In [3]:
```
import numpy as np
```

In [4]:
```
np.__version__
```

Out[4]:
```
'2.1.3'
```

Creating list

In [11]:
```
my_list=[0,1,2,3,4,5]
my_list
```

Out[11]:
```
[0, 1, 2, 3, 4, 5]
```

In [12]:
```
type(my_list)
```

Out[12]:
```
list
```

In [13]:
```
#!pip install numpy
```

In [14]:
```
arr = np.array(my_list)
```

arr

In [15]:
```
np.# we are the important function in numpy
```

```
  Cell In[15], line 1
    np.# we are the important function in numpy
       ^
SyntaxError: invalid syntax
```

In [16]:
```
np.arange(10)
```

Out[16]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

In [17]:
```
np.arange(5.0)
```

Out[17]:
```
array([0., 1., 2., 3., 4.])
```

In [18]:
```
np.arange(9)
```

Out[18]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8])
```

In [19]:
```
np.arange(0,5)
```

Out[19]:
```
array([0, 1, 2, 3, 4])
```

In [20]:
```
np.arange(20)
```

Out[20]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19])
```

In [21]:
```
np.arange(10,20)
```

Out[21]:
```
array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19])
```

In [22]:
```
np.arange(-20,10)
```

Out[22]:
```
array([-20, -19, -18, -17, -16, -15, -14, -13, -12, -11, -10,  -9,  -8,
        -7,  -6,  -5,  -4,  -3,  -2,  -1,   0,   1,   2,   3,   4,   5,
         6,   7,   8,   9])
```

In [23]:
```
np.arange(16,10)
```

Out[23]:
```
array([], dtype=int64)
```

In [24]:
```
np.arange(-16,10)
```

Out[24]:
```
array([-16, -15, -14, -13, -12, -11, -10,  -9,  -8,  -7,  -6,  -5,  -4,
        -3,  -2,  -1,   0,   1,   2,   3,   4,   5,   6,   7,   8,   9])
```

In [25]:
```
np.arange(20,10)
```

Out[25]:
```
array([], dtype=int64)
```

In [26]:
```
np.arange(30,20) # 1st arg always be < then 2nd arg
```

Out[26]:
```
array([], dtype=int64)
```

In [27]:
```
ar=np.arange(-30,20)
ar
```

Out[27]:
```
array([-30, -29, -28, -27, -26, -25, -24, -23, -22, -21, -20, -19, -18,
       -17, -16, -15, -14, -13, -12, -11, -10,  -9,  -8,  -7,  -6,  -5,
        -4,  -3,  -2,  -1,   0,   1,   2,   3,   4,   5,   6,   7,   8,
         9,  10,  11,  12,  13,  14,  15,  16,  17,  18,  19])
```

In [28]:
```
np.arange(10,10)
```

Out[28]:
```
array([], dtype=int64)
```

In [29]:
```
np.arange()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[29], line 1
----> 1 np.arange()

TypeError: arange() requires stop to be specified.
```

In [30]:
```
np.arange(10,30,5)
```

Out[30]:
```
array([10, 15, 20, 25])
```

In [31]:
```
np.arange(0,10,3)
```

Out[31]:
```
array([0, 3, 6, 9])
```

In [32]:
```
np.arange(10,30,5,8)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[32], line 1
----> 1 np.arange(10,30,5,8)

TypeError: Cannot interpret '8' as a data type
```

In [33]:
```
np.zeros(10) #parameter tunning
```

Out[33]:
```
array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

In [34]:
```
np.zeros(3,dtype=int)  # hyperparameter tunning
```

Out[34]:
```
array([0, 0, 0])
```

In [35]:
```
np.zeros((2,2),dtype=int)
```

Out[35]:
```
array([[0, 0],
       [0, 0]])
```

In [36]:
```
zero=np.zeros([2,2])
print(zero)
print(type(zero))
```

```
[[0. 0.]
 [0. 0.]]
<class 'numpy.ndarray'>
```

In [37]:
```
zero=np.zeros([2,2])
print(zero)
print('####')
print(type(zero))
```

```
[[0. 0.]
 [0. 0.]]
####
<class 'numpy.ndarray'>
```

In [38]:
```
np.zeros((2,10))
```

Out[38]:
```
array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

