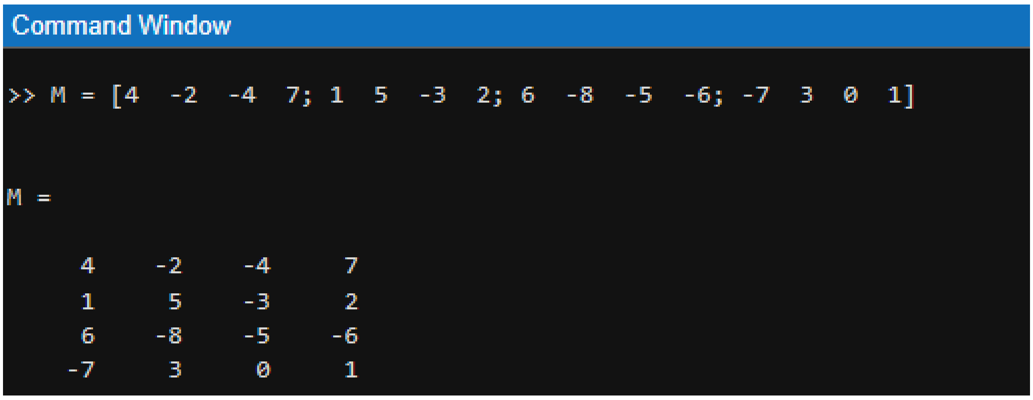
Matrix Elements and Matrix Operations using MATLAB

Objective:

To provide hands-on practice in MATLAB, covering matrix creation and matrix operations.

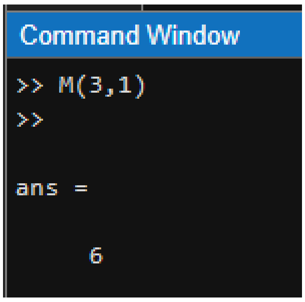
1. Create the following matrix M (4 X 4).

M = [4 -2 -4 7, 1 5 -3 2, 6 -8 -5 -6, -7 3 0 1]

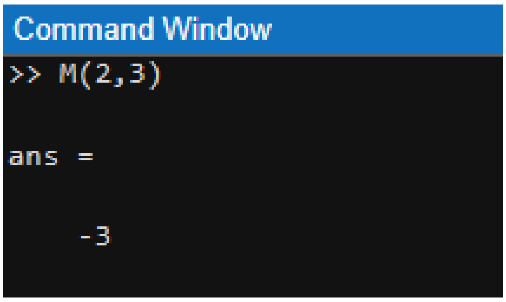


```
>> M = [4  -2  -4  7; 1  5  -3  2; 6  -8  -5  -6; -7  3  0  1]


M =

     4    -2    -4     7
     1     5    -3     2
     6    -8    -5    -6
    -7     3     0     1
```

2. Access the following elements of matrix M. M(3,1) and M(2,3)

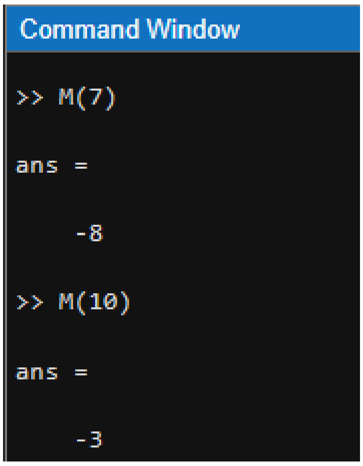


```
>> M(3,1)
>>

ans =

     6
```



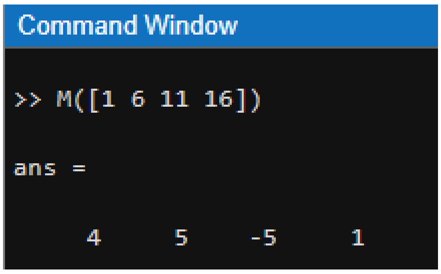
```
>> M(2,3)

ans =

    -3
```

3. Access the following elements of matrix M using a single number. (-8, -3)



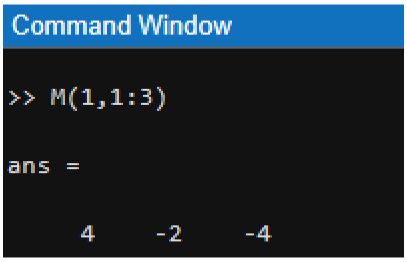
```
>> M(7)

ans =

    -8

>> M(10)

ans =

    -3
```

