```python
import numpy as np
print(np.__version__) # type of version and to check whether numpy install or not.
```

```
1.26.4
```

```python
a=np.array([1,2,3,4,5]) # one dimension al array
```

```python
a
```

```
array([ 1,  2,  3, 55, 65])
```

```python
a[2] # accessing the elemnts using list
```

```
3
```

```python
a[3]=7 # The array is mutable
```

```python
a
```

```
array([1, 2, 3, 7, 5])
```

```python
a[:4]  # Python slice notation can be used for indexing
```

```
array([1, 2, 3, 7])
```

```python
a
```

```
array([ 1,  2,  3, 55, 65])
```

```python
a.shape # shape tell about no.of columns and no.of rows
```

```
(5,)
```

```python
c=a[3:]
```

```python
c
```

```
array([40,  5])
```

```python
c[0]=55
```

```python
c[1]=65
```

```python
a
```

```
array([ 1,  2,  3, 55, 65])
```

```python
b=np.array([[1,2,3,4,5],
           [5,6,7,8,9]]) # 2D array
#The shape must be “rectangular”, not “jagged”; e.g., each row of a two-dimensional array must have the same number of columns.
```

```
b
```

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

```
b.shape
```

```
(2, 5)
```

Array attributes

```
arr=np.array([1,2,3,4,5])
```

```
arr
```

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

```
arr.ndim  # he number of dimensions of an array is contained in the
ndim attribute.
```

```
1
```

```
ab=np.array([[1,23,45],[75,99,63]])
```

```
ab
```

```
array([[ 1, 23, 45],
       [75, 99, 63]])
```

```
ab.ndim
```

```
2
```

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

```
three
```

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

       [[7, 8, 9],
        [5, 6, 9]]])
```

```
three.ndim # three dimensional
```

```
3
```

```
four=np.array([
    [[[1,2,3],[3,4,5]]],

              [[[4,5,7],[8,9,2]]]
              ])
four.ndim # Four dimensions
4
```

Shape (pecify the number of elements along each dimension.)

```
arr.shape
(5,)
ab.shape
(2, 3)
three.shape
(2, 2, 3)
len(arr.shape)==arr.ndim
True
len(ab.shape)==ab.ndim
True
len(three)==three.ndim
False
```

size

```
a.size
5
ab.size
6
three.size
12
```

dtype

```
a.dtype

dtype('int32')

arr.dtype

dtype('int32')

four.dtype

dtype('int32')
```