In [39]:
```
np.zeros((2,2))
```

Out[39]:
```
array([[0., 0.],
       [0., 0.]])
```

In [40]:
```
np.zeros((3,3))
```

Out[40]:
```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

In [41]:
```
np.zeros((10,30))
```

Out[41]:
```
array([[0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
        0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

In [42]:
```
np.zeros((10,10),dtype=int)
```

Out[42]:
```
array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
```

In [43]:
```
np.ones(3)
```

Out[43]:
```
array([1., 1., 1.])
```

In [44]:
```
np.ones((3,3))
```

Out[44]:
```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

In [45]:
```
np.ones((3,3),dtype=int)
```

Out[45]:
```
array([[1, 1, 1],
       [1, 1, 1],
       [1, 1, 1]])
```

In [46]:
```
np.ones(4,dtype=int)
```

Out[46]:
```
array([1, 1, 1, 1])
```

In [47]:
```
np.ones((2,3))
```

Out[47]:
```
array([[1., 1., 1.],
       [1., 1., 1.]])
```

In [48]:
```
np.three(2,3)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[48], line 1
----> 1 np.three(2,3)

File ~\anaconda3\Lib\site-packages\numpy\__init__.py:414, in __getattr__(attr)
    411     import numpy.char as char
    412     return char.chararray
--> 414 raise AttributeError("module {!r} has no attribute "
    415                       "{!r}".format(__name__, attr))

AttributeError: module 'numpy' has no attribute 'three'
```

In [ ]:
```
np.ones(2)
```

In [49]:
```
rand(3,2)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[49], line 1
----> 1 rand(3,2)

NameError: name 'rand' is not defined
```

In [50]:
```
random.rand(3,2)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[50], line 1
----> 1 random.rand(3,2)

NameError: name 'random' is not defined
```

In [51]:
```
np.random.rand(3)
```

Out[51]:
```
array([0.9748585 , 0.6447187 , 0.26953885])
```

In [52]:
```
np.rand(4)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[52], line 1
----> 1 np.rand(4)

File ~\anaconda3\Lib\site-packages\numpy\__init__.py:414, in __getattr__(attr)
    411     import numpy.char as char
    412     return char.chararray
--> 414 raise AttributeError("module {!r} has no attribute "
    415                       "{!r}".format(__name__, attr))

AttributeError: module 'numpy' has no attribute 'rand'
```

In [53]:
```
np.random.rand(3,5)
```

Out[53]:
```
array([[0.81311961, 0.80854835, 0.03249898, 0.06935251, 0.09597165],
       [0.15693392, 0.36703656, 0.70296439, 0.88908059, 0.58403868],
       [0.25276992, 0.7644598 , 0.80261683, 0.90290789, 0.264563  ]])
```

In [54]:
```
np.random.randint(4,6)
```

Out[54]:
```
4
```

In [55]:
```
np.random.randint(10,20)
```

Out[55]:
```
12
```

In [56]:
```
np.random.randint(10,20,5) #2nd argument is exclusive
```

Out[56]:
```
array([16, 18, 14, 14, 14], dtype=int32)
```

In [57]:
```
np.random.randint(2,20,5) #2nd argument is exclusive
```

Out[57]:
```
array([18,  4, 12, 18, 18], dtype=int32)
```

In [58]:
```
np.random.randint(0,1)
```

Out[58]:
```
0
```

In [59]:
```
np.random.randint(3,5)
```

Out[59]: 4

In [60]:
```
np.random.randint(6,7)
```

Out[60]: 6

In [61]:
```
np.random.randint(1,6,4)
```

Out[61]: array([1, 1, 1, 5], dtype=int32)

In [62]:
```
np.random.rand(3)
```

Out[62]: array([0.77924087, 0.45050472, 0.31138509])

In [63]:
```
np.random.randint(1)
```

Out[63]: 0

In [64]:
```
np.random.randint(8,9)
```

Out[64]: 8

In [65]:
```
np.random.randint(7,9)
```

Out[65]: 7

In [66]:
```
np.random.randint(0,5)
```

Out[66]: 3

In [67]:
```
np.random.randint(30,20,10)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[67], line 1
----> 1 np.random.randint(30,20,10)