4. Using the single indexing extract, the elements belong to these indexes. (1 6 11 16)



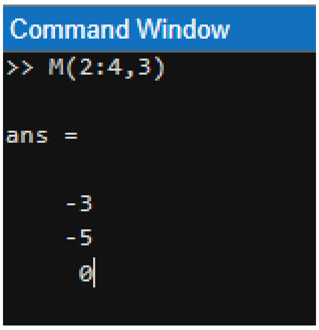
```
>> M([1 6 11 16])

ans =

     4     5    -5     1
```

5. Lists all values in row 1 which are between columns 1 and 3 inclusive.



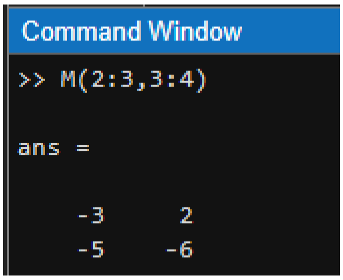
```
>> M(1,1:3)

ans =

     4    -2    -4
```

6. Lists all the values in column 3 which are between rows 2 to 4 inclusive.



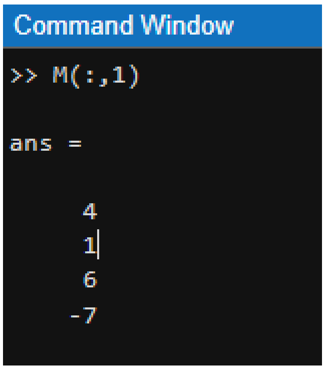
```
>> M(2:4,3)

ans =

    -3
    -5
     0
```

7. Lists the 2 by 2 block of values which lie between rows 2 to 3 and columns 3 to 4.



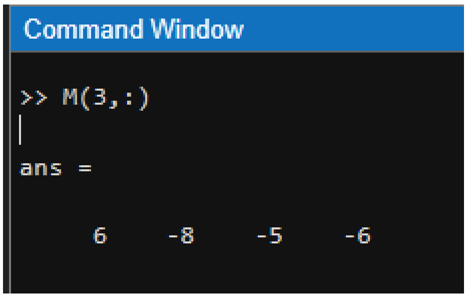
```
>> M(2:3,3:4)

ans =

    -3     2
    -5    -6
```

8. Access the entire first column in matrix M



```
>> M(:,1)

ans =

     4
     1
     6
    -7
```

9. Access the entire third row in matrix M.

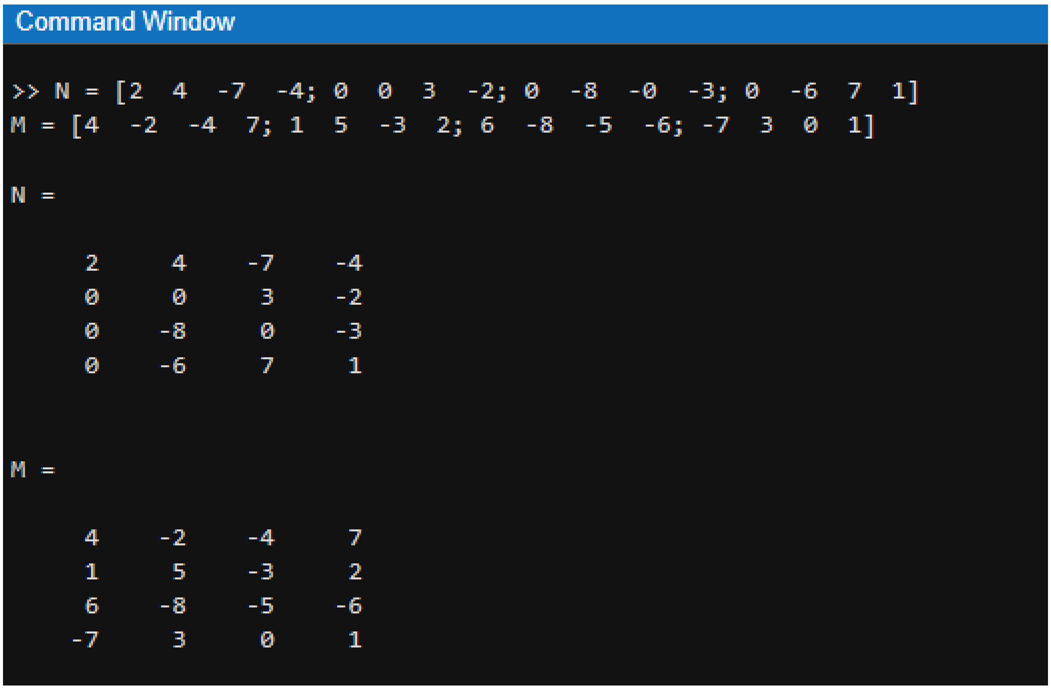


```
>> M(3,:)

ans =

     6    -8    -5    -6
```

