Practical 1: Working with Numpy.

1. **importing Numpy**

   import numpy as np

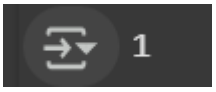
2. **Numpy arrays**

   a = np.array([1,2,3]) # 1D Array
   b = np.array([[1,2],[3,4]]) # 2D Array
   c = np.array([[[1,2],[3,4],[5,6]]]) # 3D Array

3. **Array properties**

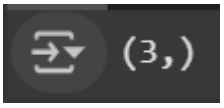
   a.ndim #Dimensions

Output:



a.shape # shape

Output:



a.size #total no of elements

Output:

a.dtype # datatype

Output:

```
dtype('int64')
```

4. Array creation methods

np.zeros((2,3)) # Array of zeroes

Output:

```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

np.ones((3,3)) # Array of Ones

Output:

```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

np.eye(3) # Identity Matrix

Output:

```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

np.full((2,2),7) #Constant value

Output:

```
array([[7, 7],
       [7, 7]])
```

np.arange(0,10,2) # Array with step

Output:

```
array([0, 2, 4, 6, 8])
```

np.linspace(0,1,5) #five valuesn from 0 to 1

Output:

```
array([0.  , 0.25, 0.5 , 0.75, 1.  ])
```

np.random.rand(2,3) # Random floats from 0 to 1

Output:

```
array([[0.66092618, 0.36695722, 0.84240356],
       [0.2449551 , 0.37839553, 0.88424946]])
```

5. **Array indexing and slicing**

a = np.array([10,20,30,40,50])
a[1] #single index

Output:

```
np.int64(20)
```

a[1:4] #slicing

Output:

```
array([20, 30, 40])
```

```
b = np.array([[1,2,3],[4,5,6]])
b[0,1]                  #Elements at row 0, col 1
```

Output:

```
np.int64(2)
```

```
b[:,1]                  # all rows, column 1
```

Output:

```
array([2, 5])
```

6. Array operations

```
b = np.array([4,6,8,10]);
c = np.array([1,3,5,7]);
d = b-c;
print(d);
print(np.subtract(c,b));
e = b+c;
print(e);
f=b*c;
print(f);
g = b/c;
print(g);
print(b==c);
e=b**c;
print(e);
f = b//c;
print(f);
```

Output:

```
[3 3 3 3]
[-3 -3 -3 -3]
[ 5  9 13 17]
[ 4 18 40 70]
[4.         2.         1.6        1.42857143]
[False False False False]
[       4      216    32768 10000000]
[4 2 1 1]
```

7. **Broadcasting**

```
a = np.array([[1],[2],[3]])
b = np.array([10,20,30])
a + b
```

Output:

```
array([[11, 21, 31],
       [12, 22, 32],
       [13, 23, 33]])
```

8. **Tuple and List**

```
a =(1,);
print(type(a));

b =[1,];
print(type(b));
```

Output:

```
<class 'tuple'>
<class 'list'>
```

9. Using numpy functions

```
a = np.array([3,6,1,5,2,7,4])
print(a.sum());
print(a.min());
print(a.max());
print(a.mean());
print(np.std(a));
print(np.argmax(a));
print(np.argmin(a));
print(np.sort(a));
```

Output:

```
28
1
7
4.0
2.0
5
2
[1 2 3 4 5 6 7]
```

10. Reshaping and Flattening

```
a = np.array([[1,2],[3,4],[5,6]])
print(a.reshape(2,3));       #Reshape
print(a.flatten());          #Flatten
```

Output:

```
[[1 2 3]
 [4 5 6]]
[1 2 3 4 5 6]
```

11. Stacking and Splitting

```
a = np.array([4,6,8,10]);
b = np.array([1,3,5,7]);
print(np.vstack((a,b)));      #Vertical Stack
print(np.hstack(a));          # Horizental Stack
print(np.split(a,2));         #Split into equal parts
```

Output:

```
[[ 4  6  8 10]
 [ 1  3  5  7]]
[ 4  6  8 10]
[array([4, 6]), array([ 8, 10])]
```

12. Where , Any,All

```
a = np.array([4,6,8,10]);
print(np.where(a > 2));      #index where condition is true
print(np.any(a > 3));        #True if any element > 3
print(np.all(a < 5));        # True if all elements < 5
```

Output:

```
(array([0, 1, 2, 3]),)
True
False
```