Name: Aishani Anavkar

Roll No: 1806

Subject: Python Programming

Topic: NumPy

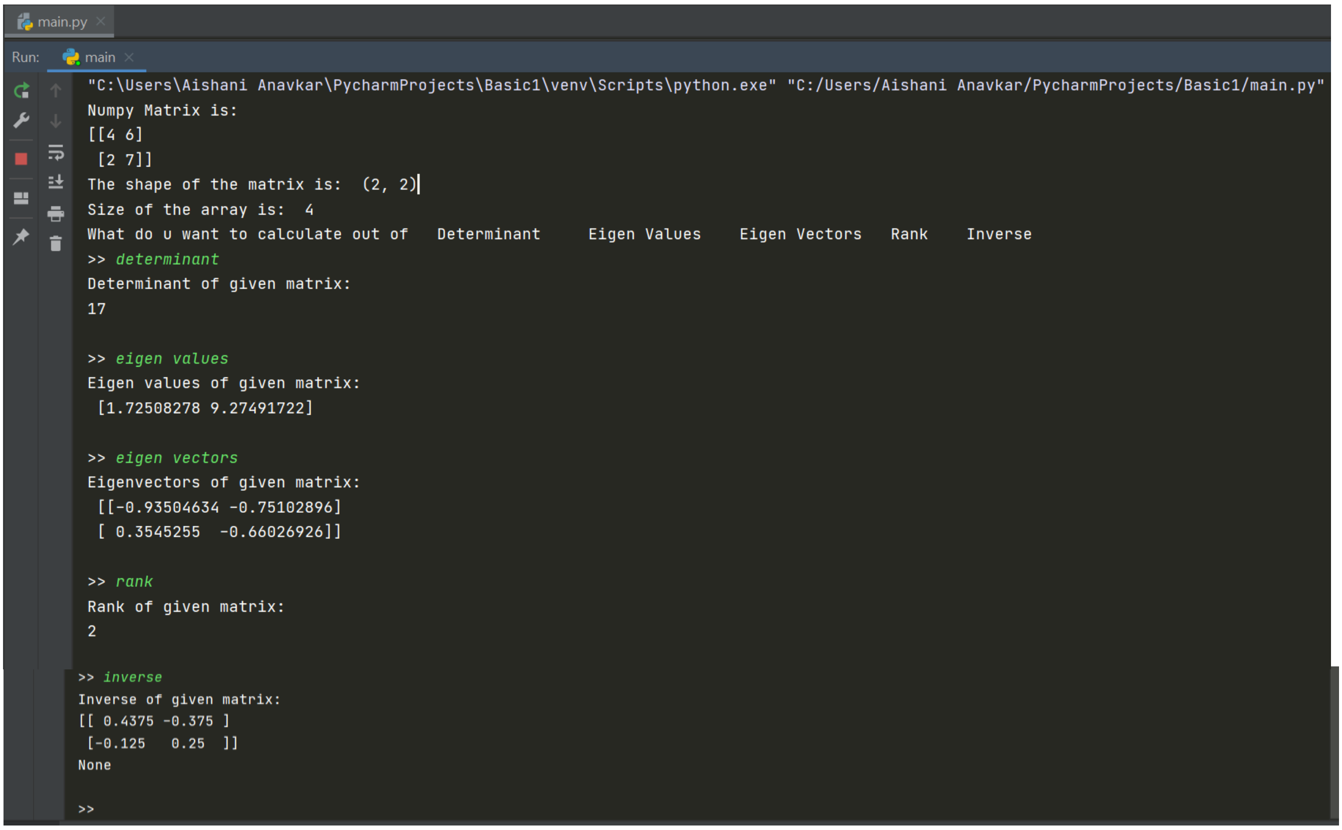
Semester 2

NUMPY PROJECT

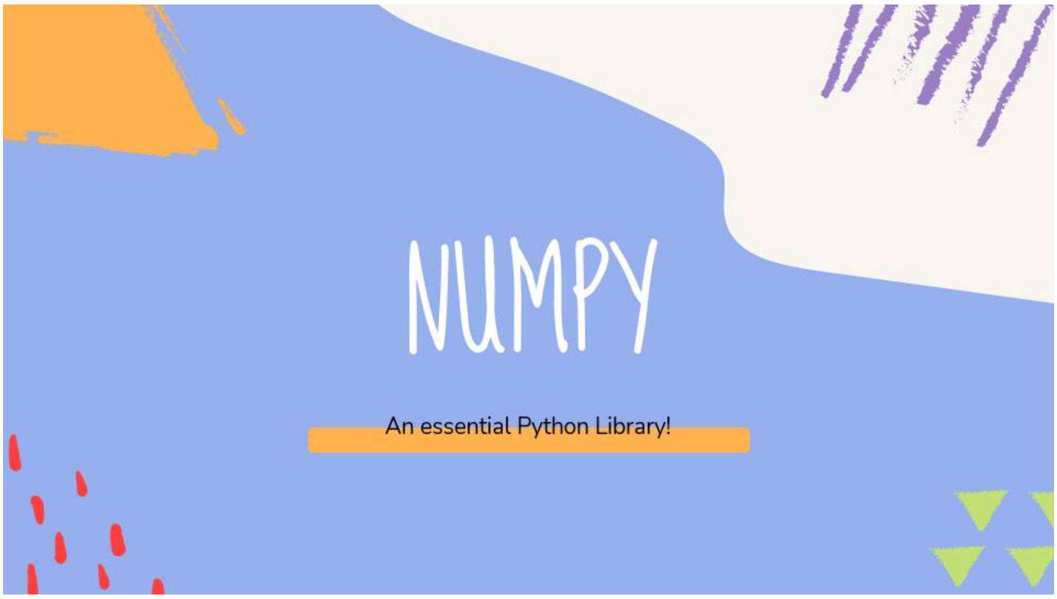
A Matrix Calculator which can calculate the determinant, eigen values, eigen vectors, rank and inverse of a matrix using NumPy library

```
import numpy as np

#forming a matrix
array = np.array([[4, 6], [2, 7]])

#Displaying the square matrix
print("Numpy Matrix is:")
print(array)
a = print("The shape of the matrix is: ", array.shape)       #<-Tells us the
dimensions of matrix
print("Size of the array is: ",array.size)        #<- tells us the size of
matrix


print("What do u want to calculate out of \t Determinant \t Eigen Values \t
Eigen Vectors \t Rank \t Inverse \t ")
while True:
    ask = input(">> ".casefold())

    if ask == "determinant":
        # calculating the determinant of matrix
        det = np.linalg.det(array) +1
        print("Determinant of given matrix:")
        print(int(det),"\n")

    elif ask == "eigen values":
        #calculating eigen values
        w, v = np.linalg.eig(array)
        # printing eigen values
        print("Eigen values of given matrix:\n",w,  "\n")
    elif ask == "eigen vectors":
        # calculating eigen vectors
        w, v = np.linalg.eig(array)
        # printing eigen vectors
        print("Eigenvectors of given matrix:\n",v,  "\n")

    elif ask == "rank":
        # calculating the rank of matrix
        rank = np.linalg.matrix_rank(array)
        #printing rank of matrix
        print("Rank of given matrix:")
        print(int(rank),"\n")

    elif ask == "inverse":
        # calculating the inverse of matrix
        print("Inverse of given matrix:")
        #printing inverse of matrix
        inv = print(np.linalg.inv(array))
        print(inv,"\n")
```

