```
import numpy as np
import random
num_rows = 2
num_columns = 2
X = np.random.random((num_rows, num_columns))
print("First random square matrix is\n",X)
n_rows = 2
n_columns = 2
Y = np.random.random((n_rows,n_columns))
print("Second random square matrix is\n",Y)
```

```
First random square matrix is
 [[0.12920698 0.88840691]
 [0.39005151 0.45417176]]
Second random square matrix is
 [[0.0210374  0.0982803 ]
 [0.85937657 0.79304925]]
```

MATRIX ADDITION WITHOUT NUMPY

```
result = [[0,0],
        [0,0]]

# iterate through rows
for i in range(len(X)):
# iterate through columns
    for j in range(len(X[0])):
        result[i][j] = X[i][j] + Y[i][j]

for r in result:
    print(r)
```

```
[0.1502443831849707, 0.986687206547859]
[1.2494280746352047, 1.2472210115933646]
```

MATRIX ADDITION WITH NUMPY

```
a = np.matrix(X)
b = np.matrix(Y)
c = a+b
print(c)
```

```
[[0.15024438 0.98668721]
 [1.24942807 1.24722101]]
```

MATRIX MULTIPLICATION WITHOUT NUMPY

```
# List to store matrix multiplication result
result = [[0, 0],
     [0, 0]]

# iterate through rows of X
for i in range(len(X)):
   # iterate through columns of Y
   for j in range(len(Y[0])):
       # iterate through rows of Y
       for k in range(len(Y)):
           result[i][j] += X[i][k] * Y[k][j]

for r in result:
   print(r)
```

```
     [0.7661942566668477, 0.717248936654244]
     [0.3985102347815859, 0.3985149522285371]
```

MATRIX MULTIPLICATION WITH NUMPY

```
result = np.dot(X, Y)

# printing the result
print("The matrix multiplication is :")
print(result)
```

```
     The matrix multiplication is :
     [[0.76619426 0.71724894]
      [0.39851023 0.39851495]]
```

TRANSPOSE WITHOUT NUMPY

```
result = [[0,0],
         [0,0]]

# iterate through rows
for i in range(len(X)):
   # iterate through columns
   for j in range(len(X[0])):
       result[j][i] = X[i][j]

for r in result:
   print(r)
```

```
     [0.12920698360858718, 0.3900515089656009]
```

```
[0.8884069082448555, 0.45417175767178397]
```

TRANSPOSE WITH NUMPY

```
print(np.transpose(X))
```

```
[[0.12920698 0.39005151]
 [0.88840691 0.45417176]]
```

DETERMINANT WITHOUT NUMPY

```
def determinant(matrix, mul):
 width = len(matrix)
 if width == 1:
    return mul * matrix[0][0]
 else:
    sign = -1
    sum = 0
    for i in range(width):
        m = []
        for j in range(1, width):
            buff = []
            for k in range(width):
                if k != i:
                    buff.append(matrix[j][k])
            m.append(buff)
        sign *= -1
        sum += mul * determinant(m, sign * matrix[0][i])
    return sum


print(determinant(X, 1))
```

```
-0.2878422922873886
```

DETERMINANT WITH NUMPY

```
det = np.linalg.det(X)

print("\nDeterminant of given 2X2 matrix:")
print(det)
```

```
Determinant of given 2X2 matrix:
-0.2878422922873886
```

INVERSE OF MATRIX,WITHOUT NUMPY

```
def det2(X):
    return X[0][0]*X[1][1] - X[0][1]*X[1][0]

def inv2(X):
    d = det2(X)
    return [[X[1][1]/d, -X[0][1]/d], [-X[1][0]/d, X[0][0]/d]]


print(inv2(X))
```

```
[[-1.5778492940096795, 3.0864363300645508], [1.355087558072127, -0.44888116538338235]]
```

INVERSE OF MATRIX WITH NUMPY

```
print(np.linalg.inv(X))
```

```
[[-1.57784929  3.08643633]
 [ 1.35508756 -0.44888117]]
```