Q1. Create two 3×3 matrices using the random function in Numpy and perform the following operations. è Product (prod) è Multiplication (multiply) è Dot Product (dot)

In [ ]:
```
import numpy as np

a = np.random.rand(3, 3)
b = np.random.rand(3, 3)

prod = np.prod(a)
mult = np.multiply(a, b)
dotp = np.dot(a, b)

print("Product (prod):", prod)
print("Multiplication (multiply):\n", mult)
print("Dot Product (dot):\n", dotp)
```

```
Product (prod): 8.810799298355092e-06
Multiplication (multiply):
 [[0.10935911 0.44911857 0.57081179]
 [0.05588168 0.48883598 0.10737001]
 [0.17675448 0.58559544 0.01591754]]
Dot Product (dot):
 [[1.17371436 1.48684462 1.24795653]
 [0.5757612  0.8517223  0.60645082]
 [0.3494817  0.62809957 0.37816073]]
```

Q2. Perform the following set operations using the Numpy functions. è Union è Intersection è Set difference è XOR

In [ ]:
```
import numpy as np

A = np.array([1, 2, 3, 4, 5])
B = np.array([3, 4, 5, 6, 7])

U = np.union1d(A, B)
I = np.intersect1d(A, B)
D_AB = np.setdiff1d(A, B)
D_BA = np.setdiff1d(B, A)
XOR = np.setxor1d(A, B)

print("Union:", U)
print("Intersection:", I)
print("Set Difference (A - B):", D_AB)
print("Set Difference (B - A):", D_BA)
print("XOR:", XOR)
```

```
Union: [1 2 3 4 5 6 7]
Intersection: [3 4 5]
Set Difference (A - B): [1 2]
Set Difference (B - A): [6 7]
XOR: [1 2 6 7]
```

Q3. Create a 1D array using Random function and perform the following operations. è Cumulative sum è Cumulative Product è Discrete difference (with n=3) è Find the unique elements from the array

In [ ]:
```
import numpy as np

a = np.random.rand(10)

cs = np.cumsum(a)
cp = np.cumprod(a)
dd = np.diff(a, n=3)
ue = np.unique(a)

print("Cumulative Sum:", cs)
print("Cumulative Product:", cp)
print("Discrete Difference (n=3):", dd)
print("Unique Elements:", ue)
```

```
Cumulative Sum: [0.44012415 1.03286997 1.16366949 1.61403817 2.40041667 3.3918775
3
 4.38623274 5.00352113 5.73615856 5.74664324]
Cumulative Product: [4.40124152e-01 2.60881749e-01 3.41232083e-02 1.53680242e-02
 1.20850839e-02 1.19818877e-02 1.19142524e-02 7.35452972e-03
 5.38820371e-03 5.64935954e-05]
Discrete Difference (n=3): [ 1.3960834  -0.76507478 -0.14736814 -0.07126053 -0.17
777315  0.872377
 -1.32991761]
Unique Elements: [0.01048468 0.13079952 0.44012415 0.45036868 0.59274581 0.617288
39
 0.73263742 0.78637851 0.99146086 0.9943552 ]
```

Q4. Create two 1D array and perform the Addition using zip(), add() and user defined function (frompyfunc())

In [ ]:
```
import numpy as np

a_input = input("Enter values for array 'a' comma-separated: ")
b_input = input("Enter values for array 'b' comma-separated: ")
a = np.array([float(x) for x in a_input.split(',')])
b = np.array([float(x) for x in b_input.split(',')])
r_z = np.array([x + y for x, y in zip(a, b)])
r_n = np.add(a, b)

def f(x, y):
    return x + y
add_f = np.frompyfunc(f, 2, 1)
r_c = add_f(a, b)

print("Result using zip():", r_z)
print("Result using np.add():", r_n)
print("Result using custom function (frompyfunc()):", r_c)
```

```
Result using zip(): [ 13.  25.  26.  49.  72. 104.]
Result using np.add(): [ 13.  25.  26.  49.  72. 104.]
Result using custom function (frompyfunc()): [13.0 25.0 26.0 49.0 72.0 104.0]
```

Q5. Find the LCM (Least Common Multiple) and GCD (Greatest Common Divisor) of an array of elements using reduce().

In [ ]:
```
from functools import reduce
import numpy as np
from math import gcd
```

```
input_str = input("Enter values for the array comma-separated: ")
a = np.array([int(x) for x in input_str.split(',')])

def calculate_lcm(x, y):
    return x * y // gcd(x, y)

def calculate_gcd(x, y):
    while y != 0:
        x, y = y, x % y
    return x

lcm = reduce(calculate_lcm, a)
gcd = reduce(calculate_gcd, a)

print("LCM:", lcm)
print("GCD:", gcd)
```

```
LCM: 360
GCD: 6
```