

## Python (Practical list 4)

**1.Learn to Create 1-D, 2-D and 3-D array using numpy in python.**

**Program:**

```
import numpy as np
```

```
print("\n1-D array")
```

```
a = np.array([1,2,3,4,5,6])
```

```
print(a)
```

```
print("\n1-D array with arange function to interval ")
```

```
a1 = np.arange(2,12,3)
```

```
print(a1)
```

```
print("\n1-D array with linspace ")
```

```
a2 = np.linspace(1,10,4)
```

```
print(a2)
```

```
print("\n2-D array")
```

```
b = np.array([[1,2,3],[4,5,6]])
```

```
print(b)
```

```
print("\n2-D array with ones element ")
```

```
b1 = np.ones((2,2))  
print(b1)
```

```
print("\n2-D array with random array ")  
b2 = np.random.rand(2,2)  
print(b2)
```

```
print("\n2-D array with identity")  
b3 = np.identity(2)  
print(b3)
```

```
print("\n3-D array")  
c = np.array([[[1,2,3],  
               [4,5,6]],  
              [[7,8,9],  
               [10,11,12]]])  
print(c)
```

```
print("\n3-D array with zeros ")  
c1 = np.zeros((2,3,4))  
print(c1)
```

```
Thorny - C:\Users\ARJUN\OneDrive\Desktop\worpy_1.py @ 24:38
File Edit View Run Device Tools Help
py_1.py py_3.py py_4.py py_5.py py_2.py
3 print("\n1-D array")
4 a = np.array([1,2,3,4,5,6])
5 print(a)
6
7 print("\n1-D array with arange function to interval ")
8
Shell
C:\>cd C:\Users\ARJUN\OneDrive\Desktop\worpy_1.py
>>> %Run py_1.py

1-D array
[1 2 3 4 5 6]

1-D array with arange function to interval
[ 2  5  8 11]

1-D array with linspace
[ 1.  4.  7. 10.]

2-D array
[[1 2 3]
 [4 5 6]]

2-D array with ones element
[[1. 1.]
 [1. 1.]]

2-D array with random array
[[0.50190281 0.661847  ]
 [0.05574326 0.31675475]]

2-D array with identity
[[1. 0.]
 [0. 1.]]

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

3-D array with zeros
[[[0. 0. 0.]
  [0. 0. 0.]
  [0. 0. 0.]]
 [[0. 0. 0.]
  [0. 0. 0.]
  [0. 0. 0.]]]

>>>
```

**2.Perform mathematical operations on 2-D arrays like addition, multiplication, subtraction and division using a single integer(b = a+5). Perform the same operations using two arrays as operands(c = a\*b).**

**Program:**

```
import numpy as np
```

```
b = np.array([[1,2,3],[4,5,6]])
```

```
print("2-D array of ..\n",b)
```

```
print("Addition of array..\n",b+5)
```

```
print("Multiplication of array..\n",b*5)
```

```
print("Subtraction of array..\n",b-5)
```

```
print("Division of array..\n",b / 5)
```

```

print(" Two element using operation ")
a = np.array([[10,20,30],[40,50,60]])
print("2-D array of a",a)
print("Addition of array..\n",a+b)
print("Multiplication of array..\n",a*b)
print("Subtraction of array..\n",a-b)
print("Division of array..\n",a/b)

```

The screenshot shows a Thonny Python IDE window. The top pane contains a Python script with the following code:

```

1
2
3 b = np.array([[1,2,3],[4,5,6]])
4
5 print("2-D array of ..\n",b)
6 print("Addition of array..\n",b+5)
7 print("Multiplication of array..\n",b*5)
8 print("Subtraction of array..\n",b-5)
9 print("Division of array..\n",b / 5)
10
11
12 print(" Two element using operation ")
13 a = np.array([[10,20,30],[40,50,60]])
14 print("2-D array of a",a)
15