OUTPUT:

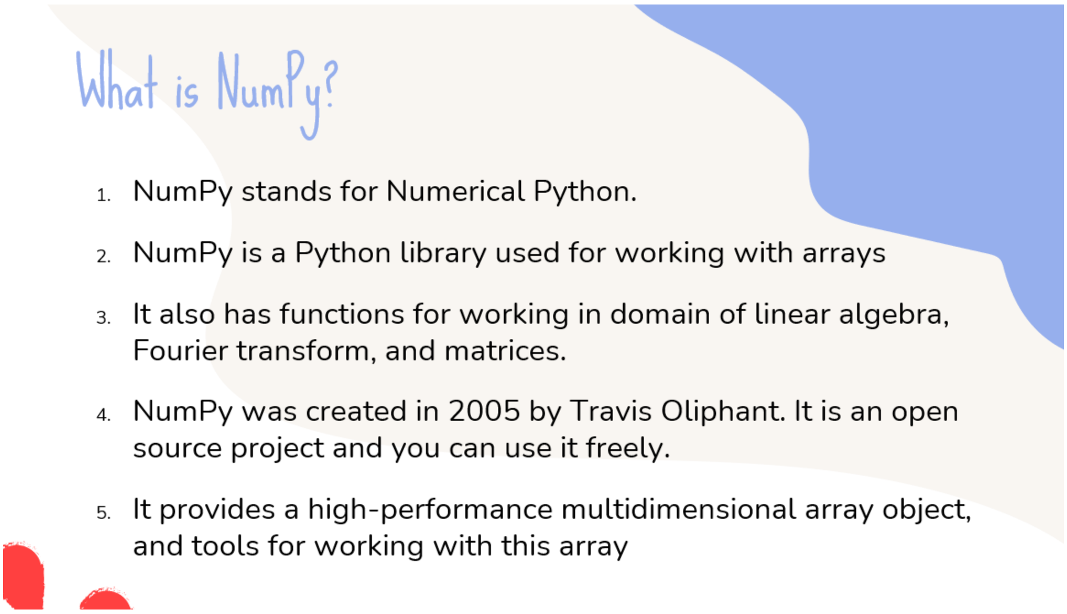


```
"C:\Users\Aishani Anavkar\PycharmProjects\Basic1\venv\Scripts\python.exe" "C:/Users/Aishani Anavkar/PycharmProjects/Basic1/main.py"
Numpy Matrix is:
[[4 6]
 [2 7]]
The shape of the matrix is:  (2, 2)
Size of the array is:  4
What do u want to calculate out of   Determinant     Eigen Values    Eigen Vectors   Rank    Inverse
>> determinant
Determinant of given matrix:
17

>> eigen values
Eigen values of given matrix:
 [1.72508278 9.27491722]

>> eigen vectors
Eigenvectors of given matrix:
 [[-0.93504634 -0.75102896]
 [ 0.3545255  -0.66026926]]

>> rank
Rank of given matrix:
2

>> inverse
Inverse of given matrix:
[[ 0.4375 -0.375 ]
 [-0.125   0.25  ]]
None

>>
```



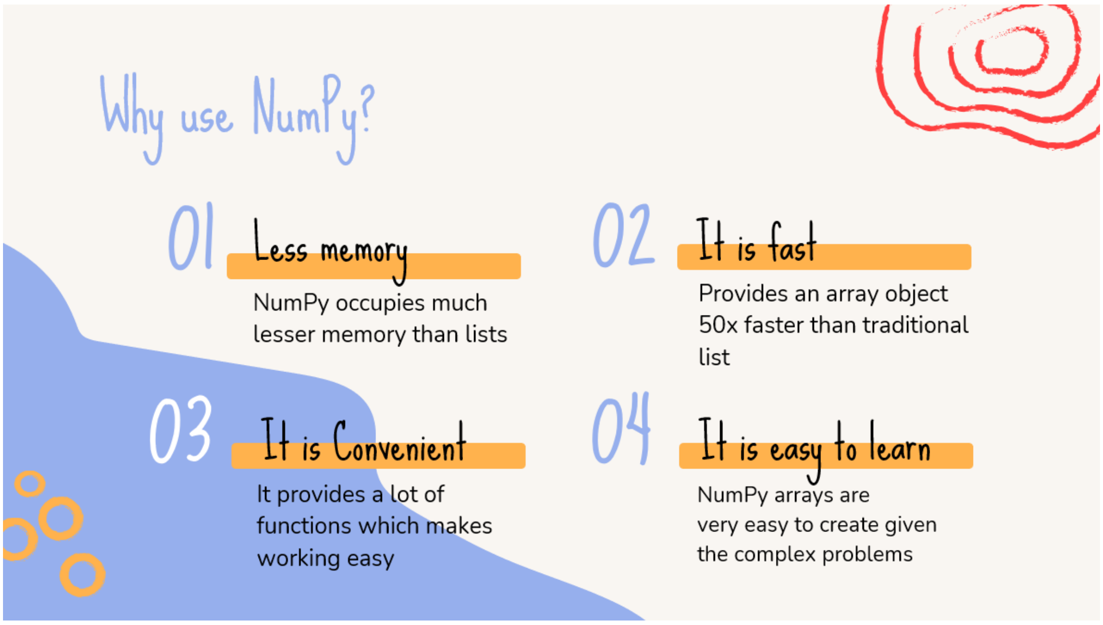
NUMPY

An essential Python Library!



