```
#Eigenvectors are widely used in Machine Learning libraries. Intutively given a
linear transformation represented by a matrix,A, eigenvectors are vectors that
when that transformation is applied, change only in scale(not direction).
More formally
Av=Kv
Here A is a square matrix, K contains the eigenvalues and v contains the
eigenvectors.
```

Q1. FINDING THE VALUES OF EIGNVECTOR ABD EIGNVALUES

In [1]:

```python
#Load Library
import numpy as np

#Create a Matrix
matrix = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(matrix)

# Calculate the Eigenvalues and Eigenvectors of that Matrix
eigenvalues ,eigenvectors=np.linalg.eig(matrix)
print(eigenvalues)
print(eigenvectors)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[ 1.61168440e+01 -1.11684397e+00 -3.38433605e-16]
[[-0.23197069 -0.78583024  0.40824829]
 [-0.52532209 -0.08675134 -0.81649658]
 [-0.8186735   0.61232756  0.40824829]]
```

Q2. FINDING DOT PRODUCT

In [3]:

```python
#Load Library
import numpy as np

#Create vector-1
vector_1 = np.array([ 1,2,3 ])

#Create vector-2
vector_2 = np.array([ 4,5,6 ])

#Calculate Dot Product
print(np.dot(vector_1,vector_2))

#Alternatively you can use @ to calculate dot products
print(vector_1 @ vector_2)
```

```
32
32
```

Q3. ADDING, SUBTRACTING AND MULTIPLICATING MATRICES

In [4]:

```
#Load Library
import numpy as np

#Create Matrix-1
matrix_1 = np.array([[1,2,3],[4,5,6],[7,8,9]])

#Create Matrix-2
matrix_2 = np.array([[7,8,9],[4,5,6],[1,2,3]])

#Add the 2 Matrices
print(np.add(matrix_1,matrix_2))


#Subtraction
print(np.subtract(matrix_1,matrix_2))

#Multiplication(Element wise, not Dot Product)
print(matrix_1*matrix_2)
```

```
[[ 8 10 12]
 [ 8 10 12]
 [ 8 10 12]]
[[-6 -6 -6]
 [ 0  0  0]
 [ 6  6  6]]
[[ 7 16 27]
 [16 25 36]
 [ 7 16 27]]
```

Q4 CALCULATING THE INVERT MATRIX

In [5]:

```
#Load Library
import numpy as np

#Create a Matrix
matrix = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(matrix)

#Calculate its inverse
print(np.linalg.inv(matrix))
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[ 3.15251974e+15 -6.30503948e+15  3.15251974e+15]
 [-6.30503948e+15  1.26100790e+16 -6.30503948e+15]
 [ 3.15251974e+15 -6.30503948e+15  3.15251974e+15]]
```

Q5. GENERATING RANDOM VALUES

In [6]:

```
#Load Library
import numpy as np

#Set seed
np.random.seed(1)

#Generate 3 random integers b/w 1 and 10
print(np.random.randint(0,11,3))

#Draw 3 numbers from a normal distribution with mean 1.0 and std 2.0
print(np.random.normal(1.0,2.0,3))
```

```
[5 8 9]
[-0.60434568  0.10224438 -1.21187015]
```