```

The bottom pane shows the output of the script:

```

In [0]: 0.0 0.0
[0. 0. 0. 0.]
[0. 0. 0. 0.]]

>>> %Run py_2.py

2-D array of ..
[[1 2 3]
 [4 5 6]]
Addition of array..
[[ 6  7  8]
 [ 9 10 11]]
Multiplication of array..
[[ 5 10 15]
 [20 25 30]]
Subtraction of array..
[[-4 -3 -2]
 [-1  0  1]]
Division of array..
[[0.2 0.4 0.6]
 [0.8 1. 1.2]]
Two element using operation
2-D array of a [[10 20 30]
 [40 50 60]]
Addition of array..
[[11 22 33]
 [44 55 66]]
Multiplication of array..
[[10 40 90]
 [160 250 360]]
Subtraction of array..
[[ 9 18 27]
 [36 45 54]]
Division of array..
[[10. 10. 10.]
 [10. 10. 10.]]

>>>

```

### 3.Perform different relational operations like >,>=,<,<=,== and !=on two arrays

#### Program:

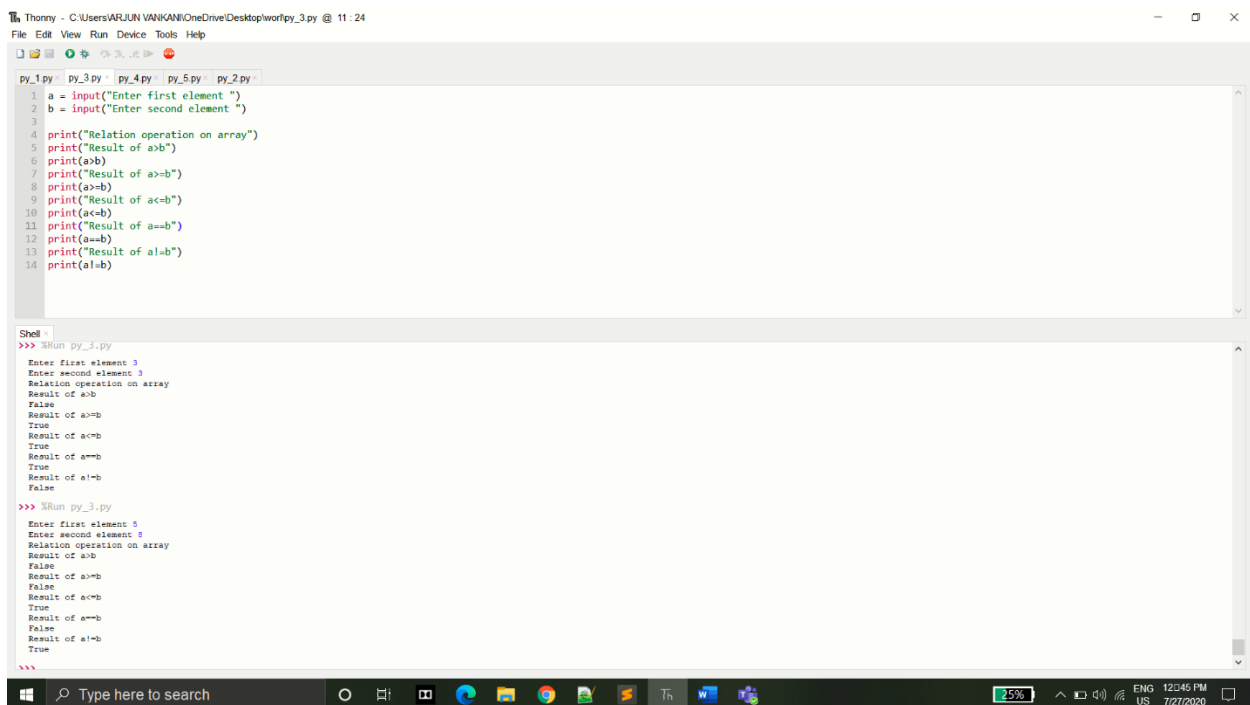
```

a = input("Enter first element ")
b = input("Enter second element ")

print("Relation operation on array")

```

```
print("Result of a>b")
print(a>b)
print("Result of a>=b")
print(a>=b)
print("Result of a<=b")
print(a<=b)
print("Result of a==b")
print(a==b)
print("Result of a!=b")
print(a!=b)
```