Matrix Operations

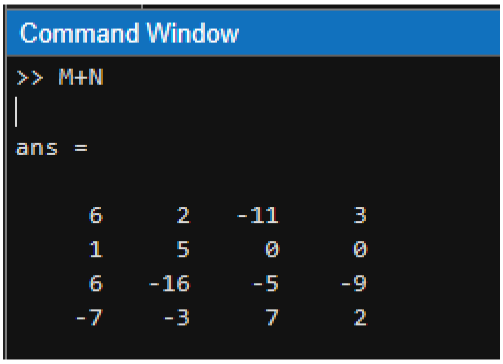
1. Create the following matrix N (4X4).

N = [2 4 -7 -4, 0 0 3 -2, 0 -8 -0 -3, 0 -6 7 1]
M = [4 -2 -4 7, 1 5 -3 2, 6 -8 -5 -6, -7 3 0 1]



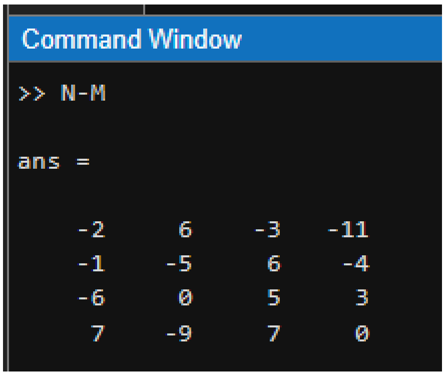
```
>> N = [2  4  -7  -4; 0  0  3  -2; 0  -8  -0  -3; 0  -6  7  1]
M = [4  -2  -4  7; 1  5  -3  2; 6  -8  -5  -6; -7  3  0  1]

N =

     2     4    -7    -4
     0     0     3    -2
     0    -8     0    -3
     0    -6     7     1


M =

     4    -2    -4     7
     1     5    -3     2
     6    -8    -5    -6
    -7     3     0     1
```

2. Add the matrices N and M.



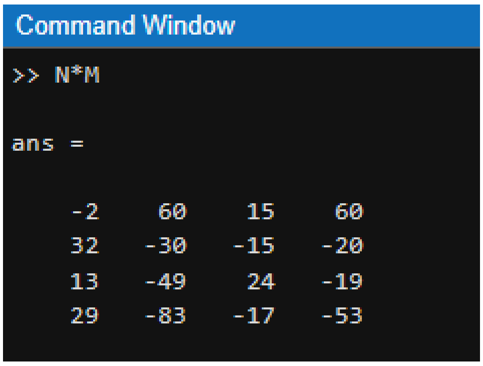
```
>> M+N

ans =

     6     2   -11     3
     1     5     0     0
     6   -16    -5    -9
    -7    -3     7     2
```

3. Subtract the matrix M from matrix N.



```
>> N-M

ans =

    -2     6    -3   -11
    -1    -5     6    -4
    -6     0     5     3
     7    -9     7     0
```