File numpy\\random\\mtrand.pyx:796, in numpy.random.mtrand.RandomState.randint()

File numpy\\random\\_bounded_integers.pyx:1425, in numpy.random._bounded_integers._rand_int32()

ValueError: low >= high
```

In [68]:
```
np.random.randint(-30,20,10)
```

Out[68]:
```
array([ -1,  19, -27, -13, -24,  14,  -2,   5,  19,  -4], dtype=int32)
```

In [69]:
```
np.random.randint(20,30,10)
```

Out[69]:
```
array([26, 25, 24, 22, 22, 21, 29, 27, 28, 21], dtype=int32)
```

In [70]:
```
np.random.randint(5,9) # GET THE VALUE <1=5
```

Out[70]:
```
7
```

In [71]:
```
np.random.randint(10,21,3)
```

Out[71]:
```
array([17, 16, 14], dtype=int32)
```

In [72]:
```
np.random.randint(10,40,(10,10)) # generate the element 10  -30vwith 4*4 mtri
```

Out[72]:
```
array([[15, 29, 21, 39, 36, 29, 11, 28, 23, 29],
       [13, 31, 33, 31, 24, 19, 23, 30, 39, 18],
       [19, 36, 14, 21, 26, 12, 27, 32, 10, 16],
       [19, 36, 27, 10, 32, 15, 23, 33, 35, 37],
       [27, 32, 20, 12, 32, 25, 23, 17, 11, 37],
       [36, 33, 37, 18, 12, 16, 29, 15, 24, 10],
       [28, 33, 19, 22, 15, 16, 29, 34, 27, 19],
       [27, 16, 19, 29, 13, 37, 34, 18, 13, 20],
       [36, 34, 16, 24, 15, 33, 37, 13, 25, 36],
       [26, 11, 21, 12, 34, 10, 11, 24, 20, 37]], dtype=int32)
```

In [73]:
```
np.random.randint(1,100,(12,12)) # generate the element 10  -30vwith 4*4 mtri
```

Out[73]:
```
array([[30,  5, 26, 96, 25, 22,  1, 89,  5, 16, 31, 39],
       [96, 45, 35, 88, 28, 72, 95, 39, 78, 43, 44, 60],
       [14, 59, 67, 32, 43, 36, 26, 11, 26, 57, 81, 13],
       [37, 33, 55, 93, 89, 51, 32, 16, 47, 20, 87, 90],
       [82, 78, 11, 50, 96, 91, 33,  3,  8, 40, 37, 93],
       [89,  7, 75, 26, 98, 33, 84, 93, 12,  6, 97, 34],
       [16, 95, 44, 90, 32, 95, 79, 71, 37, 20, 64, 67],
       [27, 55,  6, 32, 93, 50, 66, 30, 93, 21, 74,  3],
       [ 7, 79,  5, 24, 60, 50,  6, 51,  5, 58, 25, 29],
       [59, 36, 86, 25, 59,  3, 23, 71, 11, 72, 54, 45],
       [15, 71, 63, 27, 75,  9, 81, 30, 68, 10, 33, 78],
       [21, 87, 82, 35, 35, 69, 61, 15,  8, 74, 16, 82]], dtype=int32)
```

In [74]:
```
arr
```

Out[74]:
```
array([0, 1, 2, 3, 4, 5])
```

In [75]:
```
np.arange(1,13)
```

Out[75]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

In [76]:
```
np.arange(1,13).reshape(3,4)
```

Out[76]:
```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

In [77]:
```
np.arange(1,13).reshape(4,3)
```

Out[77]:
```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7,  8,  9],
       [10, 11, 12]])
```

In [78]:
```
np.arange(1,13).reshape(2,6)
```

Out[78]:
```
array([[ 1,  2,  3,  4,  5,  6],
       [ 7,  8,  9, 10, 11, 12]])
```

In [79]:
```
np.arange(1,13).reshape(6,2)
```

Out[79]:
```
array([[ 1,  2],
       [ 3,  4],
       [ 5,  6],
       [ 7,  8],
       [ 9, 10],
       [11, 12]])
```