The screenshot shows a Python IDE window titled 'Thorny - C:\Users\ARJUN VANKANI\OneDrive\Desktop\work\py\_3.py @ 11:24'. The editor contains a script that takes two inputs, 'a' and 'b', and prints the results of various comparison operations. The script is as follows:

```
1 a = input("Enter first element ")
2 b = input("Enter second element ")
3
4 print("Relation operation on array")
5 print("Result of a>b")
6 print(a>b)
7 print("Result of a>=b")
8 print(a>=b)
9 print("Result of a<=b")
10 print(a<=b)
11 print("Result of a==b")
12 print(a==b)
13 print("Result of a!=b")
14 print(a!=b)
```

The IDE also shows the output of running the script twice. The first run uses inputs 3 and 3, and the second run uses inputs 0 and 0. The output for the first run is:

```
>>> %Run py_3.py
Enter first element 3
Enter second element 3
Relation operation on array
Result of a>b
False
Result of a>=b
True
Result of a<=b
True
Result of a==b
True
Result of a!=b
False
```

The output for the second run is:

```
>>> %Run py_3.py
Enter first element 0
Enter second element 0
Relation operation on array
Result of a>b
False
Result of a>=b
False
Result of a<=b
True
Result of a==b
True
Result of a!=b
False
```

## 4.Perform operations like view(), copy(), shape(), reshape(), flatten() on arrays

### Program:

```
import numpy as np
```

```
print("Copy array operation ")
```

```
a = np.array([1,2,3])
```

```
copy1 = a.copy()
```

```
a[2] = 10
```

```
print(a)
```

```
print(copy1)
```

```
print("View array operation ...It's one type of cloning")
```

```
view1 = a.view()
```

```
a[2] = 10
```

```
print(a)
```

```
print(view1)
```

```
print("Shape array operation ")
```

```
arr1 = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
```

```
print("Shape of array :",arr1.shape)
```

```
print("Reshape array operation ")
```

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

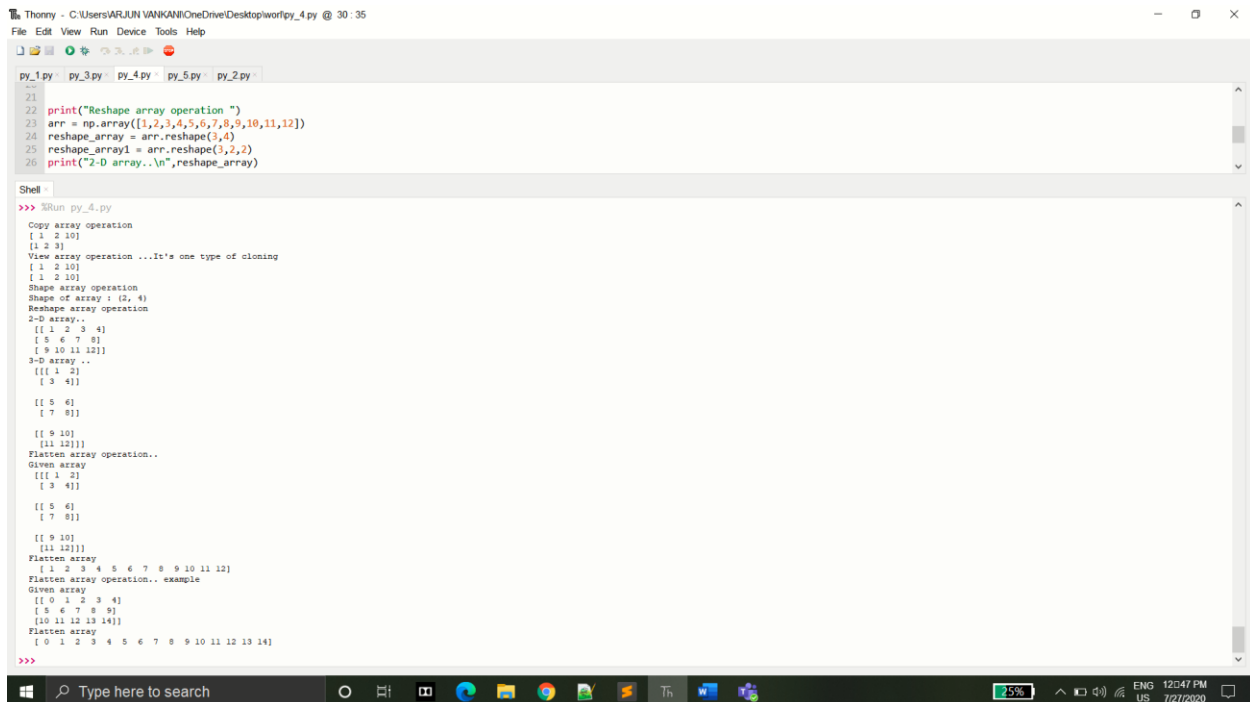
```
reshape_array = arr.reshape(3,4)
```

```
reshape_array1 = arr.reshape(3,2,2)
```

```
print("2-D array..\n",reshape_array)
print("3-D array ..\n",reshape_array1)
```

```
print("Flatten array operation..")
print("Given array\n",reshape_array1)
print("Flatten array\n ", reshape_array1.flatten())
```

```
print("Flatten array operation.. example ")
array = np.arange(15).reshape(3, 5)
print("Given array\n",array)
print("Flatten array \n", array.flatten())
```



