1. Create 5 matrices with five different dimensions (1-D,2-D,...5-D)

In [1]:
```
import pandas as pd
import numpy as np
from numpy import linalg as lg
```

In [2]:
```
a=np.array([[1]])
print(a)
b=np.array([[1,2],[2,3]])
c=np.array([[1,2,3],[4,5,6],[10,23,23]])
d=np.array([[1,2,3,4],[34,35,5,76],[12,42,36,47],[24,32,52,76]])
e=np.array([[1,2,3,4,5],[13,51,71,94,35],[15,26,38,49,50],[1,51,72,92,51],[2,3,5,76,54]
print(b)
print(c)
print(d)
print(e)
```

```
[[1]]
[[1 2]
 [2 3]]
[[ 1  2  3]
 [ 4  5  6]
 [10 23 23]]
[[ 1  2  3  4]
 [34 35  5 76]
 [12 42 36 47]
 [24 32 52 76]]
[[ 1  2  3  4  5]
 [13 51 71 94 35]
 [15 26 38 49 50]
 [ 1 51 72 92 51]
 [ 2  3  5 76 54]]
```

2. Find determinants of 5 matrices and display your output

In [3]:
```
print(lg.det(a))
print(lg.det(b))
print(lg.det(c))
print(lg.det(d))
print(lg.det(e))
```

```
1.0
-1.0
39.00000000000003
-14211.999999999965
32590.000000000142
```

3. Find inverse of the above 5 matrices and display your output

In [4]:
```
print(lg.inv(a))
```

```
[[1.]]
```

In [5]:
```
print(lg.inv(b))
```

```
[[-3.  2.]
 [ 2. -1.]]
```

In [6]:
```
print(lg.inv(c))
```

```
[[-0.58974359  0.58974359 -0.07692308]
 [-0.82051282 -0.17948718  0.15384615]
 [ 1.07692308 -0.07692308 -0.07692308]]
```

In [7]:
```
print(lg.inv(d))
```

```
[[-5.64396285e+00 -2.47678019e-02  8.35913313e-02  2.70123839e-01]
 [-5.08865747e-01  1.40726147e-03  5.20686744e-02 -6.82521813e-03]
 [-1.23332395e+00 -2.64565156e-02  2.11089220e-02  7.83141008e-02]
 [ 2.84041655e+00  2.53307064e-02 -6.27638615e-02 -1.22853926e-01]]
[[ 8.74808223e-02  4.62104940e-02  2.52838294e-02 -6.23810985e-02
  -2.54679349e-03]
 [-8.10244247e+01 -2.78631482e+00  7.64148512e+00  2.07143295e+00
   2.76403805e-01]
 [ 5.74794109e+01  1.97158638e+00 -5.41528076e+00 -1.45234735e+00
  -2.14237496e-01]
 [ 1.76376189e+00  7.93494937e-02 -1.83675974e-01 -5.53237189e-02
   7.57901197e-03]
 [-3.30638233e+00 -1.41147591e-01  3.34458423e-01  9.95704204e-02
   1.24271249e-02]]
```

In [8]:
```
print(lg.inv(e))
```

```
[[ 8.74808223e-02  4.62104940e-02  2.52838294e-02 -6.23810985e-02
  -2.54679349e-03]
 [-8.10244247e+01 -2.78631482e+00  7.64148512e+00  2.07143295e+00
   2.76403805e-01]
 [ 5.74794109e+01  1.97158638e+00 -5.41528076e+00 -1.45234735e+00
  -2.14237496e-01]
 [ 1.76376189e+00  7.93494937e-02 -1.83675974e-01 -5.53237189e-02
   7.57901197e-03]
 [-3.30638233e+00 -1.41147591e-01  3.34458423e-01  9.95704204e-02
   1.24271249e-02]]
```

4. Find the rank, diagonal and trace of the 5 matrices

In [10]:
```
print(lg.matrix_rank(a))
print(np.diag(a))
print(np.trace(a))
```

```
1
[1]
1
```

In [11]:
```
print(lg.matrix_rank(b))
print(np.diag(b))
print(np.trace(b))
```

```
2
[1 3]
4
```

