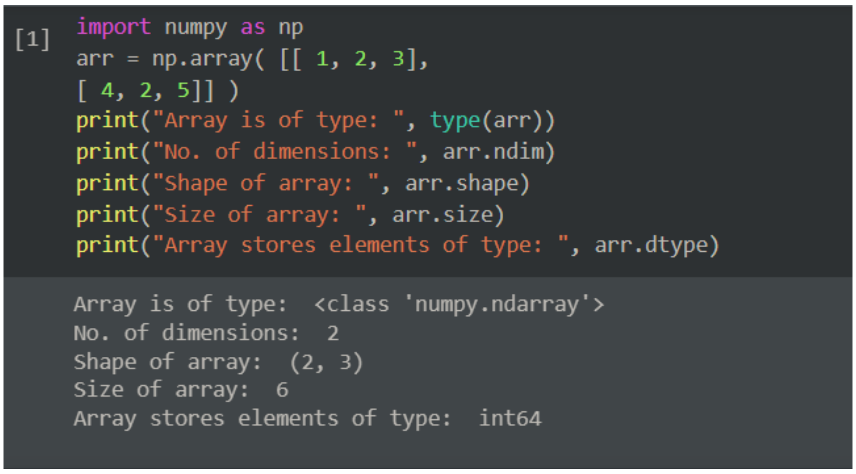
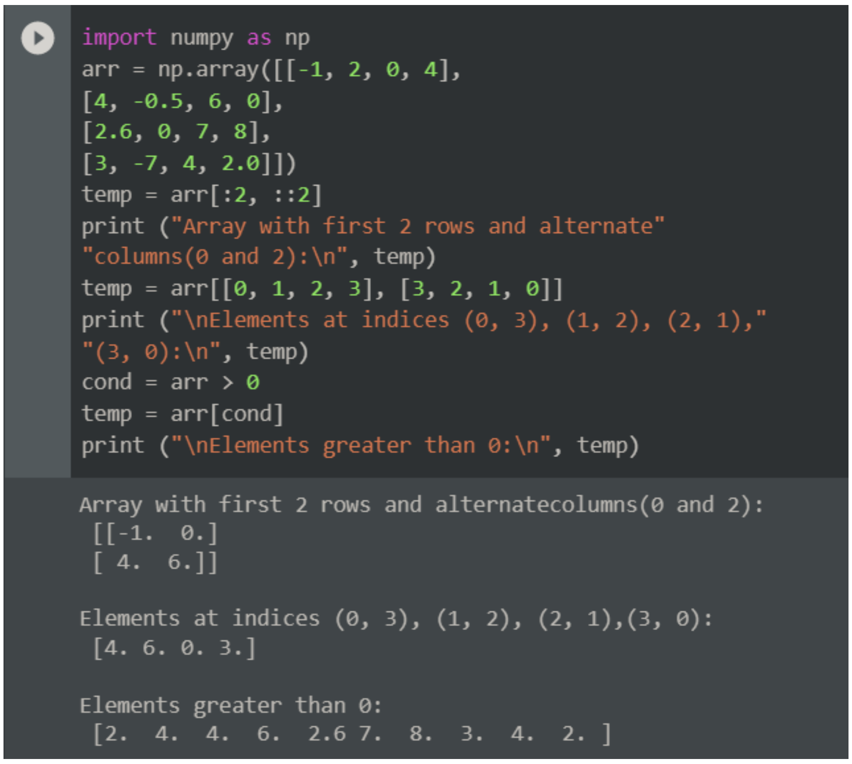
Assignment 4



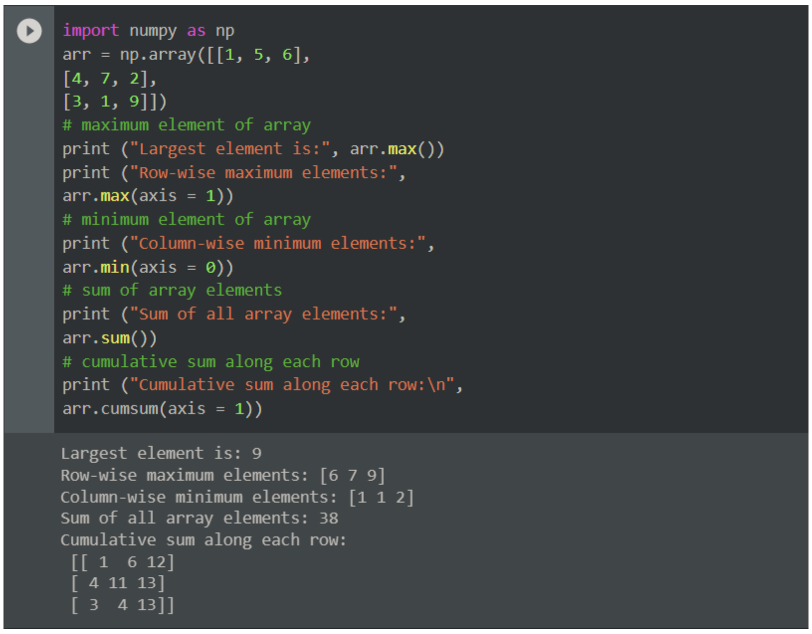
```
import numpy as np
arr = np.array( [[ 1, 2, 3],
[ 4, 2, 5]] )
print("Array is of type: ", type(arr))
print("No. of dimensions: ", arr.ndim)
print("Shape of array: ", arr.shape)
print("Size of array: ", arr.size)
print("Array stores elements of type: ", arr.dtype)
```

```
Array is of type:  <class 'numpy.ndarray'>
No. of dimensions:  2
Shape of array:  (2, 3)
Size of array:  6
Array stores elements of type:  int64
```

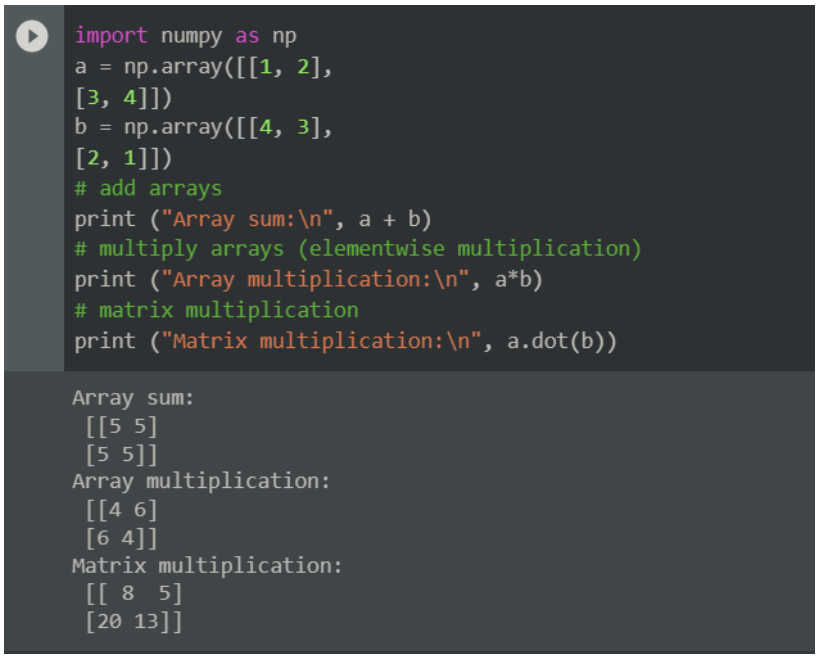


```
import numpy as np
arr = np.array([[-1, 2, 0, 4],
[4, -0.5, 6, 0],
[2.6, 0, 7, 8],
[3, -7, 4, 2.0]])
temp = arr[:2, ::2]
print ("Array with first 2 rows and alternate"
"columns(0 and 2):\n", temp)
temp = arr[[0, 1, 2, 3], [3, 2, 1, 0]]
print ("\nElements at indices (0, 3), (1, 2), (2, 1),"
"(3, 0):\n", temp)
cond = arr > 0