The screenshot shows a Python IDE window titled "Thonny - C:\Users\ARJUN VANKAN\OneDrive\Desktop\work\py\_4.py @ 30:35". The code in the editor is as follows:

```
21
22 print("Reshape array operation ")
23 arr = np.array([1,2,3,4,5,6,7,8,9,10,11,12])
24 reshape_array = arr.reshape(3,4)
25 reshape_array1 = arr.reshape(3,2,2)
26 print("2-D array..\n",reshape_array)
```

The output in the Shell window is:

```
>>> Run py_4.py
Copy array operation
[[ 1  2 10]]
View array operation ...It's one type of cloning
[[ 1  2 10]]
Shape array operation
Shape of array : (2, 4)
Reshape array operation
2-D array..
[[[ 1  2  3  4]
  [ 5  6  7  8]
  [ 9 10 11 12]]]
3-D array ..
[[[ 1  2]
  [ 3  4]]
 [[ 5  6]
  [ 7  8]]
 [[ 9 10]
  [11 12]]]
Flatten array operation..
Given array
[[[ 1  2]
  [ 3  4]]
 [[ 5  6]
  [ 7  8]]
 [[ 9 10]
  [11 12]]]
Flatten array
[[ 1  2  3  4  5  6  7  8  9 10 11 12]]
Flatten array operation.. example
Given array
[[[ 0  1  2  3  4]
  [ 5  6  7  8  9]
  [10 11 12 13 14]]]
Flatten array
[[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]]
>>>
```

**5.Create matrix in numpy and perform operations like transpose, sort, addition, multiplication, find maximum and minimum element and find sum, average and product of all elements in the matrix.**

