Numpy

In [1]:
```
import numpy as np
```

0D Array or Scalar

In [ ]:
```
x1=np.array(55)
print(x1)
print(type(x1))
```

In [7]:
```
x2=np.array([1,2,3,4,5,6,7,8,9])
print(x2)
print(type(x2))
```

```
[1 2 3 4 5 6 7 8 9]
<class 'numpy.ndarray'>
```

In [15]:
```
print(np.__version__)
```

```
1.26.4
```

In [19]:
```
x3=np.array([[1,2,3],[4,5,6]])
print(x3)
print(x3.ndim)
print(x3.shape)
```

```
[[1 2 3]
 [4 5 6]]
2
(2, 3)
```

3D Array

In [29]:
```
x4=np.array([[[2,3],[4,5]],[[6,7],[8,9]]])
print(x4)
print(x4.ndim)
print(x4.shape)
```

```
[[[2 3]
  [4 5]]

 [[6 7]
  [8 9]]]
3
(2, 2, 2)
```

Reshaping of Array

In [39]:
```
x5=np.array([1,2,3,4,5,6,7,8])
print(x5)
print(x5.ndim)
print(x5.shape)
print("_"*30)
```

```
x6=x5.reshape(2,4)
print(x6)
print(x6.ndim)
print(x6.shape)
print("_"*30)
```

```
[1 2 3 4 5 6 7 8]
1
(8,)
------------------------------
[[1 2 3 4]
 [5 6 7 8]]
2
(2, 4)
------------------------------
```

1D to 3D Array

In [41]:
```
x7=np.array([1,2,3,4,5,6,7,8,9,10,11,12])
print(x7)
print(x7.ndim)
print(x7.shape)
print("_"*30)
x8=x7.reshape(2,3,2)
print(x8)
print(x8.ndim)
print(x8.shape)
print("_"*30)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12]
1
(12,)
------------------------------
[[[ 1  2]
  [ 3  4]
  [ 5  6]]

 [[ 7  8]
  [ 9 10]
  [11 12]]]
3
(2, 3, 2)
------------------------------
```

Convert 1D to 4D OR 5D

In [57]:
```
x9=np.array([1,2,3,4,5,6,7,8,9,10,11,12])
print(x9)
print(x9.ndim)
print(x9.shape)
print("_"*30)
x10=x9.reshape(1,6,1,2)
print(x10)
print(x10.ndim)
print(x10.shape)
print("_"*30)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12]
1
(12,)
```

```
[[[[ 1  2]]

  [[ 3  4]]

  [[ 5  6]]

  [[ 7  8]]

  [[ 9 10]]

  [[11 12]]]]
4
(1, 6, 1, 2)
```

In [61]:
```
A=np.array([[4,3],[-5,9]])
B=np.array([20,96])
print(A,B)
print(A.ndim)
print(B.ndim)
sol=np.linalg.solve(A,B)
print(sol)
```

```
[[ 4  3]
 [-5  9]] [20 96]
2
1
[-2.11764706  9.49019608]
```

THREE variable EQ solution

In [78]:
```
A=np.array([[12,14,-10],[20,-15,11],[15,-4,5]])
B=np.array([20,21,12])
print("A=",A,"B=",B)
print(A.ndim)
print(B.ndim)
sol=np.linalg.solve(A,B)
print(sol)
```

```
A= [[ 12  14 -10]
 [ 20 -15  11]
 [ 15  -4   5]] B= [20 21 12]
2
1
[ 1.31798246 -1.89254386 -3.06798246]
```

In [92]:
```
from matplotlib. image import imread
hi=imread("C://Users//Purva//Downloads//CAT.jpg")
print(hi)
print(type(hi))
print(hi.ndim)
print(hi.shape)
```

```
[[[192 146 112]
  [192 146 112]
  [192 146 112]
  ...
  [137  90  60]
  [137  90  60]
  [139  92  62]]

 [[192 146 112]
  [192 146 112]
  [192 146 112]
  ...
  [136  89  59]
  [137  90  60]
  [138  91  61]]

 [[193 147 113]
  [192 146 112]
  [192 146 112]
  ...
  [136  89  59]
  [136  89  59]
  [137  90  60]]

 ...

 [[243 199 164]
  [243 199 164]
  [243 199 164]
  ...
  [170 116  80]
  [170 116  80]
  [171 117  81]]

 [[243 199 164]
  [243 199 164]
  [243 199 164]
  ...
  [168 114  78]
  [168 114  76]
  [169 115  77]]

 [[243 199 164]
  [243 199 164]
  [243 199 164]
  ...
  [167 113  77]
  [167 113  75]
  [168 114  76]]]
<class 'numpy.ndarray'>
3
(736, 736, 3)
```

In [84]:
```
from matplotlib. image import imread
hi=imread("C://Users//Purva//Downloads//dog.jpeg")
print(hi)
print(type(hi))
print(hi.ndim)
print(hi.shape)
```

```
[[[245 245 245]
  [245 245 245]
  [244 244 244]
  ...
  [176 176 176]
  [176 176 176]
  [175 175 175]]

 [[245 245 245]
  [245 245 245]
  [244 244 244]
  ...
  [178 178 178]
  [178 178 178]
  [178 178 178]]

 [[245 245 245]
  [245 245 245]
  [244 244 244]
  ...
  [182 182 182]
  [181 181 181]
  [181 181 181]]

 ...

 [[137 137 137]
  [138 138 138]
  [138 138 138]
  ...
  [106 106 106]
  [105 105 105]
  [105 105 105]]

 [[136 136 136]
  [137 137 137]
  [137 137 137]
  ...
  [104 104 104]
  [104 104 104]
  [103 103 103]]

 [[136 136 136]
  [136 136 136]
  [136 136 136]
  ...
  [103 103 103]
  [103 103 103]
  [102 102 102]]]
<class 'numpy.ndarray'>
3
(2252, 3416, 3)
```

Matrix Solving

In [99]:
```
test=np.array([[1,2j,8],[1,-3,4],[7,6,9]])
print(test)
print("_"*30)
print(np.linalg.matrix_rank(test))
print("_"*30)
```

```
print(np.trace(test))
print("_"*30)
print(np.linalg.det(test))
print("_"*30)
print(np.linalg.inv(test))
print("_"*30)
```

```
[[ 1.+0.j  0.+2.j  8.+0.j]
 [ 1.+0.j -3.+0.j  4.+0.j]
 [ 7.+0.j  6.+0.j  9.+0.j]]
______________________________
3
______________________________
(7+0j)
______________________________
(165+38j)
______________________________
[[-0.29352262+0.06759915j  0.25239806-0.16721895j  0.14873208+0.0142314j ]
 [ 0.10935156-0.025184j   -0.27050124+0.06229725j  0.02302138-0.00530189j]
 [ 0.15539433-0.03578778j -0.01597544+0.08852768j -0.01991698-0.00753427j]]
______________________________
```

In [101…
```
np.linalg.matrix_power(test,4)
```

Out[101…
```
array([[10573. +732.j,  7104.+1240.j, 18960.+1776.j],
       [ 4772. +104.j,  3485. +476.j,  8288. +272.j],
       [17504. +516.j, 10604.+2484.j, 33153.+1056.j]])
```

Eigen VALUE

In [112…
```
test1=np.array([[1,2,4],[8,7,4],[1,2,3]])
print(test1)
c,d=np.linalg.eigh(test1)
print(c)
print(d)
```

```
[[1 2 4]
 [8 7 4]
 [1 2 3]]
[-4.5578466   2.52522275 13.03262385]
[[ 0.81791781  0.15874408  0.55300161]
 [-0.57368144  0.15220468  0.80481261]
 [ 0.04358981 -0.97551733  0.21555945]]
```

In [114…
```
np.transpose(test1)
```

Out[114…
```
array([[1, 8, 1],
       [2, 7, 2],
       [4, 4, 3]])
```

In [116…
```
test1
```

Out[116…
```
array([[1, 2, 4],
       [8, 7, 4],
       [1, 2, 3]])
```

MEAN,MODE,MEDIAN

In [121…

```
from scipy import stats
speed=np.array([1,2,3,4,5,6,7,8,9,11,22,56,78,49,22,45,30])
m=np.mean(speed)
print(m)
m1=np.median(speed)
print(m1)
m2=stats.mode(speed)
print(m2)
```

```
21.058823529411764
9.0
ModeResult(mode=22, count=2)
```

In [ ]: