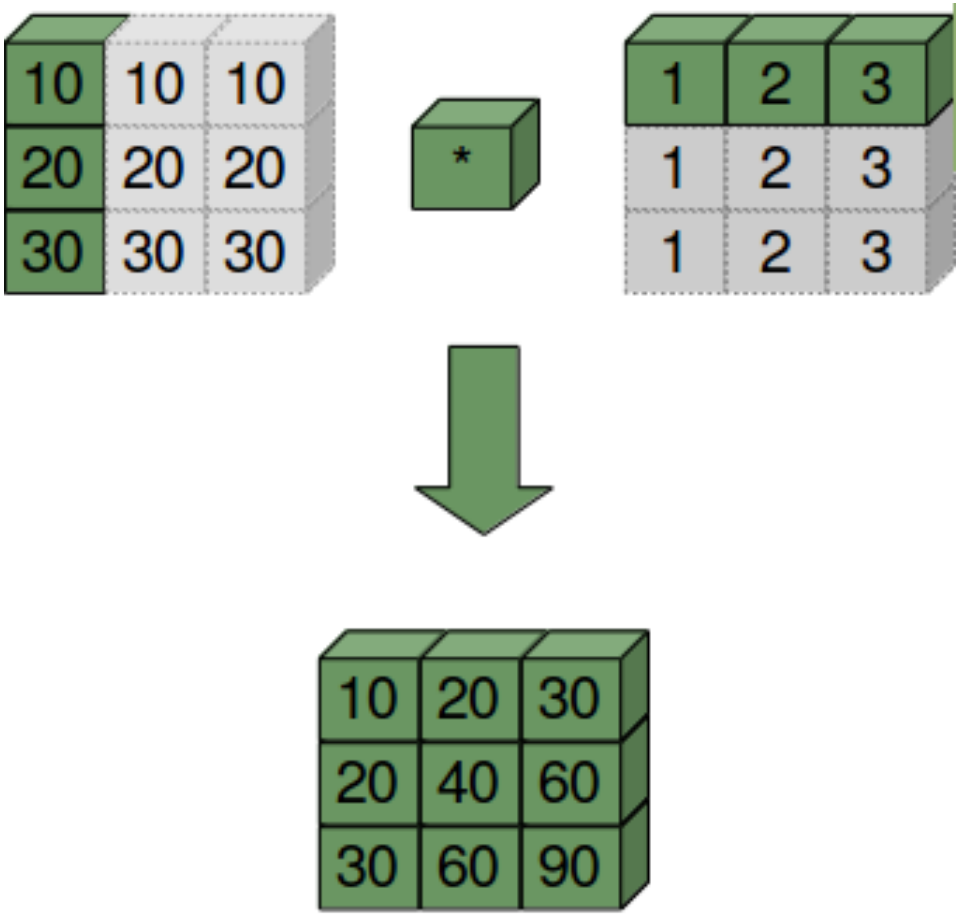

11 12 13
21 22 23
31 32 33
*
1 2 3
1 2 3
1 2 3
11 24 39
21 44 69
31 64 99

- A = np.array([10, 20, 30])

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

  A[:, np.newaxis]

  A[:, np.newaxis] * B


10 10 10
20 20 20
30 30 30
*
1 2 3
1 2 3
1 2 3
10 20 30
20 40 60
30 60 90