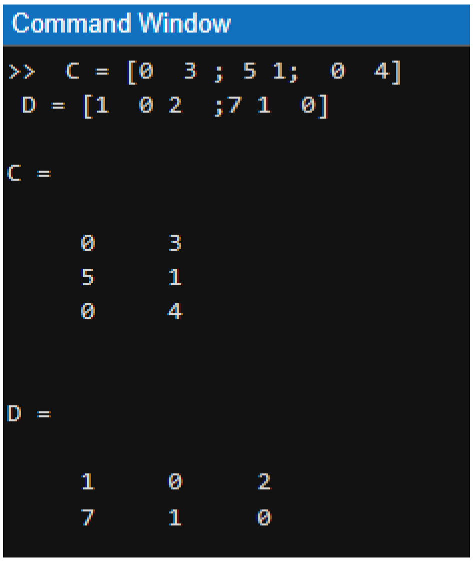
4. Multiply both matrices N and M.



```
>> N*M

ans =

    -2    60    15    60
    32   -30   -15   -20
    13   -49    24   -19
    29   -83   -17   -53
```

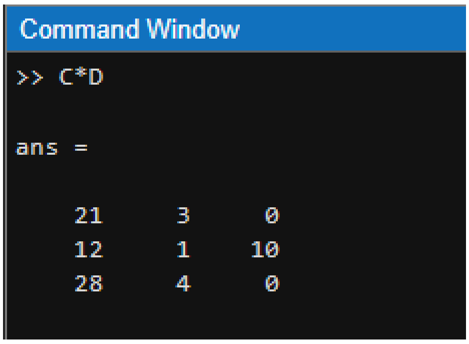
5. Create following matrix C (3X2) and matrix D (2X3).

C = [0 3 , 5 1, 0 4] and D = [1 0 2 ,7 1 0]

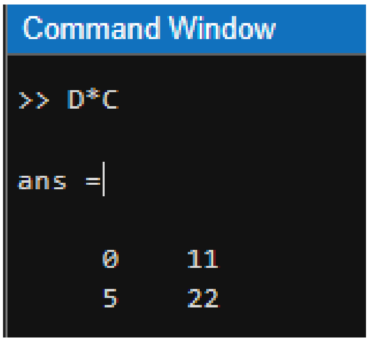


```
>>  C = [0  3 ; 5 1;  0  4]
 D = [1  0 2  ;7 1  0]

C =

     0     3
     5     1
     0     4


D =

     1     0     2
     7     1     0
```

6. Find the answers for C * D and D * C.

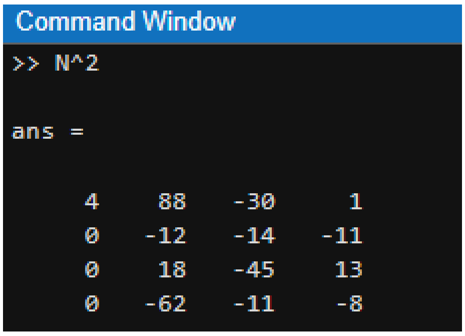


```
>> C*D

ans =

    21     3     0
    12     1    10
    28     4     0
```



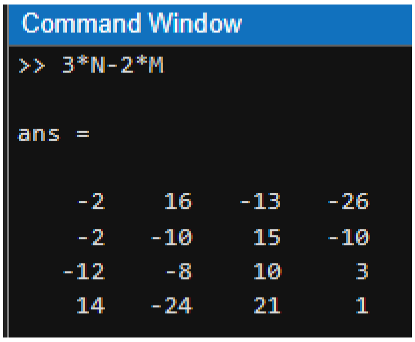
```
>> D*C

ans =

     0    11
     5    22
```

7. Find N^2



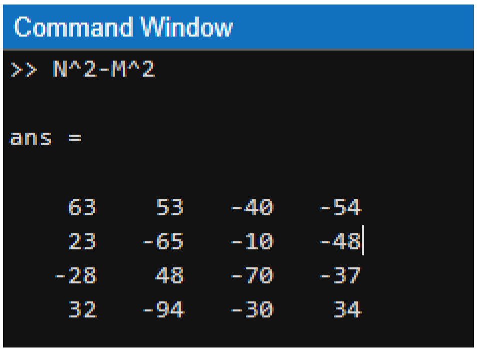
```
>> N^2

ans =

     4    88   -30     1
     0   -12   -14   -11
     0    18   -45    13
     0   -62   -11    -8
```

