In [1]: import numpy as np

#Creating an Array in Numpy

a = np.array([1, 4, 5, 8,9,11,13,15, 19,23], float)

b = np.array([11, 14, 15, 18,19,21,23,25, 29,33], int)

c = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]], in

t)

print(a)

print(type(a))

print(b)

print(type(b))

#slicing

print("Slicing:")

print(a[:2])

print(a[3:8])

print(a[0])

print(a[3:])

print(a[-5])

print(a[-5:])

#Datatype of an Array

print("Datatype of an Array:")

print(a.dtype)

print(c.dtype)

#Shaping of Array

print("Shaping an Array:")

print(a.shape)

x,y=c.shape

print(x)

print(c.shape)

print(len(a))

print(a.dtype)

print(c.dtype)

#Creation of array using command

print("Creation of array using command:")

d= np.array(range(10), int)

print(d)

e = c.reshape((4, 3))

print(e)

print("creating Array with all 1")

print(np.ones((6,7)))

print("Converting all elements of array to ZERO")

print(np.zeros\_like(a))

print("Converting all elements of array to ONE")

print(np.ones\_like(a))

print("Creating Identity Matrix")

print(np.identity(4))

#Conveting Array into MAACHINE READABLE Format

print("Conveting Array into MAACHINE READABLE Format")

f = a.tostring()

print(f)

print(np.fromstring(f))

#Build and Reshaping ar array

print("Build and Reshaping ar array")

g = np.array(range(6), float).reshape((2, 3))

print(g)

print(c.flatten())

#Concatenation of Array

print("Concatenation of Array")

i = np.array([[1, 2], [3, 4]], float)

j = np.array([[5, 6], [7,8]], float)

print("Concat Array:\n ")

print(np.concatenate((i, j)))

print("Concat Array about axis 0(Vertical):\n ")

print(np.concatenate(((i, j)),axis=0))

print("Concat Array about axis 1(Horizontal):\n ")

print(np.concatenate(((i, j)),axis=1))

#Array Mathematics

print("Addition of Array")

print(a+b)

print("Subtraction of Array")

print(a-b)

print("Multiplication of Array")

print(a\*b)

print("Division of Array")

print(a/b)

print("Modulus of Array")

print(a%b)

print("Power(Exponential) of Array")

print(a\*\*b)

print("Square Root of array")

print(np.sqrt(a))

k=np.array([1.5, 4.34, 5.12, 8.55,9.5,11.99,13.01,15.6, 19.34,2

3.67], float)

print("Floor value of array")

print(np.floor(k))

print("Ceil value of array")

print(np.ceil(k))

print("Printing Value of pi")

print(np.pi)

print("Printing Value of e")

print(np.e)

#Basic Array Operation

k=np.array([1, 4, 5, 8, 9])

print("Sum of array elements")

print(k.sum())

print("Product of array elements")

print(k.prod())

print("Mean of array elements")

print(k.mean())

print("Variance of array elements")

print(k.var())

print("Standard Deviation of array elements")

print(k.std())

print("Min of array elements")

print(k.min())

print("Max of array elements")

print(k.max())

print("Argmin of array elements")

print(k.argmin())

print("Argmax of array elements")

print(k.argmax())

print("Array in Sorted order")

print(sorted(k))

print("Unique elements in array")

print(np.unique(k))

print("Printing diagonal elements of an array")

print(c.diagonal())

print("Printing the comparison of elements")

print(a>b) # Applicable also for < == <= =>

print("Printing dot multiplication")

print(np.dot(b, a))

print("Print Determinant")

m=np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

print(np.linalg.det(m))

print("Printing Eigen Values")

vals, vecs = np.linalg.eig(m)

print("1. Eigen Values")

print(vals)

print("2. Eigen Vector" )

print(vecs)

print("Printing Inverse of an Array")

print(np.linalg.inv(m))

print("Showing Polynomial Coefficient")

print(np.poly([-1, 1, 1, 10])) #Shows Polynomial

coefficients

print("Showing Roots of an equation")

print(np.roots([1, 4, -2, 3])) #Shows Rots of an

equation with given coefficients

#Random numbers

print("Printing Random numbers with seed")

print(np.random.seed(293423))

print("Print random numbers")

print(np.random.rand(5))

print("Printing Random numbers of 2-D Array")

print(np.random.rand(2,3))

[ 1. 4. 5. 8. 9. 11. 13. 15. 19. 23.]

<class 'numpy.ndarray'>

[11 14 15 18 19 21 23 25 29 33]

<class 'numpy.ndarray'>

Slicing:

[1. 4.]

[ 8. 9. 11. 13. 15.]

1.0

[ 8. 9. 11. 13. 15. 19. 23.]

11.0

[11. 13. 15. 19. 23.]

Datatype of an Array:

float64

int32

Shaping an Array:

(10,)

4

(4, 3)

10

float64

int32

Creation of array using command:

[0 1 2 3 4 5 6 7 8 9]

[[ 1 2 3]

[ 4 5 6]

[ 7 8 9]

[10 11 12]]

creating Array with all 1

[[1. 1. 1. 1. 1. 1. 1.]