What is NumPy?

1. NumPy stands for Numerical Python.
2. NumPy is a Python library used for working with arrays
3. It also has functions for working in domain of linear algebra, Fourier transform, and matrices.
4. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.
5. It provides a high-performance multidimensional array object, and tools for working with this array



Why use NumPy?

01 Less memory

NumPy occupies much lesser memory than lists

02 It is fast

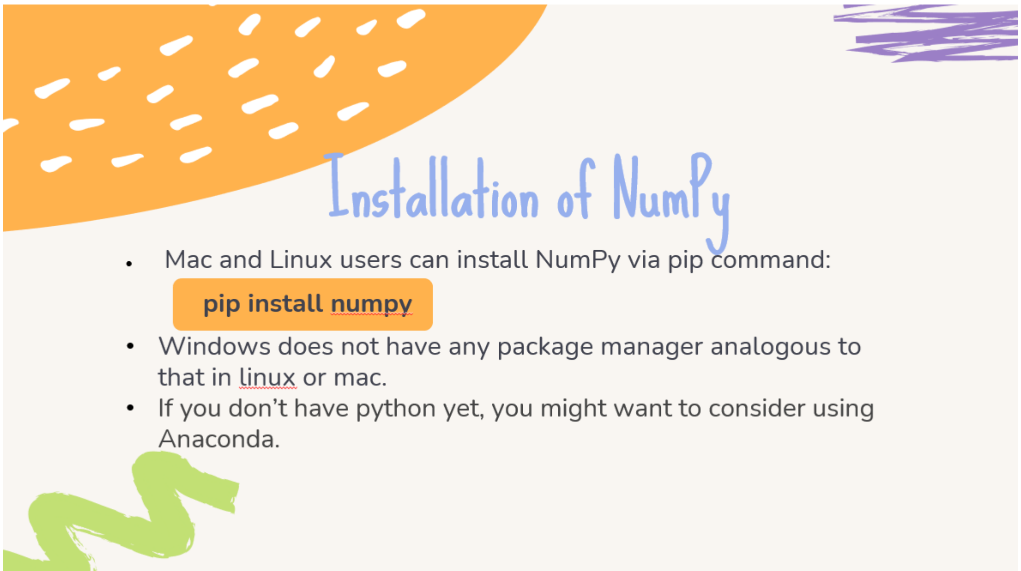
Provides an array object 50x faster than traditional list

03 It is Convenient

It provides a lot of functions which makes working easy

04 It is easy to learn

NumPy arrays are very easy to create given the complex problems

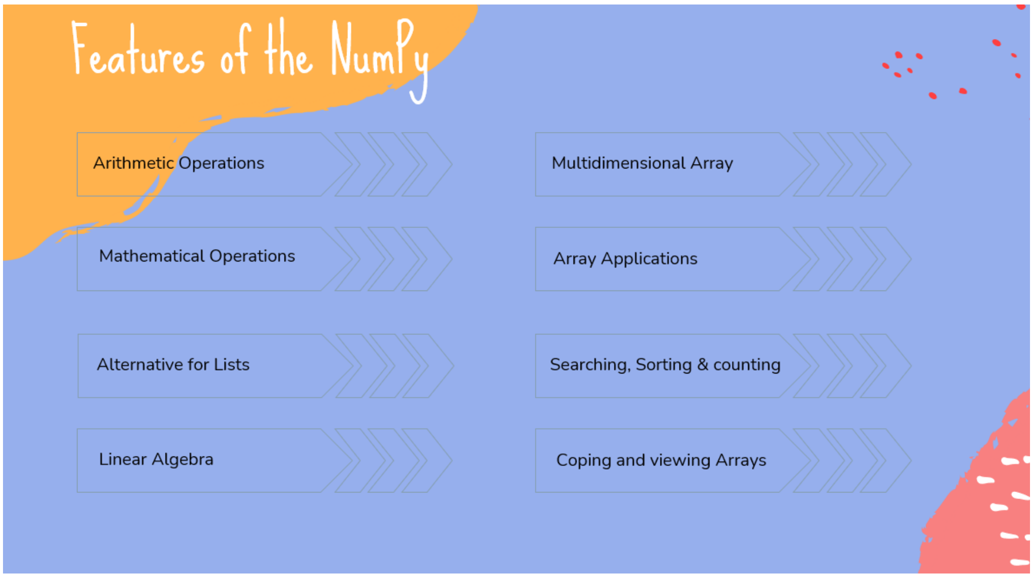


Installation of NumPy

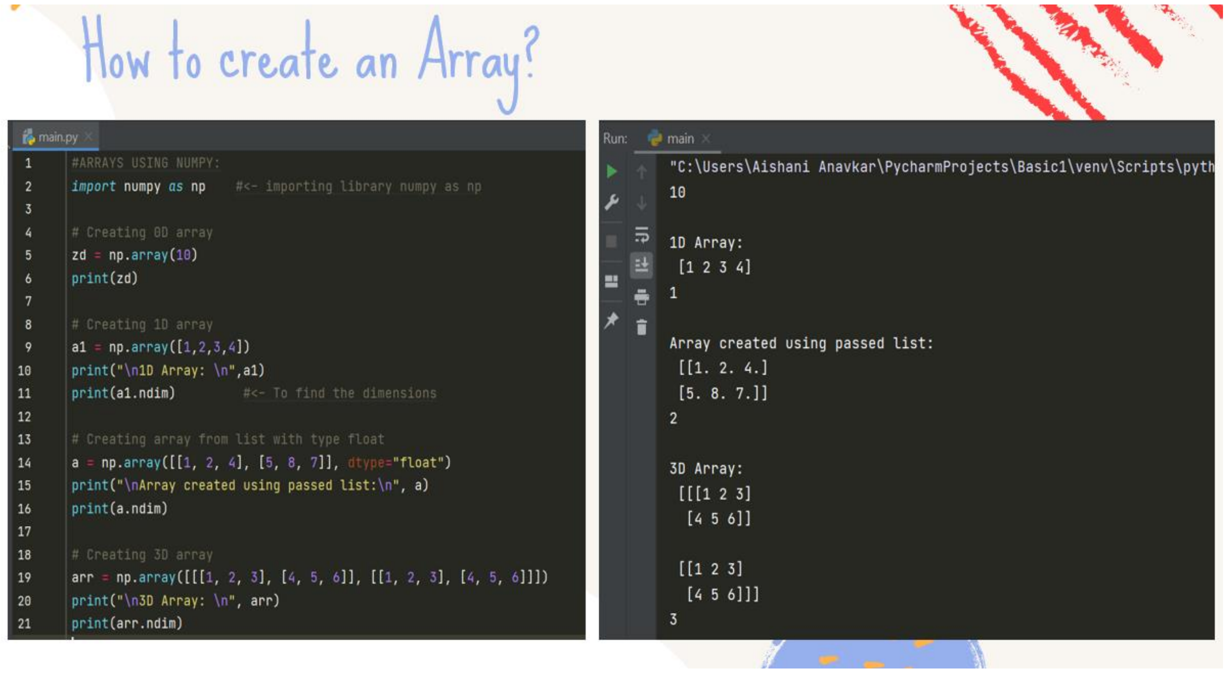
- Mac and Linux users can install NumPy via pip command:

  pip install numpy

- Windows does not have any package manager analogous to that in linux or mac.
- If you don't have python yet, you might want to consider using Anaconda.



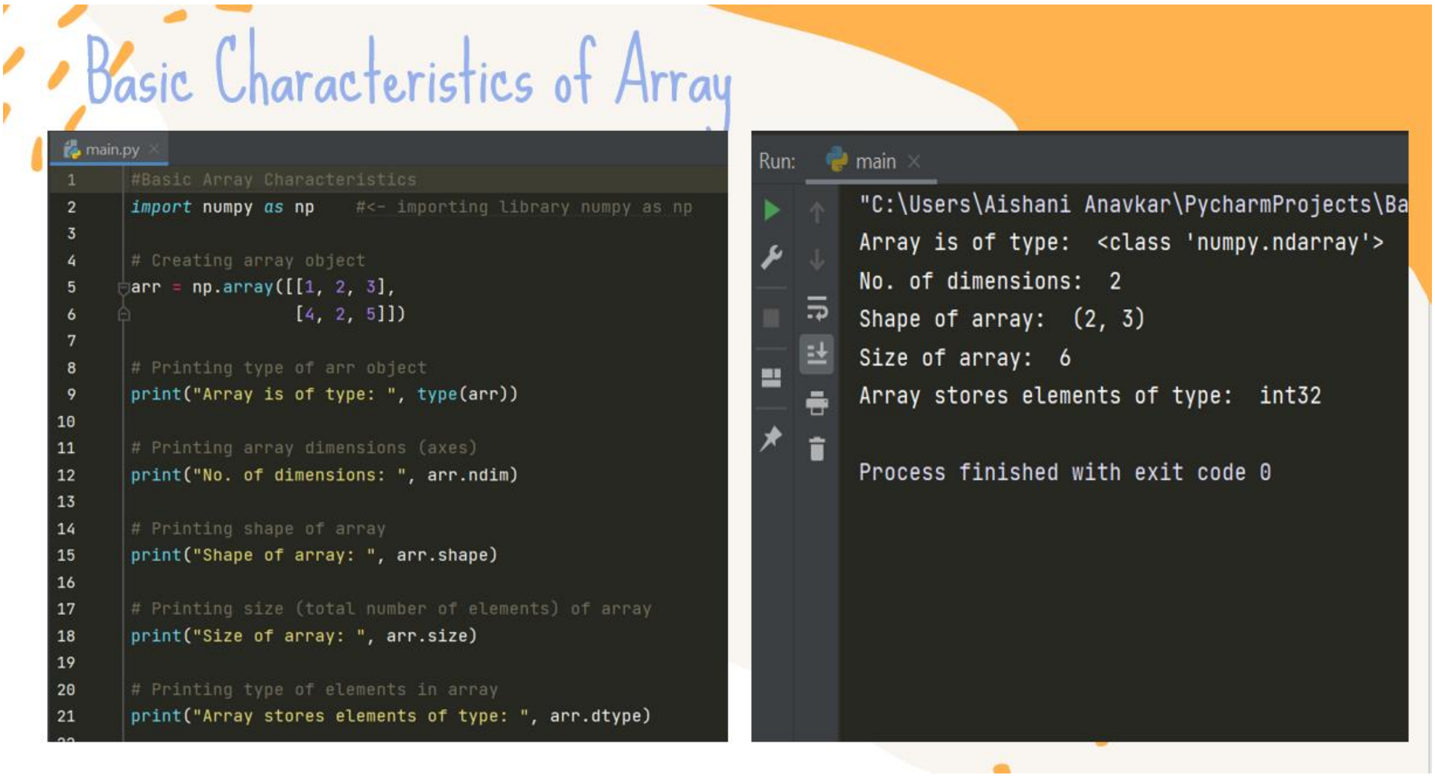
Features of the NumPy



How to create an Array?

```python
#ARRAYS USING NUMPY:
import numpy as np    #<- importing library numpy as np

# Creating 0D array
zd = np.array(10)
print(zd)

# Creating 1D array
a1 = np.array([1,2,3,4])
print("\n1D Array: \n",a1)
print(a1.ndim)          #<- To find the dimensions

# Creating array from list with type float
a = np.array([[1, 2, 4], [5, 8, 7]], dtype="float")
print("\nArray created using passed list:\n", a)
print(a.ndim)

# Creating 3D array
arr = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])
print("\n3D Array: \n", arr)
print(arr.ndim)
```

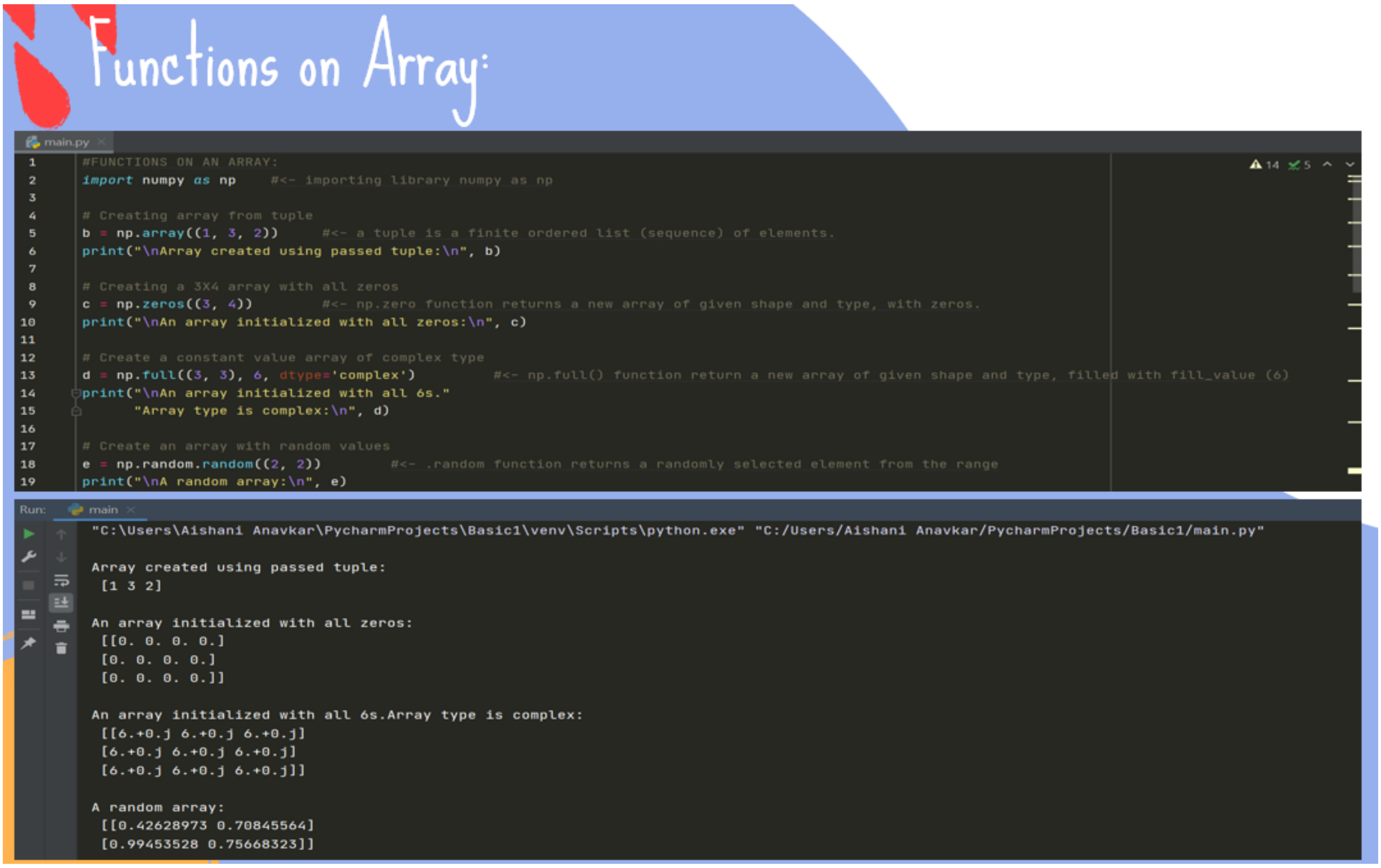
```
"C:\Users\Aishani Anavkar\PycharmProjects\Basic1\venv\Scripts\pyth
10

1D Array:
 [1 2 3 4]
1

Array created using passed list:
 [[1. 2. 4.]
 [5. 8. 7.]]
2

3D Array:
 [[[1 2 3]
  [4 5 6]]

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



Basic Characteristics of Array

```python
#Basic Array Characteristics
import numpy as np    #<- importing library numpy as np

# Creating array object
arr = np.array([[1, 2, 3],
                [4, 2, 5]])

# Printing type of arr object
print("Array is of type: ", type(arr))

# Printing array dimensions (axes)
print("No. of dimensions: ", arr.ndim)

# Printing shape of array
print("Shape of array: ", arr.shape)

# Printing size (total number of elements) of array
print("Size of array: ", arr.size)

# Printing type of elements in array
print("Array stores elements of type: ", arr.dtype)
```

```
"C:\Users\Aishani Anavkar\PycharmProjects\Ba
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:  int32

Process finished with exit code 0
```

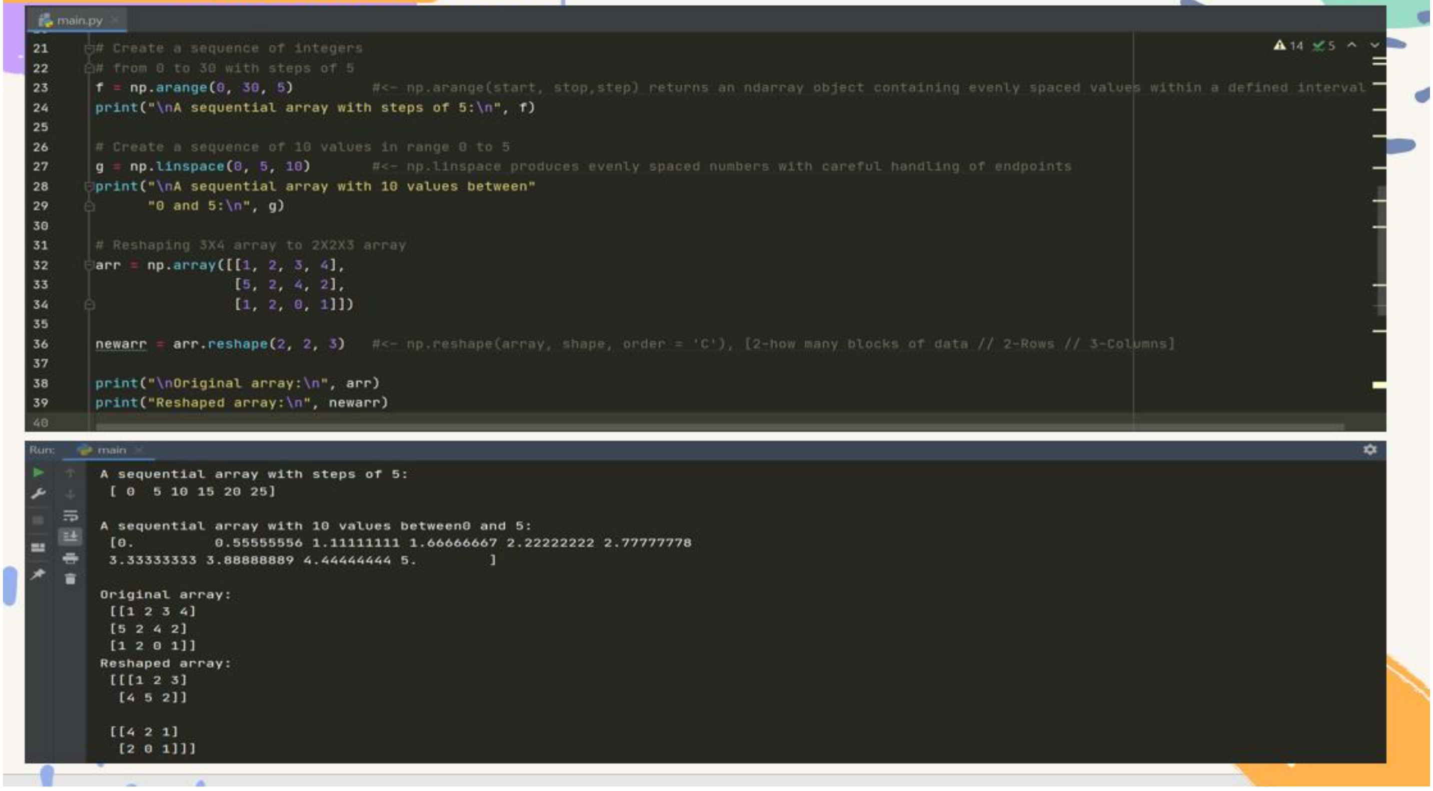


Functions on Array:

