Practical 5.2

Write following program on NumPy Array:

- Create an array of all the even integers from 30 to 70.
- Create an array of 10 zeros, other with 10 ones, and one more with10 fives.
- Create a vector of length 10 with values evenly distributed between 5 and 50.
- Create a 3x4 matrix filled with values from 10 to 21 and compute sum of all elements, sum of each column and sum of each row of a given array.
- Create a 3x4 array and find the missing data in the array.
- Calculate round, floor, ceiling, truncated and round (to the given number of decimals) of the input, element-wise of an array.
- Find the maximum and minimum value, median, Weighted average, mean, standard deviation, variance, covariance matrix, of a given flattened array, minimum and maximum value along the second axis.
- Create a structured array from given student name, height, class and their data types. Now sort by class, then height if class are equal

In [2]:
```
import numpy as np
```

Create an array of all the even integers from 30 to 70.

In [3]:
```
ar1 = np.arange(30,70,2)
ar1
```

Out[3]:
```
array([30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62,
       64, 66, 68])
```

Create an array of 10 zeros, other with 10 ones, and one more with10 fives.

In [5]:
```
ar2 = np.zeros(10)
ar3 = np.ones(10)
ar4 = np.ones(10) * 5
```

In [7]:
```
print(ar2)
print(ar3)
print(ar4)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
[5. 5. 5. 5. 5. 5. 5. 5. 5. 5.]
```

Create a vector of length 10 with values evenly distributed between 5 and 50.

In [4]:
```
ar5 = np.linspace(5,50,10)
ar5
```

```
Out[4]:  array([ 5., 10., 15., 20., 25., 30., 35., 40., 45., 50.])
```

Create a 3x4 matrix filled with values from 10 to 21 and compute sum of all elements, sum of each column and sum of each row of a given array.

```
In [11]: mat = np.arange(10,22)
         mat = mat.reshape((3,4))
         mat
```

```
Out[11]: array([[10, 11, 12, 13],[14, 15, 16, 17],[18, 19, 20, 21]])
```

```
In [28]: print("Entire sum : ",np.sum(mat))
         print("Row wise sum : ",np.sum(mat,axis=1))
         print("Column wise sum : ",np.sum(mat,axis=0))
```

```
Entire sum :  186
Row wise sum :  [46 62 78]
Column wise sum :  [42 45 48 51]
```

Create a 3x4 array and find the missing data in the array.

```
In [29]: nums = np.array([[3, 2, np.nan, 1],[10, 12, 10, 9],[5, np.nan, 1, np.nan]])
         np.isnan(nums)
```

```
Out[29]: array([[False, False, True, False], [False, False, False, False], [False,  True, False,
               True]])
```

Calculate round, floor, ceiling, truncated and round (to the given number of decimals) of the input, element-wise of an array.

```
In [7]: ar6 = np.array([3.1578, 3.55, 4.55, 2.95, -3.123, -3.7985, -5.569])
        print("Round : ",np.around(ar6))
        print("Floor : ",np.floor(ar6))
        print("Ceil : ",np.ceil(ar6))
        print("Trunc : ",np.trunc(ar6))
        print("Round upto 2 decimal places",np.around(ar6,2))
```

```
Round :  [ 3.  4.  5.  3. -3. -4. -6.]
Floor :  [ 3.  3.  4.  2. -4. -4. -6.]
Ceil :  [ 4.  4.  5.  3. -3. -3. -5.]
Trunc :  [ 3.  3.  4.  2. -3. -3. -5.]
Round upto 2 decimal places [ 3.16  3.55  4.55  2.95 -3.12 -3.8  -5.57]
```

Find the maximum and minimum value, median, Weighted average, mean, standard deviation, variance, covariance matrix, of a given flattened array, minimum and maximum value along the second axis.

```
In [12]: print("Max : ",np.max(ar6))
         print("Min : ",np.min(ar6))
         print("Median : ",np.median(ar6))
         print("Weighted Average : ",np.average(ar6[:4],weights=[0.1,0.2,0.3,0.4]))
         print("Mean : ",np.mean(ar6))
         print("Standard Deviation : ",np.std(ar6))
         print("Variance : ",np.var(ar6))
         print("Covariance matrix : ",np.cov(ar6))
         print("Max value along second axis : ",np.max(mat,axis=1))
         print("Min value along second axis : ",np.min(mat,axis=1))
```

```
Max :  4.55
Min :  -5.569
Median :  2.95
Weighted Average :  3.57078
Mean :  0.24532857142857104
Standard Deviation :  3.9051868500072295
Variance :  15.25048433346939
Covariance matrix :  17.792231722380954
Max value along second axis :  [13 17 21]
Min value along second axis :  [10 14 18]
```

Create a structured array from given student name, height, class and their data types. Now sort by class, then height if class are equal

In [16]:
```
dt = [('name','U10'),('height','i4'),('class','U3')]
student = np.array([('Dev',183,'CE'),('Arav',172,'CSD'),('Dhyan',170,'CE')] , dt
print(np.sort(student, order = ['class','height']))
```

```
[('Dhyan', 170, 'CE') ('Dev', 183, 'CE') ('Arav', 172, 'CSD')]
```