[1. 1. 1. 1. 1. 1. 1.]

[1. 1. 1. 1. 1. 1. 1.]

[1. 1. 1. 1. 1. 1. 1.]

[1. 1. 1. 1. 1. 1. 1.]

[1. 1. 1. 1. 1. 1. 1.]]

Converting all elements of array to ZERO

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

Converting all elements of array to ONE

[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]

Creating Identity Matrix

[[1. 0. 0. 0.]

[0. 1. 0. 0.]

[0. 0. 1. 0.]

[0. 0. 0. 1.]]

Conveting Array into MAACHINE READABLE Format

b'\x00\x00\x00\x00\x00\x00\xf0?\x00\x00\x00\x00\x00\x00\x10@

\x00\x00\x00\x00\x00\x00\x14@\x00\x00\x00\x00\x00\x00 @\x00

\x00\x00\x00\x00\x00"@\x00\x00\x00\x00\x00\x00&@\x00\x00\x00

\x00\x00\x00\*@\x00\x00\x00\x00\x00\x00.@\x00\x00\x00\x00\x00

\x003@\x00\x00\x00\x00\x00\x007@'

[ 1. 4. 5. 8. 9. 11. 13. 15. 19. 23.]

Build and Reshaping ar array

[[0. 1. 2.]

[3. 4. 5.]]

[ 1 2 3 4 5 6 7 8 9 10 11 12]

Concatenation of Array

Concat Array:

[[1. 2.]

[3. 4.]

[5. 6.]

[7. 8.]]

Concat Array about axis 0(Vertical):

[[1. 2.]

[3. 4.]

[5. 6.]

[7. 8.]]

Concat Array about axis 1(Horizontal):

[[1. 2. 5. 6.]

[3. 4. 7. 8.]]

Addition of Array

[12. 18. 20. 26. 28. 32. 36. 40. 48. 56.]

Subtraction of Array

[-10. -10. -10. -10. -10. -10. -10. -10. -10. -10.]

Multiplication of Array

[ 11. 56. 75. 144. 171. 231. 299. 375. 551. 759.]

Division of Array

[0.09090909 0.28571429 0.33333333 0.44444444 0.47368421 0.52

380952

0.56521739 0.6 0.65517241 0.6969697 ]

Modulus of Array

[ 1. 4. 5. 8. 9. 11. 13. 15. 19. 23.]

Power(Exponential) of Array

[1.00000000e+00 2.68435456e+08 3.05175781e+10 1.80143985e+16

1.35085172e+18 7.40024994e+21 4.17539054e+25 2.52511683e+29

1.21298220e+37 8.65004942e+44]

Square Root of array

[1. 2. 2.23606798 2.82842712 3. 3.31

662479

3.60555128 3.87298335 4.35889894 4.79583152]

Floor value of array

[ 1. 4. 5. 8. 9. 11. 13. 15. 19. 23.]

Ceil value of array

[ 2. 5. 6. 9. 10. 12. 14. 16. 20. 24.]

Printing Value of pi

3.141592653589793

Printing Value of e

2.718281828459045

Sum of array elements

27

Product of array elements

1440

Mean of array elements

5.4

Variance of array elements

8.24

Standard Deviation of array elements

2.870540018881465

Min of array elements

1

Max of array elements

9

Argmin of array elements

0

Argmax of array elements

4

Array in Sorted order

[1, 4, 5, 8, 9]

Unique elements in array

[1 4 5 8 9]

Printing diagonal elements of an array

[1 5 9]

Printing the comparison of elements

[False False False False False False False False False Fals

e]

Printing dot multiplication

C:\Users\Subhadeep Chakrabort\Anaconda3\lib\site-packages\ip

ykernel\_launcher.py:55: DeprecationWarning: The binary mode

of fromstring is deprecated, as it behaves surprisingly on u

nicode inputs. Use frombuffer instead

2672.0

Print Determinant

-9.51619735392994e-16

Printing Eigen Values

1. Eigen Values

[ 1.61168440e+01 -1.11684397e+00 -9.75918483e-16]

2. Eigen Vector

[[-0.23197069 -0.78583024 0.40824829]

[-0.52532209 -0.08675134 -0.81649658]

[-0.8186735 0.61232756 0.40824829]]

Printing Inverse of an Array

[[ 3.15251974e+15 -6.30503948e+15 3.15251974e+15]

[-6.30503948e+15 1.26100790e+16 -6.30503948e+15]

[ 3.15251974e+15 -6.30503948e+15 3.15251974e+15]]

Showing Polynomial Coefficient

[ 1. -11. 9. 11. -10.]

Showing Roots of an equation

[-4.5797401 +0.j 0.28987005+0.75566815j 0.28987005

-0.75566815j]

Printing Random numbers with seed

None

Print random numbers

[0.33677247 0.52693437 0.79529578 0.78867702 0.02147624]

Printing Random numbers of 2-D Array

[[0.84612516 0.0704939 0.1526965 ]

[0.77831701 0.80821151 0.82198398]]