In [13]:
```
print(lg.matrix_rank(c))
print(np.diag(c))
print(np.trace(c))
```

```
3
[ 1  5 23]
29
```

In [14]:
```
print(lg.matrix_rank(d))
print(np.diag(d))
print(np.trace(d))
```

```
4
[ 1 35 36 76]
148
```

In [15]:
```
print(lg.matrix_rank(e))
print(np.diag(e))
print(np.trace(e))
```

```
5
[ 1 51 38 92 54]
236
```

5. Find Eigen value and eigen vector for 5 matrices

In [17]:
```
print(lg.eig(a))
print(lg.eigvals(a))
```

```
(array([1.]), array([[1.]]))
[1.]
```

In [18]:
```
print(lg.eig(b))
print(lg.eigvals(b))
```

```
(array([-0.23606798,  4.23606798]), array([[-0.85065081, -0.52573111],
       [ 0.52573111, -0.85065081]]))
[-0.23606798  4.23606798]
```

In [19]:
```
print(lg.eig(c))
print(lg.eigvals(c))
```

```
(array([30.13790432+0.j        , -0.56895216+0.98506088j,
       -0.56895216-0.98506088j]), array([[-0.11604763+0.j       ,  0.253292  +0.43476j
,
```

```
         0.253292  -0.43476j  ],
       [-0.24802607+0.j       ,  0.5425542 -0.2163048j,
         0.5425542 +0.2163048j],
       [-0.96177753+0.j       , -0.6369255 +0.j       ,
        -0.6369255 -0.j       ]]))
[30.13790432+0.j         -0.56895216+0.98506088j -0.56895216-0.98506088j]
```

In [20]:

```
print(lg.eig(d))
print(lg.eigvals(d))
```

```
(array([141.35899153 +0.j        ,  -0.17351665 +0.j        ,
         3.40726256+23.82869563j,   3.40726256-23.82869563j]), array([[-0.03779557+0.j
,  0.87455569+0.j        ,
        -0.01970461+0.00506148j, -0.01970461-0.00506148j],
       [-0.52155391+0.j        ,  0.07494112+0.j        ,
         0.74818542+0.j        ,  0.74818542-0.j        ],
       [-0.51515505+0.j        ,  0.19275512+0.j        ,
        -0.16647283-0.50665115j, -0.16647283+0.50665115j],
       [-0.67909373+0.j        , -0.43861331+0.j        ,
        -0.29124874+0.26565063j, -0.29124874-0.26565063j]]))
[141.35899153 +0.j          -0.17351665 +0.j
   3.40726256+23.82869563j   3.40726256-23.82869563j]
```

In [21]:

```
print(lg.eig(e))
print(lg.eigvals(e))
```

```
(array([ 2.10444516e+02 +0.j       ,  1.44928297e+01+12.7261968j,
        1.44928297e+01-12.7261968j, -1.25991626e-01 +0.j       ,
       -3.30418405e+00 +0.j       ]), array([[-0.03078259+0.j        ,  0.06280806-0.016
79185j,
         0.06280806+0.01679185j,  0.0030276 +0.j        ,
         0.07825903+0.j        ],
       [-0.60593844+0.j        , -0.66527711+0.j        ,
        -0.66527711-0.j        , -0.81534525+0.j        ,
        -0.81654431+0.j        ],
       [-0.36648874+0.j        ,  0.47556578-0.16888978j,
         0.47556578+0.16888978j,  0.57735112+0.j        ,
         0.54017958+0.j        ],
       [-0.62486939+0.j        , -0.25832382-0.05777016j,
        -0.25832382+0.05777016j,  0.02102197+0.j        ,
         0.10972265+0.j        ],
       [-0.32728471+0.j        ,  0.3996614 +0.26209761j,
         0.3996614 -0.26209761j, -0.03777195+0.j        ,
        -0.1526364 +0.j        ]]))
[ 2.10444516e+02 +0.j         1.44928297e+01+12.7261968j
  1.44928297e+01-12.7261968j -1.25991626e-01 +0.j
 -3.30418405e+00 +0.j       ]
```

In [ ]: