3) Using Numpy module , Perform the follwing operations .

a) Demonstarte Array aggregations functions.

Sum() :- Use to find the sum of the given array.

max() :- It returns the maximum values among the elements of given array.

min() :- It returns the minimum values among the elements of given array. ¶

mean() :- It returns the Mean(Averge) of the input array.

std () :- It shows the standard deviation of array.

median () :- It returns the median value of array.

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

In [21]:
```
a=np.array([20,26,73,84,34,97,45,72])
b=np.array([45,23,66,63,44,85,23,41])
print("set of a array is :",a)
print("set of b array is :",b)
```

```
set of a array is : [20 26 73 84 34 97 45 72]
set of b array is : [45 23 66 63 44 85 23 41]
```

1) Sum () :-

In [26]:

```
s=(a.sum())
print("Sum of a array is :",s)
     # OR
#print("sum of a array is :",np.sum(a))
```

```
Sum of array is : 451
```

2) Max () :-

In [27]:

```
# m=(b.max())
# print("Maximum values of array is :",m)
      # OR
print("Maximum values of array is :",np.max(b))
```

```
Maximum values of array is : 85
```

3) Min () :-

In [28]:

```
# mi=(a.min())
# print("Minimum value of array is :-",mi)
           # OR
print("Minimum value of array is :",np.min(a))
```

```
Minimum value of array is : 20
```

4) Mean () :-

In [29]:

```
# me=(a.mean())
# print("Averge value of array is :",me)
          # OR
print("Averge value of array is :",np.mean(a))
```

```
Averge value of array is : 56.375
```

5) std () :-

In [32]:

```
# st=(b.std())
# print("Standerd diviation of array is :",st)
               # OR
print("Standerd diviation of array is :",np.std(b))
```

```
Standerd diviation of array is : 20.116846174288852
```

6) median () :-

In [36]:

```
# md=(a.std())
# print("Median of a array is :",md)
            # OR
print("Median of a array is :",np.median(a))
```

```
Median of a array is : 58.5
```

b) Demonstarte the vectorized operations.

In [39]:

```
import numpy as np

# creating arrays

a1= np.array([10, 20, 30])
b1= np.array([1, 2, 3])
```

In [41]:

```
print("set of a1 array is :",a1)
print("set of b1 array is :",b1)
```

```
set of a1 array is : [10 20 30]
set of b1 array is : [1 2 3]
```

In [4]:

```
# Arithmetic Operations
 # Addition
 # Subtraction
 # Multiplication
 # Division

print("Addition:", a + b)
print("Subtraction:", a - b)
print("Multiplication:", a * b)
print("Division:", a / b)
```

```
Addition: [11 22 33]
Subtraction: [ 9 18 27]
Multiplication: [10 40 90]
Division: [10. 10. 10.]
```

In [38]:

```
#  Mathematical Functions

print("Square root of a:", np.sqrt(a))
```

```
Square root of a: [3.16227766 4.47213595 5.47722558]
```

In [39]:

```
print("Sum of a:", np.sum(a))
print("Max of b:", np.max(b))
print("Mean of a:", np.mean(a))
```

```
Sum of a: 60
Max of b: 3
Mean of a: 20.0
```

c) Demonstarte the map , filter , reduce , lambda functions with data frame.

In [43]:

```
import pandas as pd
from functools import reduce

data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David'],
    'Age': [25, 30, 35, 40],
    'Salary': [50000, 60000, 70000, 80000]
}

df = pd.DataFrame(data)
df
```

Out[43]:

	Name	Age	Salary
0	Alice	25	50000
1	Bob	30	60000
2	Charlie	35	70000
3	David	40	80000

1) map () :-

In [44]:

```
def add(x):
    return x + 2000

Salary_List = df['Salary'].map(add)
print("Added Salaries:\n", Salary_List)
```

```
Added Salaries:
 0    52000
1    62000
2    72000
3    82000
Name: Salary, dtype: int64
```

2) filter () :-

In [45]:

```
def get(age):
    if age > 30:
        return True
l1=(df['Age'])
res=list(filter(get,l1))
print("Grater then 30 years Age :",res)
```

```
Grater then 30 years Age : [35, 40]
```

3) reduce () :-

In [46]:

```

def add(x, y):
    return x + y

total_salary = reduce(add, df['Salary'])
print("Total Salary:", total_salary)
```

```
Total Salary: 260000
```

In [47]:

```

def max_value(x, y):
    return x if x > y else y

max_age = reduce(max_value, df['Age'])
print("Maximum Age:", max_age)
```

```
Maximum Age: 40
```

4) lambda () :-

In [48]:

```
# Age grater then 30 years using lambda

old= df[df['Age'].apply(lambda x: x > 30)]
print(old)
```

```
      Name  Age  Salary
2  Charlie   35   70000
3    David   40   80000
```

In [49]:

```
# Sum of salary using lambda

total= reduce(lambda x, y: x + y, df['Salary'])
print("Total Salary:", total)
```

```
Total Salary: 260000
```