```python
#FUNCTIONS ON AN ARRAY:
import numpy as np    #<- importing library numpy as np

# Creating array from tuple
b = np.array((1, 3, 2))      #<- a tuple is a finite ordered list (sequence) of elements.
print("\nArray created using passed tuple:\n", b)

# Creating a 3X4 array with all zeros
c = np.zeros((3, 4))         #<- np.zero function returns a new array of given shape and type, with zeros.
print("\nAn array initialized with all zeros:\n", c)

# Create a constant value array of complex type
d = np.full((3, 3), 6, dtype='complex')         #<- np.full() function return a new array of given shape and type, filled with fill_value (6)
print("\nAn array initialized with all 6s."
      "Array type is complex:\n", d)

# Create an array with random values
e = np.random.random((2, 2))        #<- .random function returns a randomly selected element from the range
print("\nA random array:\n", e)
```

```
"C:\Users\Aishani Anavkar\PycharmProjects\Basic1\venv\Scripts\python.exe" "C:/Users/Aishani Anavkar/PycharmProjects/Basic1/main.py"

Array created using passed tuple:
 [1 3 2]

An array initialized with all zeros:
 [[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]

An array initialized with all 6s.Array type is complex:
 [[6.+0.j 6.+0.j 6.+0.j]
 [6.+0.j 6.+0.j 6.+0.j]
 [6.+0.j 6.+0.j 6.+0.j]]

A random array:
 [[0.42628973 0.70845564]
 [0.99453528 0.75668323]]
```

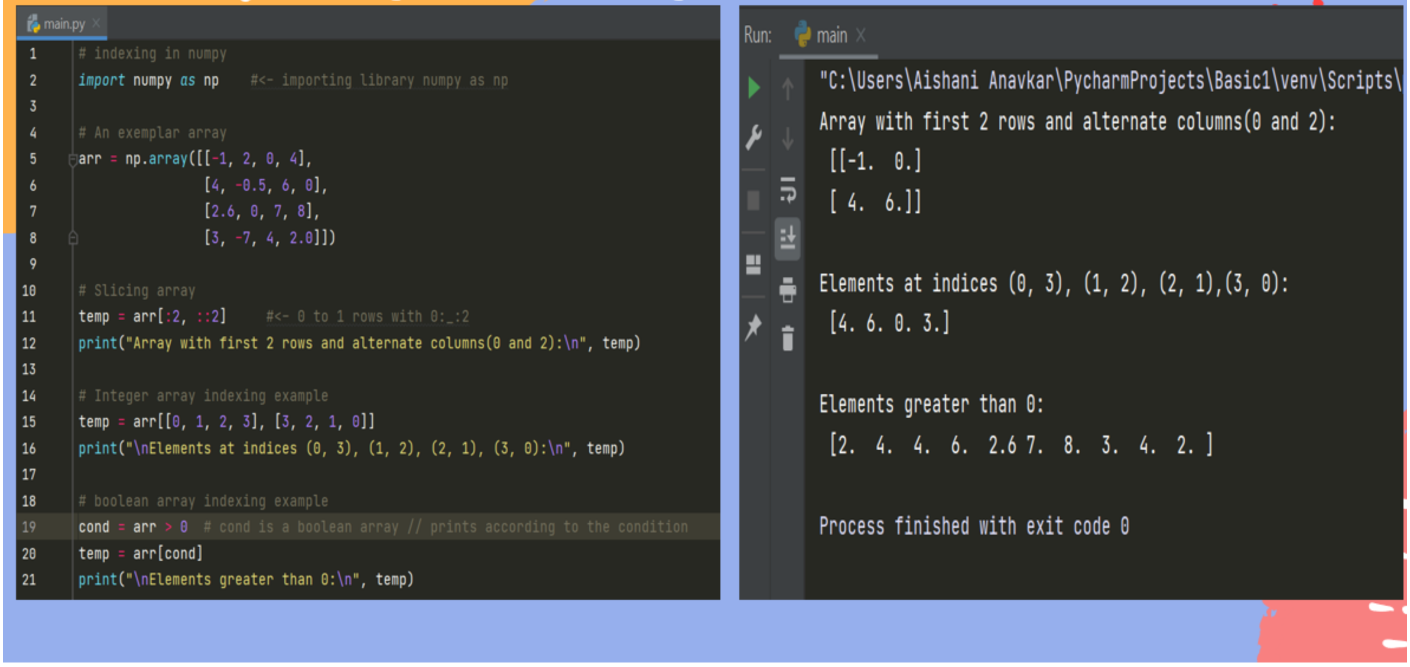
Functions on Array



```python
# Create a sequence of integers
# from 0 to 30 with steps of 5
f = np.arange(0, 30, 5)         #<- np.arange(start, stop,step) returns an ndarray object containing evenly spaced values within a defined interval
print("\nA sequential array with steps of 5:\n", f)

# Create a sequence of 10 values in range 0 to 5
g = np.linspace(0, 5, 10)       #<- np.linspace produces evenly spaced numbers with careful handling of endpoints
print("\nA sequential array with 10 values between"
      "0 and 5:\n", g)