8. Find 3N – 2M



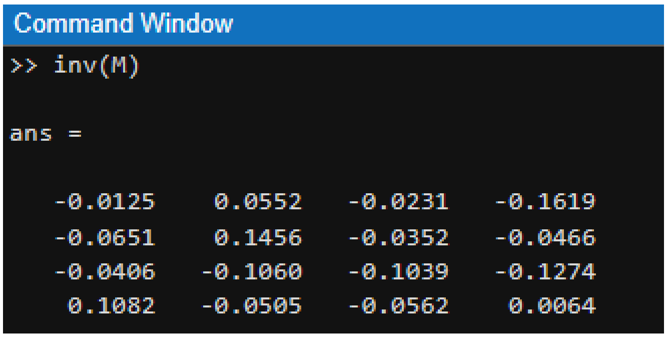
```
>> 3*N-2*M

ans =

    -2    16   -13   -26
    -2   -10    15   -10
   -12    -8    10     3
    14   -24    21     1
```

9. Find (N^2 - M^2)

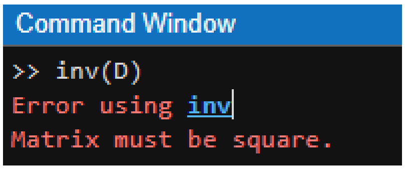


```
>> N^2-M^2

ans =

    63    53   -40   -54
    23   -65   -10   -48
   -28    48   -70   -37
    32   -94   -30    34
```

10. Find the inverse matrices of M and D.



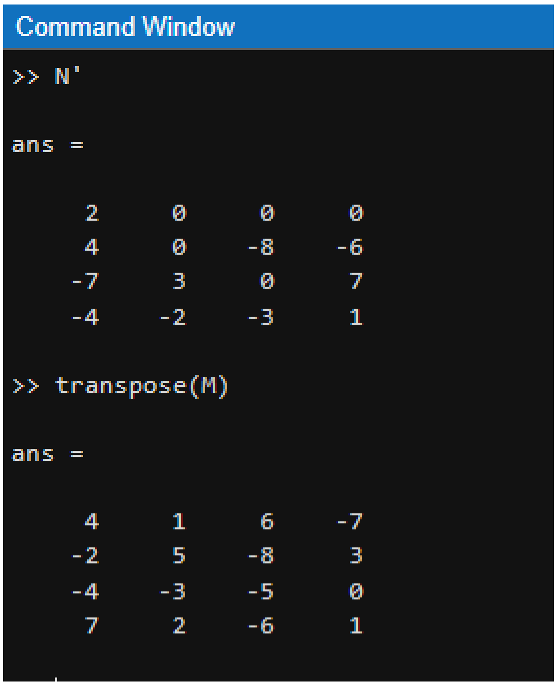
```
>> inv(M)

ans =

   -0.0125    0.0552   -0.0231   -0.1619
   -0.0651    0.1456   -0.0352   -0.0466
   -0.0406   -0.1060   -0.1039   -0.1274
    0.1082   -0.0505   -0.0562    0.0064
```



```
>> inv(D)
Error using inv
Matrix must be square.
```