In [80]:
```
np.arange(1,13).reshape(12,1)
```

Out[80]:
```
array([[ 1],
       [ 2],
       [ 3],
       [ 4],
       [ 5],
       [ 6],
       [ 7],
       [ 8],
       [ 9],
       [10],
       [11],
       [12]])
```

In [81]:
```
np.arange(1,13).reshape(1,12)
```

Out[81]:
```
array([[ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12]])
```

In [82]:
```
np.arange(1,21).reshape(5,4)
```

Out[82]:
```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12],
       [13, 14, 15, 16],
       [17, 18, 19, 20]])
```

In [83]:
```
np.arange(1,13).reshape(5,4)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[83], line 1
----> 1 np.arange(1,13).reshape(5,4)

ValueError: cannot reshape array of size 12 into shape (5,4)
```

In [84]:
```
np.arange(1,21).reshape(5,4)
```

Out[84]:
```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12],
       [13, 14, 15, 16],
       [17, 18, 19, 20]])
```

In [85]:
```
np.arange(1,21).reshape(4,5)
```

Out[85]:
```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15],
       [16, 17, 18, 19, 20]])
```

slicing in Matrix

In [86]:
```
b = np.random.randint(10,20,(5,4))
b
```

Out[86]:
```
array([[16, 14, 13, 11],
       [14, 15, 18, 10],
       [19, 11, 11, 17],
       [11, 18, 11, 11],
       [13, 13, 17, 18]], dtype=int32)
```

In [87]:
```
b[:]
```

Out[87]:
```
array([[16, 14, 13, 11],
       [14, 15, 18, 10],
       [19, 11, 11, 17],
       [11, 18, 11, 11],
       [13, 13, 17, 18]], dtype=int32)
```

In [88]:
```
b[0]
```

Out[88]:
```
array([16, 14, 13, 11], dtype=int32)
```

In [89]:
```
b[4]
```

Out[89]:
```
array([13, 13, 17, 18], dtype=int32)
```

In [90]:
```
b[3]
```

Out[90]:
```
array([11, 18, 11, 11], dtype=int32)
```

In [91]:
```
b[-3]
```

Out[91]:
```
array([19, 11, 11, 17], dtype=int32)
```

In [92]:
```
b[-4]
```

Out[92]:
```
array([14, 15, 18, 10], dtype=int32)
```

In [93]:
```
b[1:3]
```

Out[93]:
```
array([[14, 15, 18, 10],
       [19, 11, 11, 17]], dtype=int32)
```

In [94]:
```
b[2:5]
```

Out[94]:
```
array([[19, 11, 11, 17],
       [11, 18, 11, 11],
       [13, 13, 17, 18]], dtype=int32)
```

In [95]:
```
b[4:3]
```

Out[95]:
```
array([], shape=(0, 4), dtype=int32)
```

In [96]:
```
type(b)
```

Out[96]:
```
numpy.ndarray
```

In [97]:
```
b[1:3]
```

Out[97]:
```
array([[14, 15, 18, 10],
       [19, 11, 11, 17]], dtype=int32)
```

In [98]:
```
b[-4]
```

Out[98]:
```
array([14, 15, 18, 10], dtype=int32)
```

In [99]:
```
b[3]
```

Out[99]:
```
array([11, 18, 11, 11], dtype=int32)
```

In [100…
```
b[1,3]
```

Out[100…
```
np.int32(10)
```

In [101…
```
b[0,3]
```

Out[101…
```
np.int32(11)
```

In [102…
```
b[1,2]
```

Out[102…
```
np.int32(18)
```

In [103…
```
b[1:2]
```

Out[103…
```
array([[14, 15, 18, 10]], dtype=int32)
```

In [104…
```
b[1,-1]
```

Out[104…
```
np.int32(10)
```

In [105…
```
b[2:3]
```

Out[105…
```
array([[19, 11, 11, 17]], dtype=int32)
```

In [106…
```
b[0:2]
```

Out[106…
```
array([[16, 14, 13, 11],
       [14, 15, 18, 10]], dtype=int32)
```

In [107…
```
b[0:-2]
```

Out[107…
```
array([[16, 14, 13, 11],
       [14, 15, 18, 10],
       [19, 11, 11, 17]], dtype=int32)
```