# Reshaping 3X4 array to 2X2X3 array
arr = np.array([[1, 2, 3, 4],
                [5, 2, 4, 2],
                [1, 2, 0, 1]])

newarr = arr.reshape(2, 2, 3)   #<- np.reshape(array, shape, order = 'C'), [2-how many blocks of data // 2-Rows // 3-Columns]

print("\nOriginal array:\n", arr)
print("Reshaped array:\n", newarr)
```

```
A sequential array with steps of 5:
 [ 0  5 10 15 20 25]

A sequential array with 10 values between0 and 5:
 [0.         0.55555556 1.11111111 1.66666667 2.22222222 2.77777778
 3.33333333 3.88888889 4.44444444 5.        ]

Original array:
 [[1 2 3 4]
 [5 2 4 2]
 [1 2 0 1]]
Reshaped array:
 [[[1 2 3]
  [4 5 2]]

 [[4 2 1]
  [2 0 1]]]
```

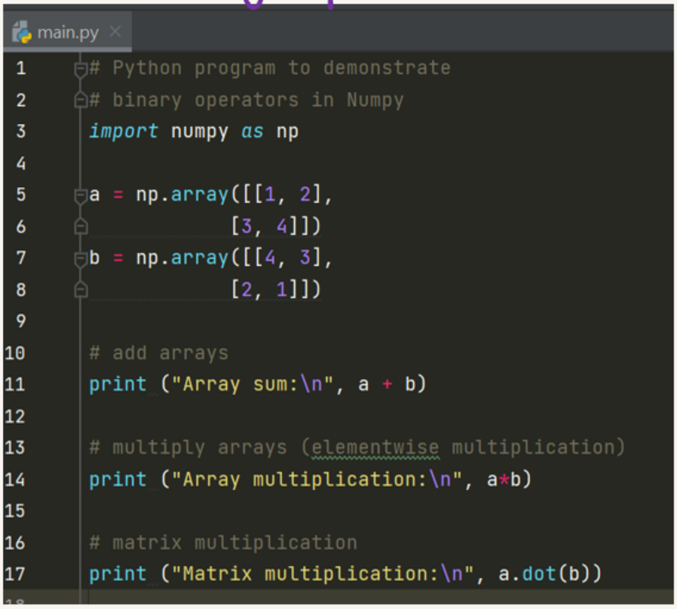
Array slicing and indexing:



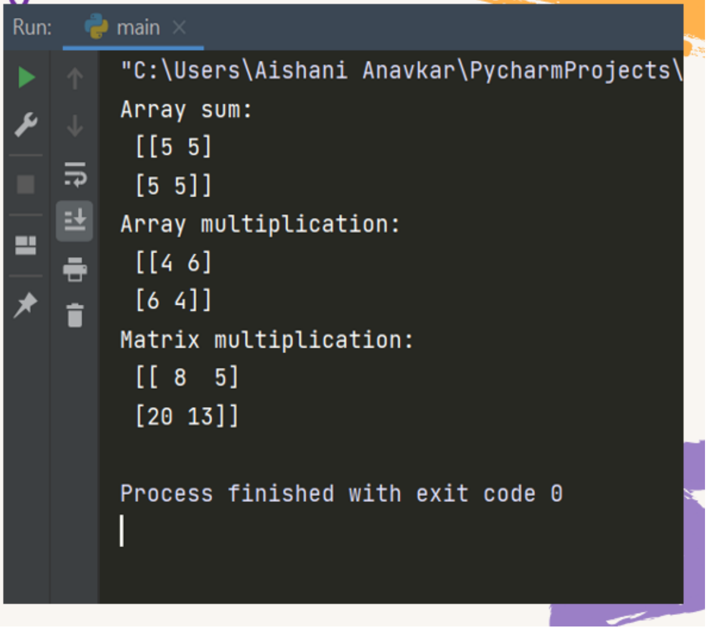
```python
# indexing in numpy
import numpy as np    #<- importing library numpy as np

# An exemplar array
arr = np.array([[-1, 2, 0, 4],
                [4, -0.5, 6, 0],
                [2.6, 0, 7, 8],
                [3, -7, 4, 2.0]])

# Slicing array
temp = arr[:2, ::2]     #<- 0 to 1 rows with 0:_:2
print("Array with first 2 rows and alternate columns(0 and 2):\n", temp)

# Integer array indexing example
temp = arr[[0, 1, 2, 3], [3, 2, 1, 0]]
print("\nElements at indices (0, 3), (1, 2), (2, 1), (3, 0):\n", temp)

# boolean array indexing example
cond = arr > 0  # cond is a boolean array // prints according to the condition
temp = arr[cond]
print("\nElements greater than 0:\n", temp)
```

```
"C:\Users\Aishani Anavkar\PycharmProjects\Basic1\venv\Scripts\
Array with first 2 rows and alternate columns(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. ]

Process finished with exit code 0
```

Binary Operators in NumPy:

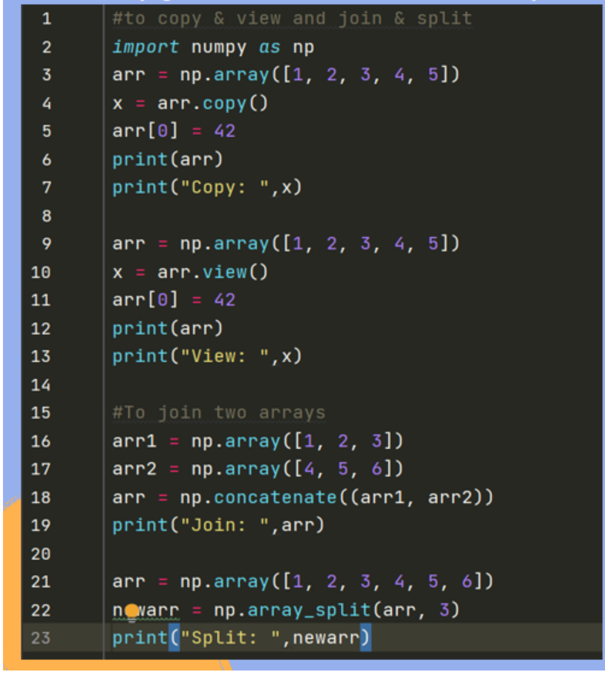