**Program:**

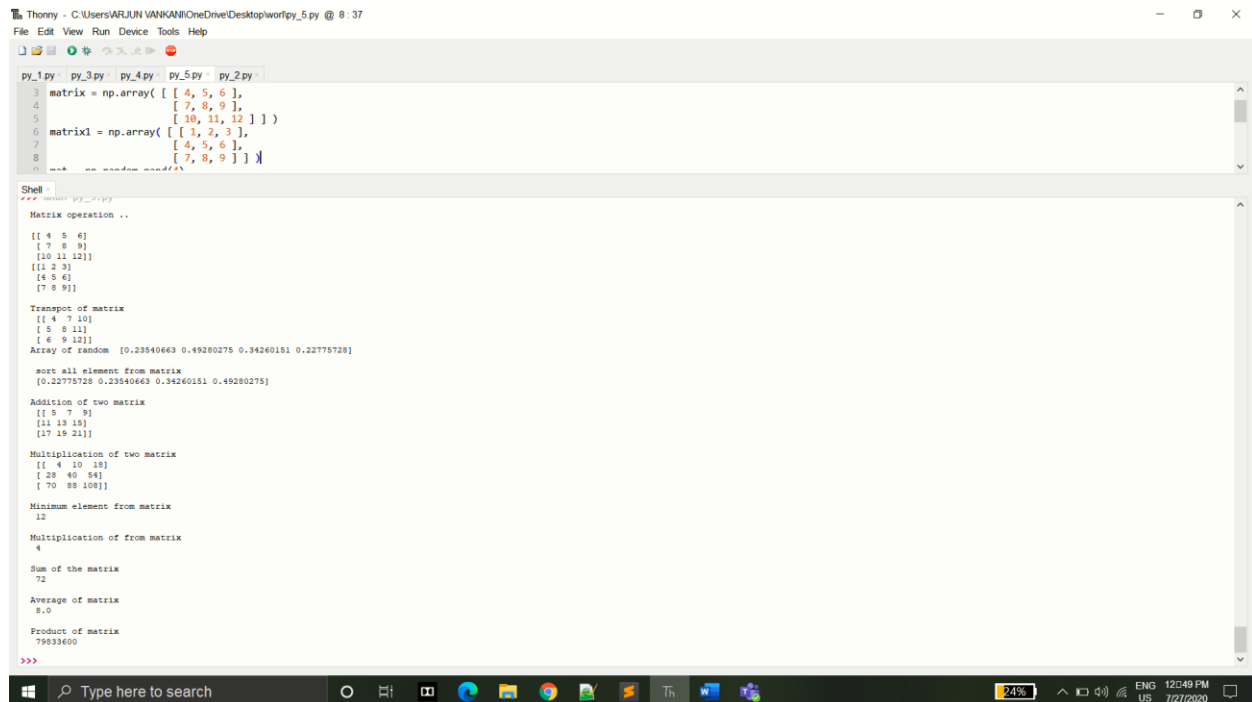
```
import numpy as np
print("Matrix operation ..\n")
matrix = np.array( [ [ 4, 5, 6 ],
                    [ 7, 8, 9 ],
                    [ 10, 11, 12 ] ] )
matrix1 = np.array( [ [ 1, 2, 3 ],
                    [ 4, 5, 6 ],
                    [ 7, 8, 9 ] ] )
mat = np.random.rand(4)
print(matrix)
print(matrix1)
print("\nTranspot of matrix\n",matrix.transpose())
print("Array of random ",mat)
mat.sort()
print("\n sort all element from matrix\n",mat)
print("\nAddition of two matrix \n",matrix+matrix1)
print("\nMultiplication of two matrix \n",matrix*matrix1)
print("\nMinimum element from matrix \n",matrix.min())
print("\nMultiplication of from matrix \n",matrix.max())
```



```
print("\nSum of the matrix \n",matrix.sum())
```

```
print("\nAverage of matrix \n",matrix.mean())
```

```
print("\nProduct of matrix \n",matrix.prod())
```



The screenshot shows a Python IDE window titled 'Thorny - C:\Users\ARJUN VANKAN\OneDrive\Desktop\wor\py\_5.py @ 8:37'. The code editor contains the following Python code:

```
3 matrix = np.array( [ [ 4, 5, 6 ],
4                   [ 7, 8, 9 ],
5                   [ 10, 11, 12 ] ] )
6 matrix1 = np.array( [ [ 1, 2, 3 ],
7                   [ 4, 5, 6 ],
8                   [ 7, 8, 9 ] ] )
```

The Shell window shows the output of the code:

```
>>> import numpy as np
Matrix operation ..

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

Transpose of matrix
[[ 4  7 10]
 [ 5  8 11]
 [ 6  9 12]]
Array of random [0.23540663 0.49280275 0.34260151 0.22775728]

sort all element from matrix
[0.22775728 0.23540663 0.34260151 0.49280275]

Addition of two matrix
[[ 5  7  9]
 [11 13 15]
 [17 19 21]]

Multiplication of two matrix
[[ 4 10 18]
 [28 40 54]
 [70 88 108]]

Minimum element from matrix
10

Multiplication of from matrix
4

Sum of the matrix
72

Average of matrix
8.0

Product of matrix
79833600
>>>
```

The Windows taskbar at the bottom shows the search bar, taskbar icons, and system tray with the date 1/21/2020 and time 12:49 PM.