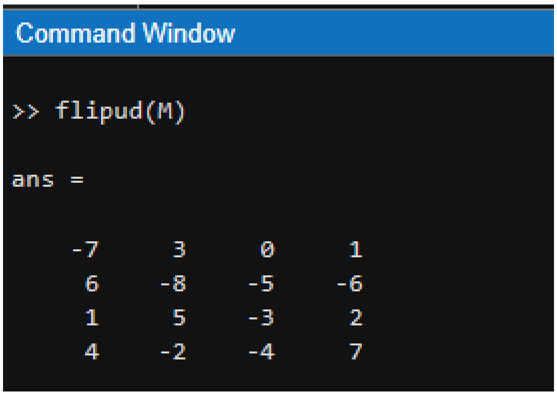
11. Find N^T and C^T

You can use both ways



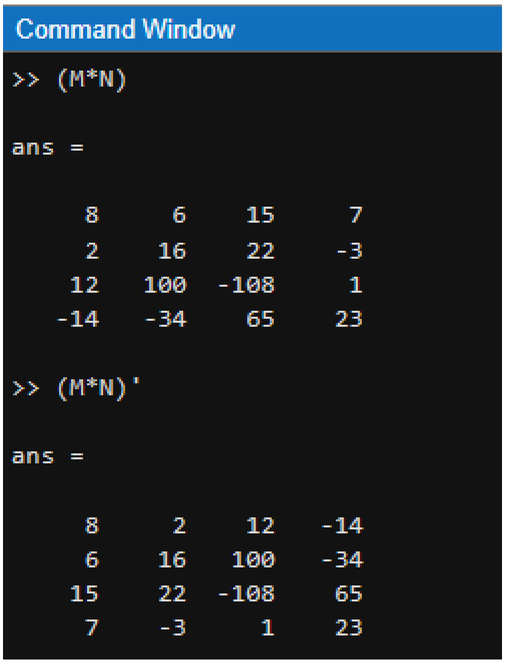
```
>> N'

ans =

     2     0     0     0
     4     0    -8    -6
    -7     3     0     7
    -4    -2    -3     1

>> transpose(M)

ans =

     4     1     6    -7
    -2     5    -8     3
    -4    -3    -5     0
     7     2    -6     1
```

12. Return M with the order of element flipped upside down along the first dimension.



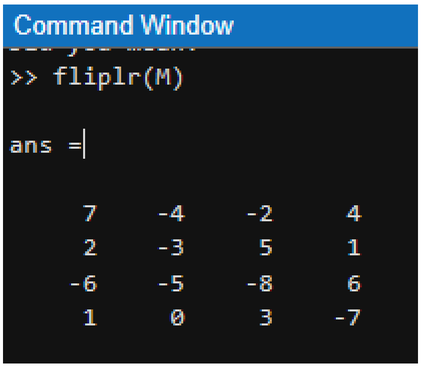
```
>> flipud(M)

ans =

    -7     3     0     1
     6    -8    -5    -6
     1     5    -3     2
     4    -2    -4     7
```

13. Find $(MN)^T$



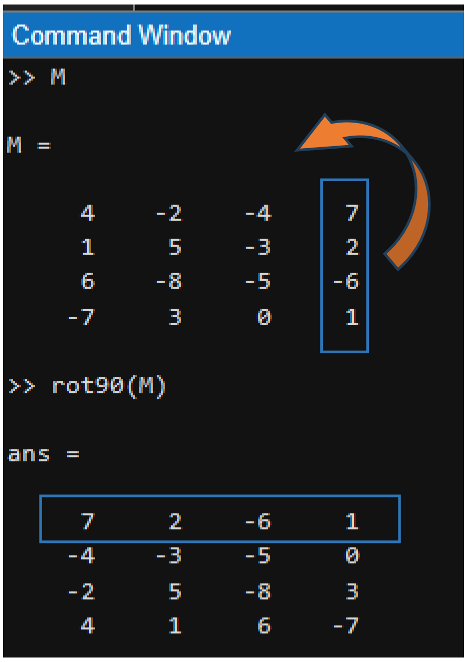
```
>> (M*N)

ans =

     8     6    15     7
     2    16    22    -3
    12   100  -108     1
   -14   -34    65    23

>> (M*N)'

ans =

     8     2    12   -14
     6    16   100   -34
    15    22  -108    65
     7    -3     1    23
```

14. Return M with the order of element fliplr flip left to right along the first dimension.



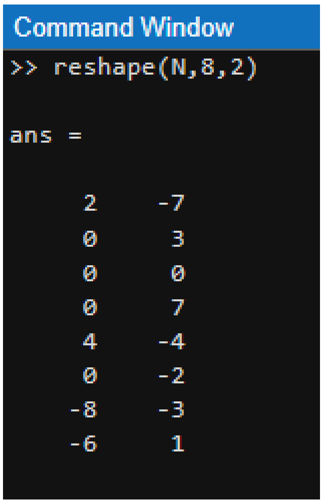
```
>> fliplr(M)

ans =

     7    -4    -2     4
     2    -3     5     1
    -6    -5    -8     6
     1     0     3    -7
```