```python
# Python program to demonstrate
# binary operators in Numpy
import numpy as np

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))
```

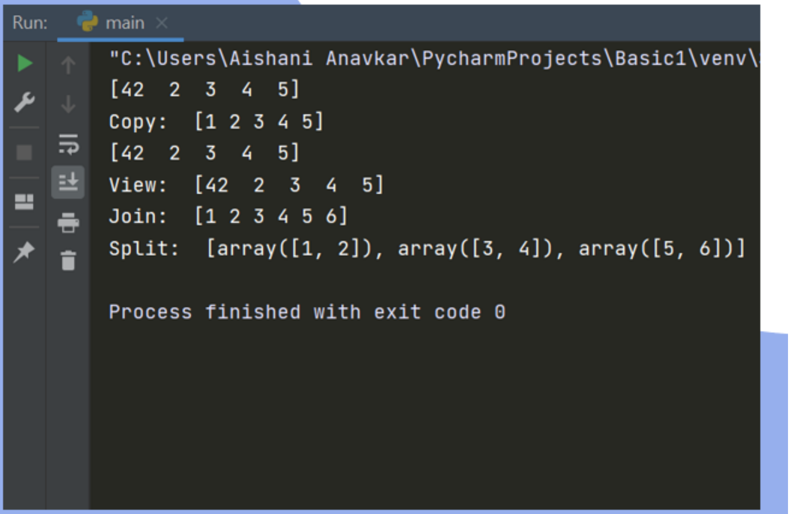


```
"C:\Users\Aishani Anavkar\PycharmProjects\
Array sum:
 [[5 5]
 [5 5]]
Array multiplication:
 [[4 6]
 [6 4]]
Matrix multiplication:
 [[ 8  5]
 [20 13]]

Process finished with exit code 0
```

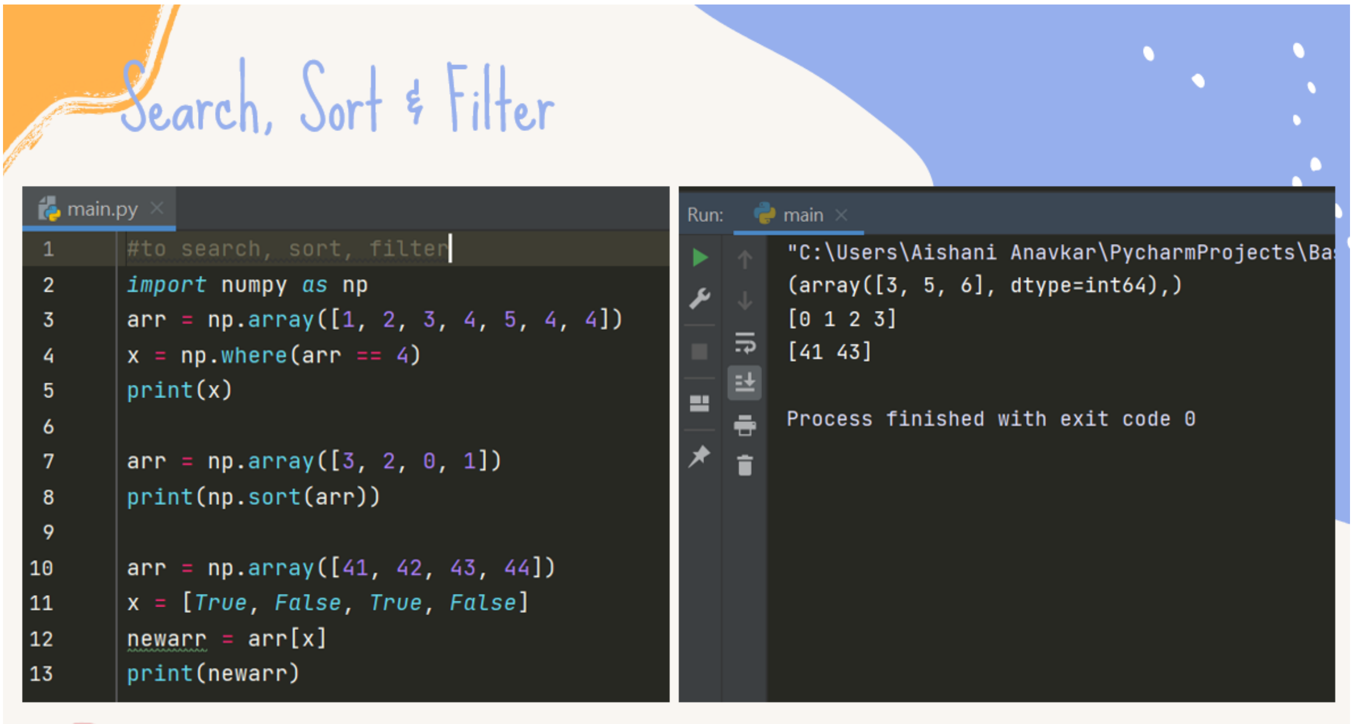
Copy & View, Join & Split



```python
#to copy & view and join & split
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
x = arr.copy()
arr[0] = 42
print(arr)
print("Copy: ",x)

arr = np.array([1, 2, 3, 4, 5])
x = arr.view()
arr[0] = 42
print(arr)
print("View: ",x)

#To join two arrays
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.concatenate((arr1, arr2))
print("Join: ",arr)

arr = np.array([1, 2, 3, 4, 5, 6])
newarr = np.array_split(arr, 3)
print("Split: ",newarr)
```



```
"C:\Users\Aishani Anavkar\PycharmProjects\Basic1\venv\
[42  2  3  4  5]
Copy:  [1 2 3 4 5]
[42  2  3  4  5]
View:  [42  2  3  4  5]
Join:  [1 2 3 4 5 6]
Split:  [array([1, 2]), array([3, 4]), array([5, 6])]

Process finished with exit code 0
```



Search, Sort & Filter

```python
#to search, sort, filter
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x = np.where(arr == 4)
print(x)

arr = np.array([3, 2, 0, 1])
print(np.sort(arr))

arr = np.array([41, 42, 43, 44])
x = [True, False, True, False]
newarr = arr[x]
print(newarr)
```

```
"C:\Users\Aishani Anavkar\PycharmProjects\Ba
(array([3, 5, 6], dtype=int64),)
[0 1 2 3]
[41 43]

Process finished with exit code 0
```