# Practical No :- 3

**Name:-** DATTARAJ SUBHASH VAGARE

**Div:-** H1

**Roll No :-** 819

**PRN No:-** 202201050037

## Q. Prepare/Take datasets for any real-life application. Read a dataset into an array.

## Perform the following operations on it:

1. Perform all matrix operations
2. Horizontal and vertical stacking of Numpy Arrays
3. Custom sequence generation
4. Arithmetic and Statistical Operations, Mathematical Operations, Bitwise Operators
5. Copying and viewing arrays
6. Data Stacking, Searching, Sorting, Counting, Broadcasting

**CODE**

```
import numpy as np
m1=np.array([[2,4,6],[8,1,3],[5,7,9]])
m2=np.array([[12,13,14],[10,11,5],[9,15,10]])


#addition
add_result=m1+m2
print("add result:")
print(add_result)

#subtraction
sub_result=m1-m2
print("sub result:")
print(sub_result)
sub_result=m1-m2
print("sub result:")
print(sub_result)
```

```
# multiplication

multiplication_result=np.dot(m1,m2)
print("multiplication result:")
print(multiplication_result)

# division

division_result=m2%m1
print("division result:")
print(division_result)

#inverse

inverse_result=np.linalg.inv(m1)
print("\n inverse result:")
print(inverse_result)
```

```
#transpose

transpose_result=np.linalg.inv(m1)
print("\n transpose result:")
print(transpose_result)

#view

array=np.array([2,4,10,11,5])
array.view
array[0]=50
print("array view:")
print(array)
```

```
#coppy


array=np.array([2,4,10,11,5])
array.copy
print("array copy:")


print(array)


#Horizontal and vertical stacking


verticalstack_result=np.vstack((m1,m2))
print("vertical stack:")
print(verticalstack_result)
Horizontalstack_result=np.hstack((m1,m2))
print("Horizontal stack:")
print(Horizontalstack_result)
```

```
# Bitwise Operators


bitwise_and = np.bitwise_and(m1,m2)
print("bitwise_and:")
print(bitwise_and)


bitwise_or = np.bitwise_or(m1,m2)
print("bitwise_or:")
print(bitwise_or)
```

# OUTPUT

```
>>>
========== RESTART: C:/Users/Admin/s1.py ==========
add result:
[[14 17 20]
 [18 12  8]
 [14 22 19]]
sub result:
[[-10  -9  -8]
 [ -2 -10  -2]
 [ -4  -8  -1]]
multiplication result:
[[118 160 108]
 [133 160 147]
 [211 277 195]]
division result:
[[0 1 2]
 [2 0 2]
 [4 1 1]]

 inverse result:
[[-0.22222222  0.11111111  0.11111111]
 [-1.05555556 -0.22222222  0.77777778]
 [ 0.94444444  0.11111111 -0.55555556]]

 transpose result:
[[-0.22222222  0.11111111  0.11111111]
 [-1.05555556 -0.22222222  0.77777778]
 [ 0.94444444  0.11111111 -0.55555556]]
array view:
[50  4 10 11  5]
array copy:
[ 2  4 10 11  5]
vertical stack:
[[ 2  4  6]
 [ 8  1  3]
 [ 5  7  9]
 [12 13 14]
 [10 11  5]
 [ 9 15 10]]
Horizontal stack:
[[ 2  4  6 12 13 14]
 [ 8  1  3 10 11  5]
 [ 5  7  9  9 15 10]]
Horizontal stack:
[[ 2  4  6 12 13 14]
 [ 8  1  3 10 11  5]
 [ 5  7  9  9 15 10]]
bitwise_and:
[[0 4 6]
 [8 1 1]
 [1 7 8]]
bitwise_or:
[[14 13 14]
 [10 11  7]
 [13 15 11]]
```