In [108…
```
b
```

Out[108…
```
array([[16, 14, 13, 11],
       [14, 15, 18, 10],
       [19, 11, 11, 17],
       [11, 18, 11, 11],
       [13, 13, 17, 18]], dtype=int32)
```

In [109…
```
b[-5,3]
```

Out[109…
```
np.int32(11)
```

In [110…
```
b[-5,-3]
```

Out[110…
```
np.int32(14)
```

Operation

In [112…
```
import numpy as np
```

In [114…
```
#create an array from a list
a=np.array([1,2,3])
print("array a:",a)
```

```
array a: [1 2 3]
```

In [115…
```
# create an array with evenly spaced values
b=np.arange(0,10,2) # values from 0 to 10 with step 2
print("array b:",b)
```

```
array b: [0 2 4 6 8]
```

In [116…
```
# create an array with evenly spaced values
b=np.arange(0,10,3) # values from 0 to 10 with step 2
print("array b:",b)
```

```
array b: [0 3 6 9]
```

In [117…
```
# create an array with evenly spaced values
b=np.arange(0,10,4) # values from 0 to 10 with step 2
print("array b:",b)
```

```
array b: [0 4 8]
```

In [118…
```
# create an array filled with zeros
d=np.zeros((2,3)) # 2x3 array of zeros
print("array d:\n",d)
```

```
array d:
 [[0. 0. 0.]
 [0. 0. 0.]]
```

In [121…
```
# create an array filled with ones
e=np.ones((3,2)) # 3x2 array of ones
print("array e:\n",e)
```

```
array e:
 [[1. 1.]
 [1. 1.]
 [1. 1.]]
```

In [122…
```
#create an identity matrix
f=np.eye(4) #4x4 identity matrix
print("identity matrix f:\n",f)
```

```
identity matrix f:
 [[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
```

In [ ]:
```
# Array Manupulation
```

In [123…
```
# reshape an array
a1=np.array([1,2,3])
reshaped=np.reshape(a1,(1,3)) # Reshape to 1x3
print("Reshaped array:",reshaped)
```

```
Reshaped array: [[1 2 3]]
```

In [124…
```
# reshape an array
a1=np.array([1,2,3])
reshaped=np.reshape(a1,(3,1)) # Reshape to 1x3
print("Reshaped array:",reshaped)
```

```
Reshaped array: [[1]
 [2]
 [3]]
```

In [126…
```
# reshape an array
a1=np.array([1,2,3])
reshaped=np.reshape(a1,(3,1)) # Reshape to 1x3
print("Reshaped array:",reshaped)
```

```
Reshaped array: [[1]
 [2]
 [3]]
```

In [129…
```
# Flatten an array
f1=np.array([[1,2,],[3,4]])
flattened=np.ravel(f1) # flatten to 1d array
print("Flattened array:",flattened)
```

```
Flattened array: [1 2 3 4]
```

In [130…
```python
# Flatten an array
f1=np.array([[1,2,],[3,4],[4,5],[7,8]])
flattened=np.ravel(f1) # flatten to 1d array
print("Flattened array:",flattened)
```

```
Flattened array: [1 2 3 4 4 5 7 8]
```

In [131…
```python
# Transpose an array
e1=np.array([[1,2],[3,4]])
transposed=np.transpose(e1) # transposed the array
print("Transposed array:\n",transposed)
```

```
Transposed array:
 [[1 3]
 [2 4]]
```

In [132…
```python
# stack arrays vertically
a2=np.array([1,2])
b2=np.array([3,4])
stacked=np.vstack([a2,b2]) # stack a and b vertically
print("stacked arrays:\n",stacked)
```

```
stacked arrays:
 [[1 2]
 [3 4]]
```

python program to generate OTP

In [1]:
```python
import random

def generate_otp(length=4):
    """Generate a numeric OTP of a specified length."""
    digits = '02451357'
    otp = ''.join(random.choice(digits) for _ in range(length))
    return otp

# Example usage
otp_length = 6 # You can change this to any length you prefer
```

```
otp = generate_otp(otp_length)
print(f"Your OTP is: {otp}")
```

```
Your OTP is: 534052
```

