In [10]:

```
#SP1

import numpy as np
from math import *

A = [ 7.2, -4.3, 0.6, 1.7]

B = [-11.0, 11.8, 2.4, -1.9]

dot = 0

for A,B in zip(A,B):
    dot += A*B

print(f"The dot product of A and B is {dot}")
```

```
The dot product of A and B is -131.73
```

In [31]:

```
#SP2

import numpy as np
from math import *

A = [ 7.2, -4.3, 0.6]

B = [-11.0, 11.8, 2.4]

def cross_product(A,B):

    Cross =    [A[1]*B[2] - A[2]*B[1],
               A[2]*B[0] - B[2]*A[0],
               A[0]*B[1] - A[1]*B[0]]

    return Cross

cross_product(A,B)
```

Out[31]:

```
[-17.4, -23.880000000000003, 37.66000000000001]
```

In [25]:

```
#SP3

import numpy as np
from math import *


A = [[7.2, -4.3],
     [3.5, 6.7]]


B = [[9.2, -4.3],
     [3.5, 10.7]]

R = [[0, 0],
     [0,0]]


for i in range(len(A)):
        for j in range(len(B)):
                   for k in range(len(R)):
                       R[i][j] += A[i][k] * B[k][j]


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

```
[51.19, -76.97]
[55.64999999999999, 56.64]
2
```

In [1]:

```
#SP4

import numpy as np
from math import *
```

```
A = [[1,2,3],
     [3,1,-3],
     [-3,4,7]]

A_x = [[-5,2,3],
       [4,1,-3],
       [-7,4,7]]


A_y = [[1,-5,3],
       [3,4,-3],
       [-3,-7,7]]

A_z = [[1,2,-5],
     [3,1,4],
     [-3,4,-7]]


def det_A():
    det1 = ((A[0][0] * A[1][1] * A[2][2]) + (A[0][1] * A[1][2] * A[2][0]) + (A[0][2] * A[1][0] * A[2][1])) - ((A
    return det1


def det_Ax():
    det2 = ((A_x[0][0] * A_x[1][1] * A_x[2][2]) + (A_x[0][1] * A_x[1][2] * A_x[2][0]) + (A_x[0][2] * A_x[1][0] *
    return det2


def det_Ay():
    det3 = ((A_y[0][0] * A_y[1][1] * A_y[2][2]) + (A_y[0][1] * A_y[1][2] * A_y[2][0]) + (A_y[0][2] * A_y[1][0] *
    return det3



def det_Az():
    det4 = ((A_z[0][0] * A_z[1][1] * A_z[2][2]) + (A_z[0][1] * A_z[1][2] * A_z[2][0]) + (A_z[0][2] * A_z[1][0] *
    return det4


x = det_Ax()/det_A()
y = det_Ay()/det_A()
z = det_Az()/det_A()


print("x =", x)
print("y =", y)
print("z =", z)
```

```
x = -1.0
y = 1.0
z = -2.0
```

In [44]:

```
# SP5

import numpy as np
from math import *

A =  np.array([[1,2,3,3],[0,1,-3,5],[0,0,1,8],[0,0,0,7]])

b = np.array([-5,4,7,-7])

n = 4

x = np.zeros(n)

x[3]= b[n-1]/A[n-1][n-1]
x3 = x[3]

x = np.append(x,x3)


for i in range(2,-1,-1):
    x[i] = (b[i] - get_sum(A,x,i))/A[i][i]
    x = np.append(x,x[i])


def get_sum(A,x,i):
            summ = 0
            for j in range(i+1,n,1):
                summ += A[i][j]*x[j]
            return summ
```

```
print("x_3= ",x[3])
print("x_2=",x[2])
print("x_1= ",x[1])
print("x_0= ",x[0])
```

```
x_3=  -1.0
x_2= 15.0
x_1=  54.0
x_0=  -155.0
```

In [33]:

```
#SP6

import numpy as np
from math import *

def add_mat(X,Y,Z):
    for i in range(len(A)):
        for j in range(len(B)):
                Z[i][j] = X[i][j] + Y[i][j]
    return Z

def sub_mat(X,Y,Z):
    for i in range(len(A)):
        for j in range(len(B)):
                Z[i][j] = X[i][j] - Y[i][j]
    return Z


A = [[7.2, -4.3],
     [0.6, 1.7]]
B = [[-11.0, 11.8],
     [2.4, -1.9]]
C_1 = [[0,0],
     [0,0]]

C_2 = [[0,0],
     [0,0]]


print("A + B =", add_mat(A,B,C_1))
print("A - B =", sub_mat(A,B,C_2))
```

```
A + B = [[-3.8, 7.500000000000001], [3.0, -0.19999999999999996]]
A - B = [[18.2, -16.1], [-1.7999999999999998, 3.5999999999999996]]
```

In [10]:

```
[[1,2,3,3],
 [0,1,-3,5],
 [0,0,1,8],
 [0,0,0,7]]

[-5,4,7,-7]
```

```
0
```

In [ ]:

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js