15. Rotate matrix M in 90 degrees.



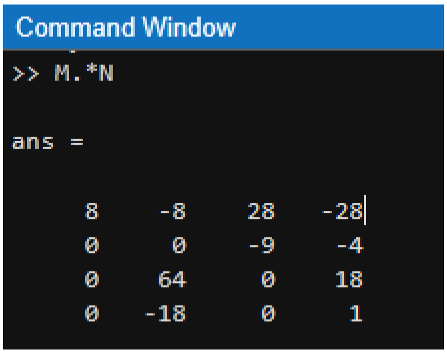
```
>> M

M =

     4    -2    -4     7
     1     5    -3     2
     6    -8    -5    -6
    -7     3     0     1

>> rot90(M)

ans =

     7     2    -6     1
    -4    -3    -5     0
    -2     5    -8     3
     4     1     6    -7
```

16. Reshape the matrix N in to (8 x 2)



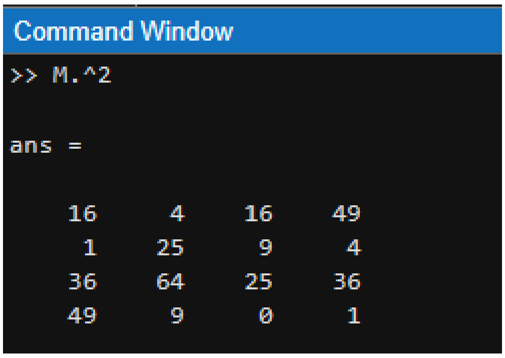
```
>> reshape(N,8,2)

ans =

     2    -7
     0     3
     0     0
     0     7
     4    -4
     0    -2
    -8    -3
    -6     1
```

17. Produce the matrix whose elements are the products of the corresponding elements of M and N.



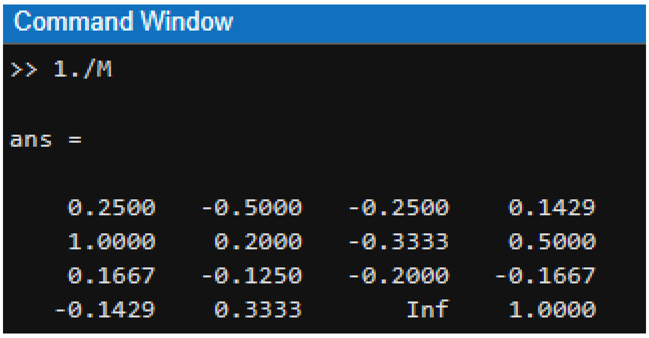
```
>> M.*N

ans =

     8    -8    28   -28
     0     0    -9    -4
     0    64     0    18
     0   -18     0     1
```

18. Produces a matrix each element of which is a square of the corresponding elements of M.



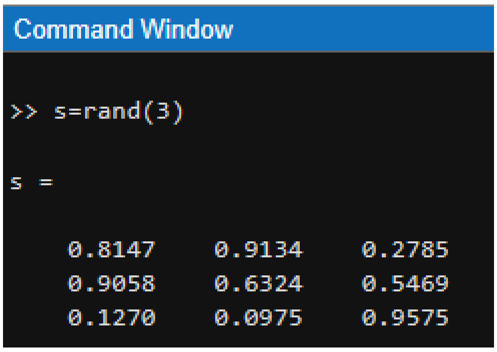
```
>> M.^2

ans =

    16     4    16    49
     1    25     9     4
    36    64    25    36
    49     9     0     1
```

19. Produce a matrix of reciprocals by writing 1./M

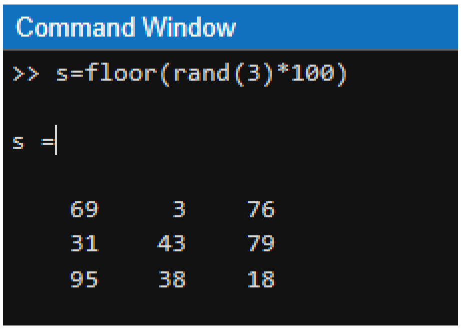


```
>> 1./M

ans =

    0.2500   -0.5000   -0.2500    0.1429
    1.0000    0.2000   -0.3333    0.5000
    0.1667   -0.1250   -0.2000   -0.1667
   -0.1429    0.3333       Inf    1.0000
```

20. Construct random integer matrices by multiplying the results of **rand** by 3. Then use the floor function to take the integer part of the result.

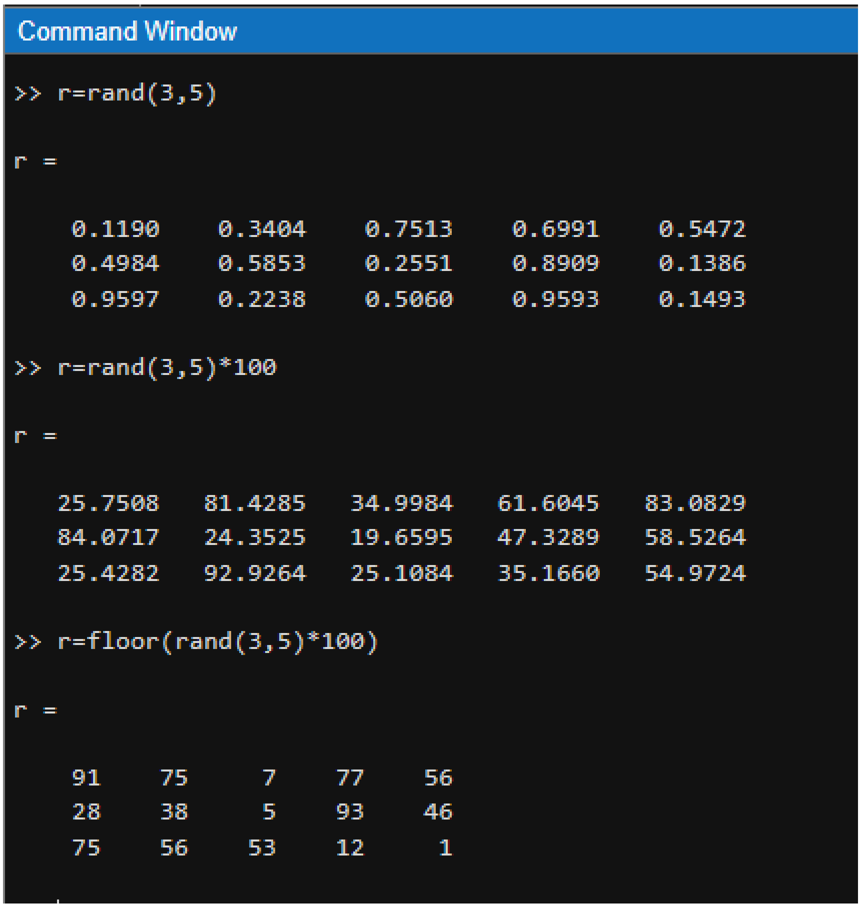


```
>> s=rand(3)

s =

    0.8147    0.9134    0.2785
    0.9058    0.6324    0.5469
    0.1270    0.0975    0.9575
```



```
>> s=floor(rand(3)*100)

s =

    69     3    76
    31    43    79
    95    38    18
```

21. Construct random integer matrices by multiplying the results of **randn** by 3 and 5. Then use the floor function to take the integer part of the result.



```
>> r=rand(3,5)

r =

    0.1190    0.3404    0.7513    0.6991    0.5472
    0.4984    0.5853    0.2551    0.8909    0.1386
    0.9597    0.2238    0.5060    0.9593    0.1493

>> r=rand(3,5)*100

r =

   25.7508   81.4285   34.9984   61.6045   83.0829
   84.0717   24.3525   19.6595   47.3289   58.5264
   25.4282   92.9264   25.1084   35.1660   54.9724

>> r=floor(rand(3,5)*100)

r =

    91    75     7    77    56
    28    38     5    93    46
    75    56    53    12     1
```

Exercise 03 (For You do your own)

1. Create the matrix X = [5 0 4 , 0 1 0, 1 2 9]
2. Find the inverse of X.
3. Find the transpose of X.
4. Use the command flipud and fliplr.
5. Rotate matrix X in 90 degrees.
6. Create the matrix P = [1 0 9 2, 3 0 12 4, -5 7 -7 6]
7. Reshape the matrix (2 * 6) and (4 * 3)