Practical Set 6

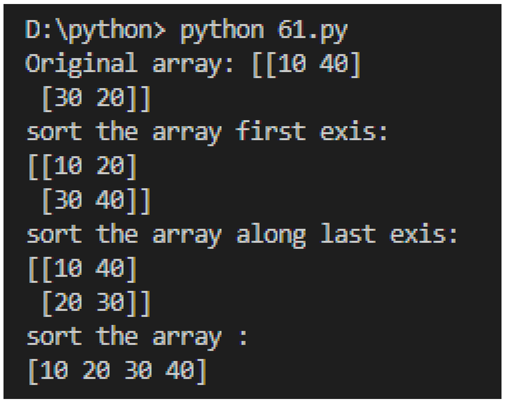
1. **Write a NumPy program to sort a given array of shape 2 along the first axis, last axis and on flattened array**

- **Sourcecode:**

```
import numpy as np

a1=np.array([[10,40],[30,20]])
print("Original array:",a1)
print("sort the array first exis:")
print(np.sort(a1,axis=0))
print("sort the array along last exis:")
print(np.sort(a1))
print("sort the array :")
print(np.sort(a1,axis=None))
```

- **Output:**



```
D:\python> python 61.py
Original array: [[10 40]
 [30 20]]
sort the array first exis:
[[10 20]
 [30 40]]
sort the array along last exis:
[[10 40]
 [20 30]]
sort the array :
[10 20 30 40]
```

2. **Write a NumPy program to create a structured array from given student name, height, class and their data types. Now sort by class, then height if class are equal.**

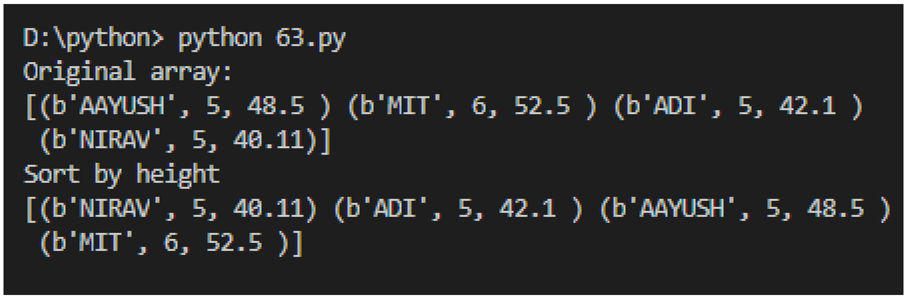
- **Sourcecode:**

```
import numpy as np

data_type = [('name', 'S15'), ('class', int), ('height', float)]
students_details = [('AAYUSH', 5, 48.5), ('MIT', 6, 52.5),('ADI', 5, 42.10), ('NIRAV', 5, 40.11)]

students = np.array(students_details, dtype=data_type)
print("Original array:")
print(students)
print("Sort by height")
print(np.sort(students, order='height'))
```

- **Output:**

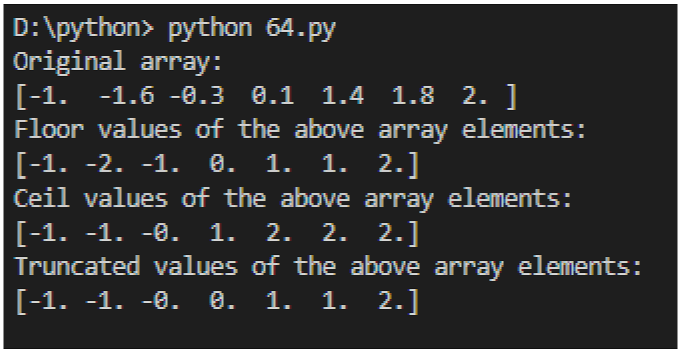


```
D:\python> python 63.py
Original array:
[(b'AAYUSH', 5, 48.5 ) (b'MIT', 6, 52.5 ) (b'ADI', 5, 42.1 )
 (b'NIRAV', 5, 40.11)]
Sort by height
[(b'NIRAV', 5, 40.11) (b'ADI', 5, 42.1 ) (b'AAYUSH', 5, 48.5 )
 (b'MIT', 6, 52.5 )]
```

3. **Write a NumPy program to get the floor, ceiling and truncated values of the elements of a numpy array.**
   - **Sourcecode:**

```
import numpy as np
ar= np.array([-1, -1.6, -0.3, 0.1, 1.4, 1.8, 2.0])
print("Original array:",ar)
print()
print("Floor values of the above array elements:")
print(np.floor(ar))
print("Ceil values of the above array elements:")
print(np.ceil(ar))
print("Truncated values of the above array elements:")
print(np.trunc(ar))
```

- **Output:**

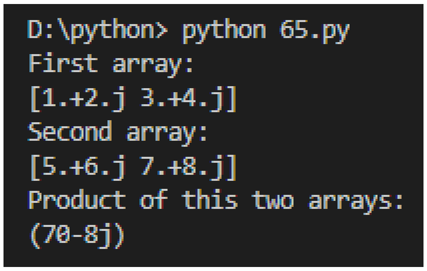


```
D:\python> python 64.py
Original array:
[-1.  -1.6 -0.3  0.1  1.4  1.8  2. ]
Floor values of the above array elements:
[-1. -2. -1.  0.  1.  1.  2.]
Ceil values of the above array elements:
[-1. -1. -0.  1.  2.  2.  2.]
Truncated values of the above array elements:
[-1. -1. -0.  0.  1.  1.  2.]
```

4. **Write a NumPy program to multiply a matrix by another matrix of complex numbers and create a new matrix of complex numbers**
   - **Sourcecode:**

```
import numpy as np
x = np.array([1+2j,3+4j])
print("First array:")
print(x)
y = np.array([5+6j,7+8j])
print("Second array:")
print(y)
z = np.vdot(x, y)
print("Product of this two arrays:")
print(z)
```

   - **Output:**

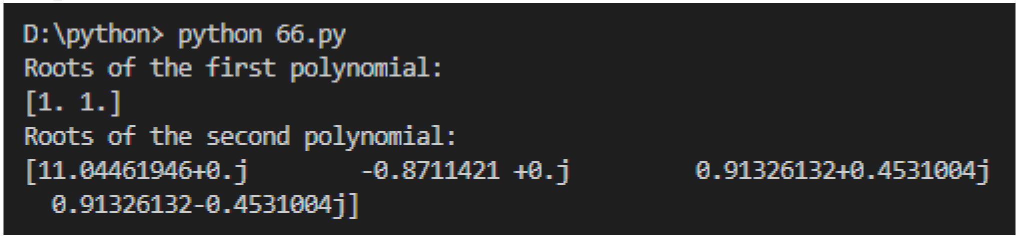


```
D:\python> python 65.py
First array:
[1.+2.j 3.+4.j]
Second array:
[5.+6.j 7.+8.j]
Product of this two arrays:
(70-8j)
```

5. **Write a NumPy program to find the roots of the following polynomials.**
   a) x^2 - 4x + 7.
   b) x^4 - 11x^3 + 9x^2 + 11x – 10
   - **Sourcecode:**

```
import numpy as np
print("Roots of the first polynomial:")
print(np.roots([1, -2, 1]))
print("Roots of the second polynomial:")
print(np.roots([1, -12, 10, 7, -10]))
```

   - **Output:**



```
D:\python> python 66.py
Roots of the first polynomial:
[1. 1.]
Roots of the second polynomial:
[11.04461946+0.j        -0.8711421 +0.j         0.91326132+0.4531004j
  0.91326132-0.4531004j]
```