In [2]:
```
def wish():
    print('Dabang delhi champion')
wish()

def wish():
    print('Dabang delhi champion')
wish()

def wish():
    print('Dabang delhi champion')
wish()
```

```
Dabang delhi champion
Dabang delhi champion
Dabang delhi champion
```

In [6]:
```
def wish():
    print('Dabang delhi champion')
wish()

wish()

wish()
```

```
Dabang delhi champion
Dabang delhi champion
Dabang delhi champion
```

In [7]:
```
list1=['a','b','g',2,5]
print(list1.pop)
```

```
<built-in method pop of list object at 0x000002845A35AD40>
```

In [8]:
```
x = [4, 5, 6]
y = x.copy()
x.append(4)
print(x)
```

```
[4, 5, 6, 4]
```

In [9]:
```
import numpy as np #array
```

In [10]:
```
pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\santo\anaconda3\lib\site-packages (2.1.3)Note: you may need to restart the ker
nel to use updated packages.
```

In [11]:
```
pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\users\santo\anaconda3\lib\site-packages (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: numpy>=1.23 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (2.1.3)
Requirement already satisfied: packaging>=20.0 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\santo\anaconda3\lib\site-packages (from matplotlib) (2.9.0.post
0)
Requirement already satisfied: six>=1.5 in c:\users\santo\anaconda3\lib\site-packages (from python-dateutil>=2.7->matplotlib)
(1.17.0)
Note: you may need to restart the kernel to use updated packages.
```

In [12]:
```
pip install pillow
```

```
Requirement already satisfied: pillow in c:\users\santo\anaconda3\lib\site-packages (11.1.0)Note: you may need to restart the k
ernel to use updated packages.
```

In [13]:
```
image=(r"C:\Users\santo\OneDrive\Desktop\ikkk.jpg")
```

In [14]:
```
image
```

Out[14]:
```
'C:\\Users\\santo\\OneDrive\\Desktop\\ikkk.jpg'
```

In [19]:
```
import numpy as np
```

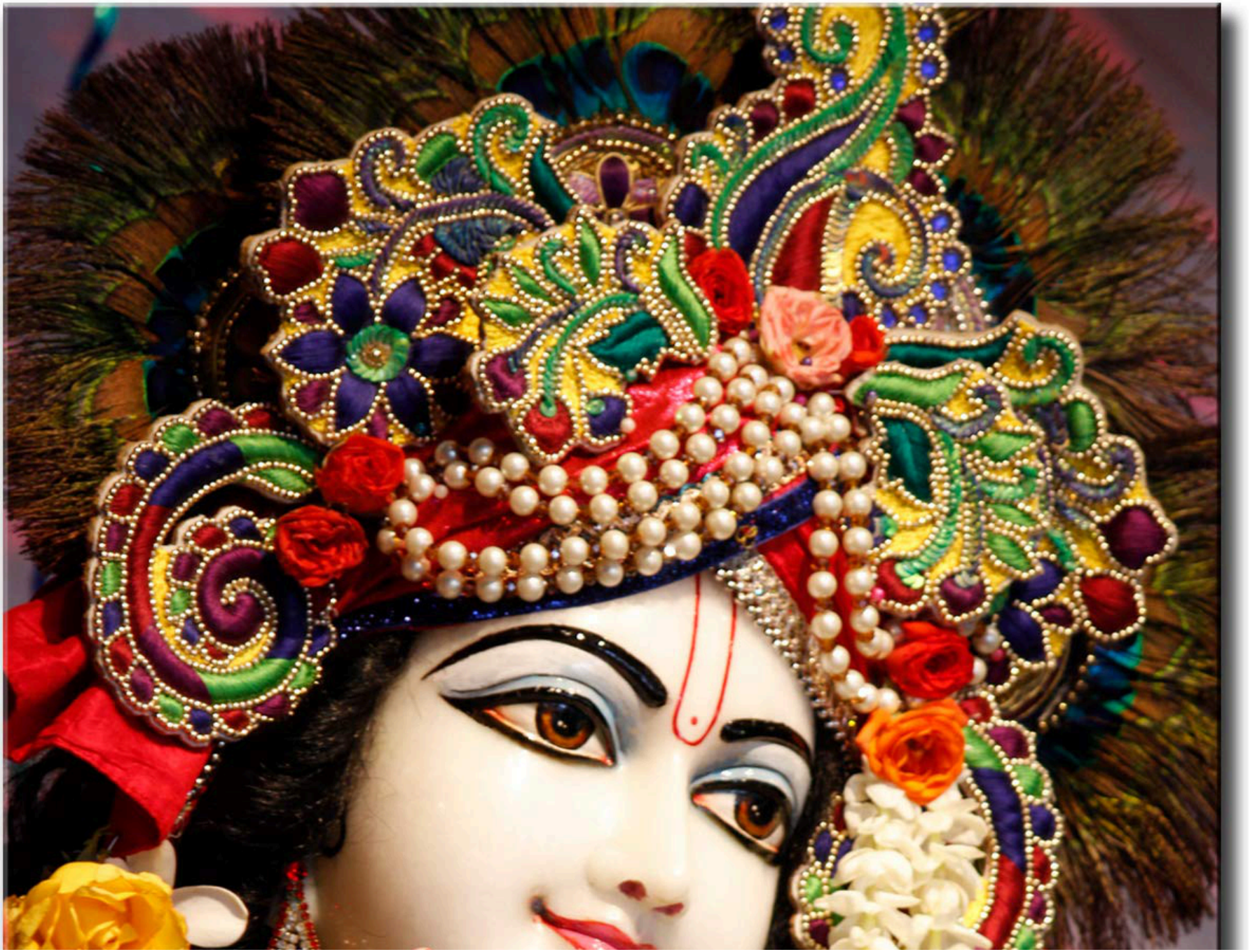
In [20]:
```
import matplotlib.pyplot as plt
```

