Importing libraries

In [1]:
```
import numpy as np
from scipy import linalg as la
```

Defining the inputs

In [2]:
```
A = np.array([[0,3,-1],
              [-1,4,-2],
              [1,3,1]])

B = np.array([[2,-1,2],
              [-1,0,1],
              [-1,2,2]])


v = np.array([2,-1,4])
u = np.array([-2,1,5])

a = -2
b = 1
```

Question 1 1) Calculate

> **To solve:** $\|v\|_1 v + au$
>
> **Given:** v = [2 -1 4], u = [-2 1 5], a = -2
>
> $\|v\|_1$ = This is the L1 norm of vector v
>
> $\|v\|_1 = \sqrt[1]{}\ (|2| + |-1| + |4|) = \sqrt[1]{}\ (2 + 1 + 4) = 7$
>
> $\|v\|_1 v$ = scalar multiplication of 7 * vector v = 7 * [2 -1 4] = [14 -7 28]
>
> au = scalar multiplication of a * vector u = -2 * [-2 1 5] = [4 -2 -10]
>
> $\|v\|_1 v + au$ = [14 -7 28] + [4 -2 -10] (vector addition) = **[18 -9 18]**

2) Python Validation

In [3]:
```
#L1 norm of vector v
v_norm = la.norm(v,1)

#scalar multiplication v_norm * vector v
v_norm_times_v = v_norm*v

#scalar multiplicaition a * vector u
a_times_u = a*u

#sum of the entities
result = v_norm_times_v + a_times_u
print("Result :", result)
```

```
Result : [18. -9. 18.]
```

Question 2 1) Calculate

To solve: **Using the cosine formula, and assuming the angle between vectors v and u is equal to θ, calculate cos θ.**

Cosine formula : **cos θ = v.u / (||v|| ||u||)**

v.u = scalar product of vectors v and u = (2*-2)+(-1*1)+(4*5) = -4-1+20 = 15

||v|| = √ 4+1+16 = √21

||u|| = √ 4+1+25 = √30

cos θ = 15/(√21 * √30) = 15/√630 = 5/√70 = **0.59761430504**

2) Python Validation

In [4]:
```
#scalar product of vectors v and u
v_dot_u = np.dot(v,u)

#vector norm of vector v
v_norm = la.norm(v,2)

#vector norm of vector u
u_norm = la.norm(u,2)

#cosθ = v_dot_u /(v_norm*u_norm)
Cosine_Theta = v_dot_u / (v_norm*u_norm)

print('Cosine Theta : ', Cosine_Theta)
print('Angle  [rad] : ',round(np.arccos(Cosine_Theta)/np.pi,3),'pi')
```

```
Cosine Theta :  0.5976143046671969
Angle  [rad] :  0.296 pi
```

Question 3 1) Calculate

To solve: $a(\boldsymbol{A}{\cdot}v)$

Given: a = -2, v = [2 -1 4], A=[[0 3 -1] [-1 4 -2] [1 3 1]]

Since A is a 3x3 matrix and v is a vector with shape 3x1, matrix multiplication is possible. The result of A.v is of the shape 3x1, i.e., 3 rows and 1 column

A.v = [(0*2) + (3* -1) + (-1 * 4) (-1* 2) + (4* -1) + (-2 * 4) (1*2) + (3* -1) + (1* 4)] = [-7 -14 3]

$a(\boldsymbol{A}{\cdot}v)$ = scalar multiplication of a * vector A.v = -2 * [-7 -14 3] = **[14 28 -6]**

2) Python Validation

In [5]:
```
#dot product of matrix A and vector v
A_dot_v = A.dot(v)

#scalar multiplication of a and vector A.v
a_times_A_dot_v = a * A_dot_v

print("Result :", a_times_A_dot_v)
```

```
Result : [14 28 -6]
```

Question 4 1) Calculate

Q:4

To solve: Let $tr(\boldsymbol{B})$ and $\boldsymbol{L}$ be the trace and lower triangular matrix of matrix $\boldsymbol{B}$, respectively. Calculate $\boldsymbol{A}\cdot\boldsymbol{B}^T + tr(\boldsymbol{B}) * \boldsymbol{L}$

Given: A=[[0 3 -1] [-1 4 -2] [1 3 1]], B=[[2 -1 2] [-1 0 1] [-1 2 2]]

$\boldsymbol{B}^T$ = Transpose of matrix B =[[2 -1 -1] [-1 0 2] [2 1 2]]

$\boldsymbol{A}\cdot\boldsymbol{B}^T$ = Dot product of matrix A and Transpose of matrix B results in a 3x3 matrix

= [[(0*2 + 3*-1 + -1*2) (0*-1 + 3*0 + -1*1) (0*-1 + 3*2 + -1*2)]
[(-1*2 + 4*-1 + -2*2) (-1*-1 + 4*0 + -2*1) (-1*-1 + 4*2 + -2*2)]
[(1*2 + 3*-1 + 1*2) (1*-1 + 3*0 + 1*1) (1*-1 + 3*2 + 1*2)]]

= [[-5 -1 4]
[-10 -1 5]
[1 0 7]]

$tr(\boldsymbol{B})$ = Trace of matrix B = Sum of diagonal elements = 2+0+2 = 4

$\boldsymbol{L}$ = lower triangular matrix of matrix $\boldsymbol{B}$ = All elements above the main diagonal are 0

$\boldsymbol{L}$ = [[2 0 0]
[-1 0 0]
[-1 2 2]]

$tr(\boldsymbol{B}) * \boldsymbol{L}$ = trace of matrix B times the lower triangular matrix L

= 4 * [[2 0 0]
[-1 0 0]
[-1 2 2]]

= [[8 0 0]
[-4 0 0]
[-4 8 8]]

$\boldsymbol{A}\cdot\boldsymbol{B}^T + tr(\boldsymbol{B}) * \boldsymbol{L}$ = sum of results from previous steps

= [[-5 -1 4]
[-10 -1 5]
[1 0 7]] + [[8 0 0]
[-4 0 0]
[-4 8 8]]

**=[[3 -1 4]
[-14 -1 5]
[-3 8 15]]**

2) Python Validation

In [6]:
```
#A·BT +tr(B)*L

#transpose of matrix B
BTrans = B.T

#dot product of matrix A and transpose of matric B
A_dot_BTrans = A.dot(BTrans)

#trace of matrix B ( sum of diagonal elements)
BTrace = B.trace()

#lower triangular matrix of matrix B
L = la.tril(B)

#trace(B) times lower triangular matrix L
BTrace_times_L = BTrace*L
```

```
#final result
result = A_dot_BTrans + BTrace_times_L

print("Result :")
print(result,'\n')
```

```
Result :
[[  3  -1   4]
 [-14  -1   5]
 [ -3   8  15]]
```