temp = arr[cond]
print ("\nElements greater than 0:\n", temp)
```

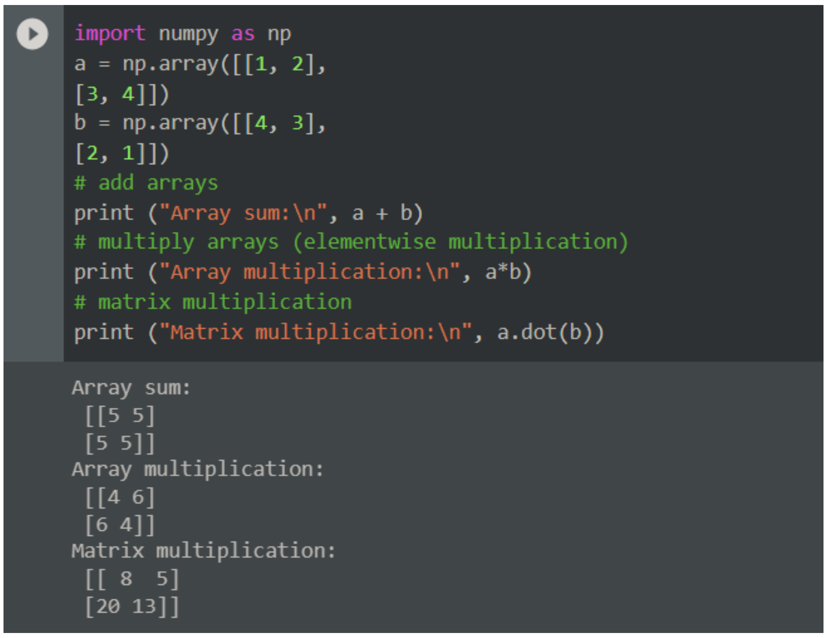
```
Array with first 2 rows and alternatecolumns(0 and 2):
 [[-1.  0.]
 [ 4.  6.]]

Elements at indices (0, 3), (1, 2), (2, 1),(3, 0):
 [4. 6. 0. 3.]

Elements greater than 0:
 [2.  4.  4.  6.  2.6 7.  8.  3.  4.  2. ]
```


```python
import numpy as np
arr = np.array([[1, 5, 6],
[4, 7, 2],
[3, 1, 9]])
# maximum element of array
print ("Largest element is:", arr.max())
print ("Row-wise maximum elements:",
arr.max(axis = 1))
# minimum element of array
print ("Column-wise minimum elements:",
arr.min(axis = 0))
# sum of array elements
print ("Sum of all array elements:",
arr.sum())
# cumulative sum along each row
print ("Cumulative sum along each row:\n",
arr.cumsum(axis = 1))
```

```
Largest element is: 9
Row-wise maximum elements: [6 7 9]
Column-wise minimum elements: [1 1 2]
Sum of all array elements: 38
Cumulative sum along each row:
 [[ 1  6 12]
 [ 4 11 13]
 [ 3  4 13]]
```

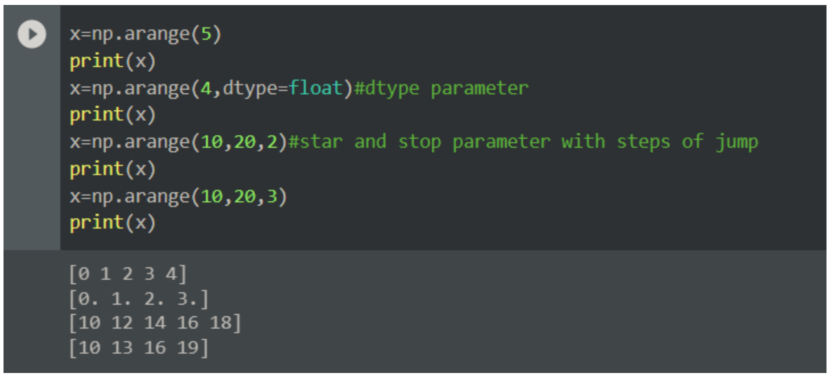

```python
import numpy as np
a = np.array([[1, 2],
[3, 4]])
b = np.array([[4, 3],
[2, 1]])
# add arrays
print ("Array sum:\n", a + b)
# multiply arrays (elementwise multiplication)
print ("Array multiplication:\n", a*b)
# matrix multiplication
print ("Matrix multiplication:\n", a.dot(b))
```

```
Array sum:
 [[5 5]
 [5 5]]
Array multiplication:
 [[4 6]
 [6 4]]
Matrix multiplication:
 [[ 8  5]
 [20 13]]
```



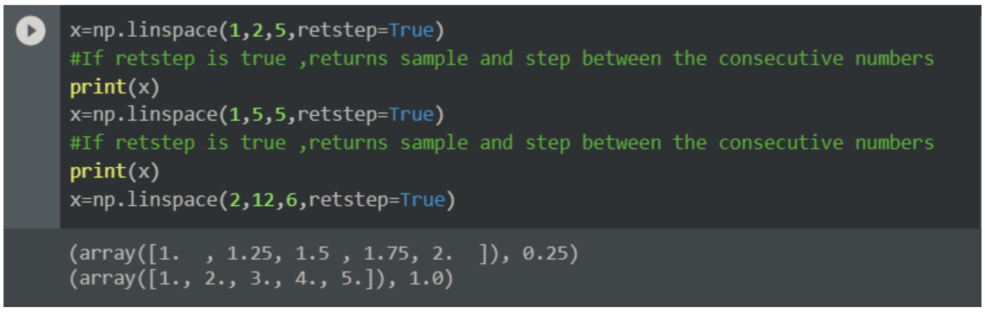
```
import numpy as np
a = np.array([[1, 2],
[3, 4]])
b = np.array([[4, 3],
[2, 1]])
# add arrays
print ("Array sum:\n", a + b)
# multiply arrays (elementwise multiplication)
print ("Array multiplication:\n", a*b)
# matrix multiplication
print ("Matrix multiplication:\n", a.dot(b))
```

```
Array sum:
 [[5 5]
 [5 5]]
Array multiplication:
 [[4 6]
 [6 4]]
Matrix multiplication:
 [[ 8  5]
 [20 13]]
```



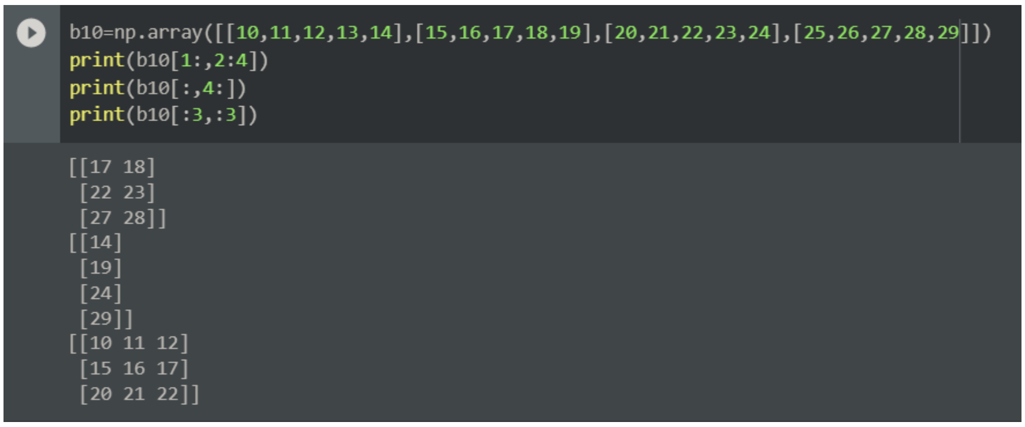
```
x=np.arange(5)
print(x)
x=np.arange(4,dtype=float)#dtype parameter
print(x)
x=np.arange(10,20,2)#star and stop parameter with steps of jump
print(x)
x=np.arange(10,20,3)
print(x)
```

```
[0 1 2 3 4]
[0. 1. 2. 3.]
[10 12 14 16 18]
[10 13 16 19]
```



```
x=np.linspace(1,2,5,retstep=True)
#If retstep is true ,returns sample and step between the consecutive numbers
print(x)
x=np.linspace(1,5,5,retstep=True)
#If retstep is true ,returns sample and step between the consecutive numbers
print(x)
x=np.linspace(2,12,6,retstep=True)
```

```
(array([1.  , 1.25, 1.5 , 1.75, 2.  ]), 0.25)
(array([1., 2., 3., 4., 5.]), 1.0)
```



```
b10=np.array([[10,11,12,13,14],[15,16,17,18,19],[20,21,22,23,24],[25,26,27,28,29]])
print(b10[1:,2:4])
print(b10[:,4:])
print(b10[:3,:3])
```

```
[[17 18]
 [22 23]
 [27 28]]
[[14]
 [19]
 [24]
 [29]]
[[10 11 12]
 [15 16 17]
 [20 21 22]]
```