In [21]:
```
from PIL import Image
```

In [22]:
```
feature_image= Image.open(r"C:\Users\santo\OneDrive\Desktop\ikkk.jpg")
```

In [23]:
```
feature_image
```

Out[23]:



In [27]:
```
myimage= Image.open(r"D:\Pictures\.thumbnails\1000516873.jpg")
```

In [28]:
```
myimage
```

Out[28]:

In [29]:
```
print(type(feature_image))
print(type(myimage))
```

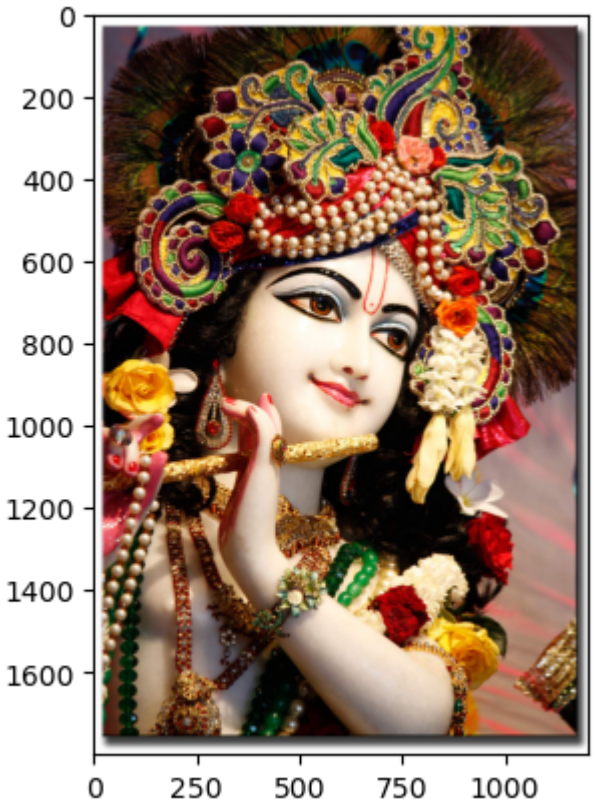
```
<class 'PIL.JpegImagePlugin.JpegImageFile'>
<class 'PIL.JpegImagePlugin.JpegImageFile'>
```

In [30]:
```
fea_arr= np.asarray(feature_image)
fea_arr
```

Out[30]:
```
array([[[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       ...,

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
```

```
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]]], dtype=uint8)
```

In [31]:
```
plt.imshow(fea_arr)
plt.show()
```



In [ ]: