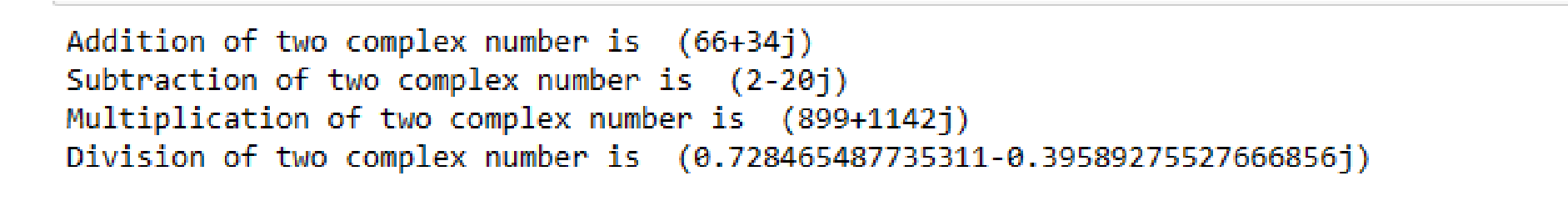
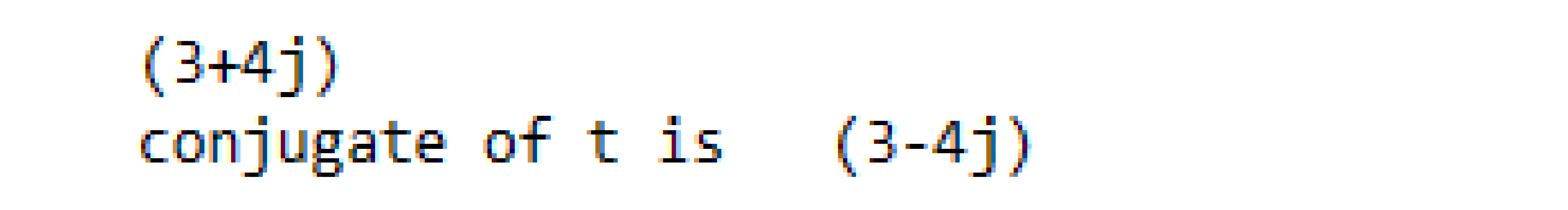
# Practical 1\_1

c1=34+7j c2=32+27j c1+c2 print("Addition of two complex number is ", c1+c2) print("Subtraction of two complex number is ", c1-c2) print("Multiplication of two complex number is ", c1\*c2) print("Division of two complex number is ", c1/c2) Output



# Practical 1\_2

t=3+4j print(t) m=t.conjugate() print("conjugate of t is ",m)

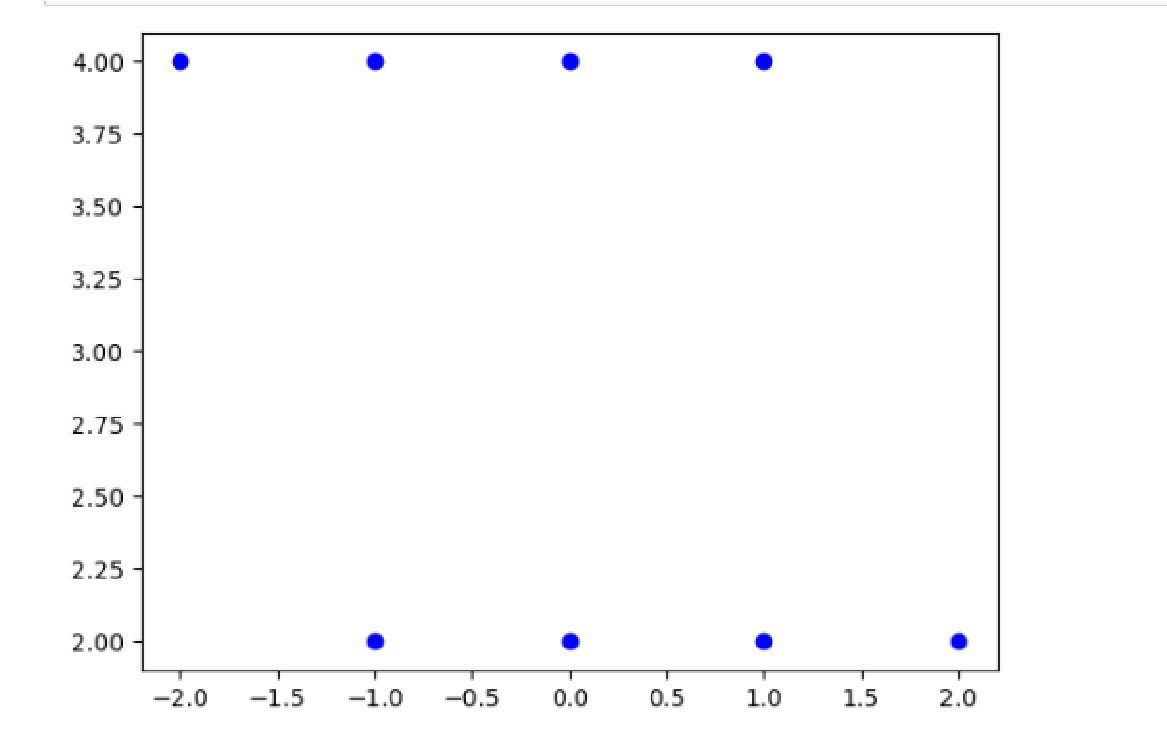
Output

# Practical 1\_3

import matplotlib.pyplot as plt x=3+2j

a=[-2+4j,-1+2j,0+2j,1+2j,2+2j,-1+4j,0+4j,1+4j] A=[x.real for x in a] B=[x.imag for x in a] plt.scatter(A,B,color="blue") plt.show()

Output:-



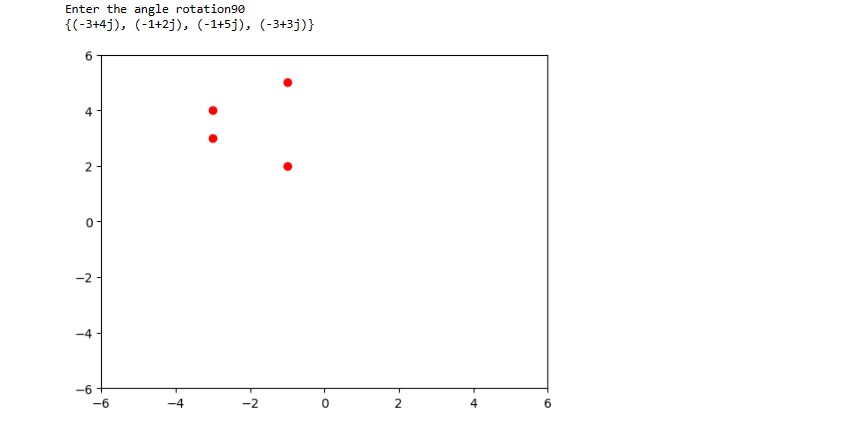
# Practical 1\_4

import matplotlib.pyplot as plt s={3+3j,4+3j,2+1j,5+1j,2+1j} angle=int(input("Enter the angle rotation")) if angle==90: s1={x\*1j for x in s} print(s1) x=[x.real for x in s1] y=[x.imag for x in s1] plt.plot(x,y,'ro') plt.axis([-6,6,-6,6]) plt.show() elif angle==180: s1={x\*-1 for x in s1} print(s1) x=[x.real for x in s1] y=[x.imag for x in s1] plt.plot(x,y,'ro') plt.axis([-6,6,-6,6])

plt.show() else:

print("invalid angle")

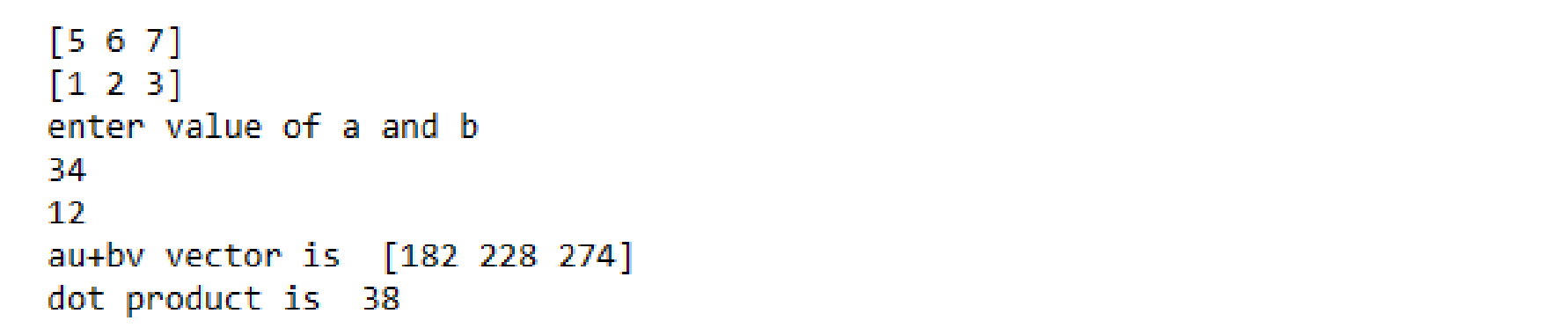
Output



# Practical 2

import numpy as np #enter vector as n-list x=np.array([5,6,7]) y=np.array([1,2,3]) print(x) print(y) print("enter value of a and b") a=int(input()) b=int(input()) c=a\*x+b\*y d=np.dot(x,y) print("au+bv vector is " , c) print("dot product is ",d)

Output



# Practical 3\_1

import numpy as np M=np.array([[1,1,1],[3,4,7],[9,6,3]])

M

print("matrix M is ",M) Y=M[0:1]

Y

print("first row of matrix M is ",Y)

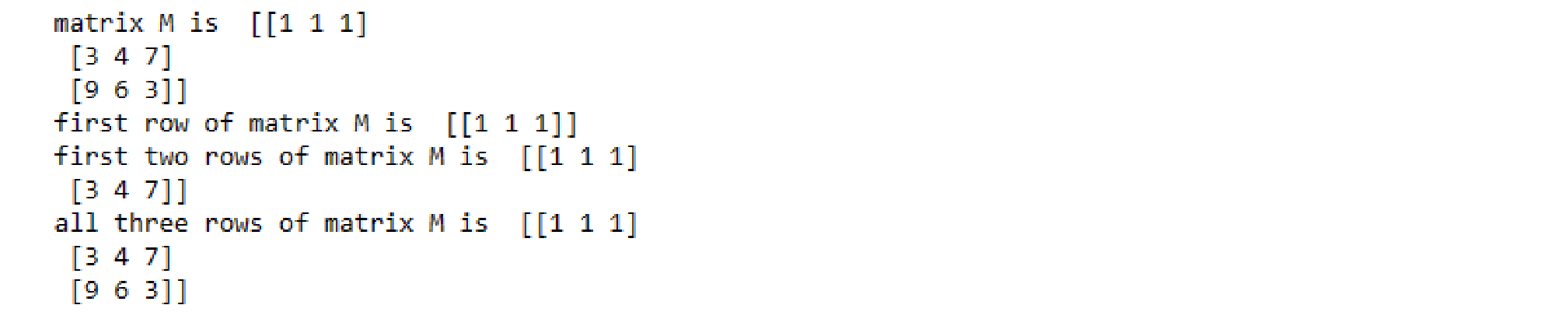
X=M[0:2]

print("first two rows of matrix M are:- ",x)

t=M[0:3]

print("all three rows of matrix M are:-,t)

**Output:-**



# Practical 3\_2

import numpy as np M=np.array([[1,1,1],[3,4,7],[9,6,3]])

M

print("matrix M is:-",M)

Y=M[0:1]

Y

print("first column of matrix M is :-",Y) X=M[0:2]

print("first two columns of matrix M are:- ",x)

t=M[0:3]

print("all three columns of matrix M are ",t)

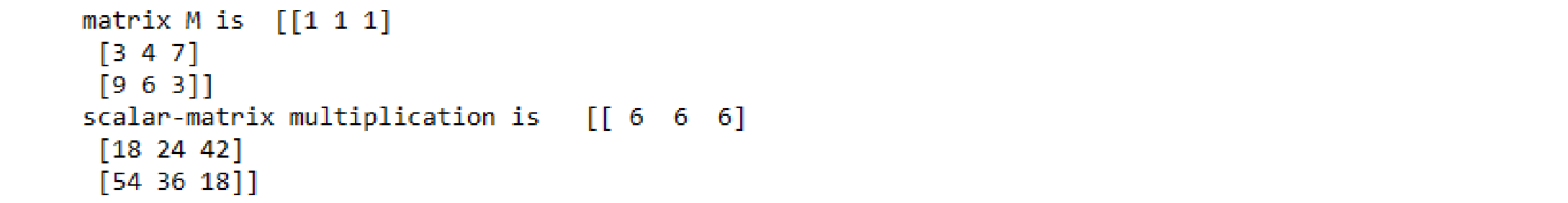
**OUTPUT:-**



# Practical 3\_3

import numpy as np M=np.array([[1,1,1],[3,4,7],[9,6,3]])

M #matrix M is print("matrix M is ",M) a=6 scalar=a\*M

print("scalar-matrix multiplication is ",scalar) **Output:-**

# Practical 3\_4

TRANSPOSE OF THE MATRIX M

x=[[12,7],[4,5],[3,8]] t=[[0,0,0],[0,0,0]] print("original matrix") print(x) print("transpose of matrix") for i in range(len(x)):

for j in range(len(x[0])): t[j][i]=x[i][j]

for r in t: print(r)

**Output:-**

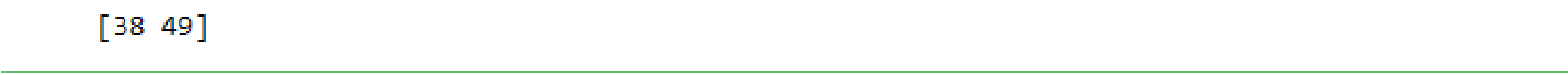


# Practical 4\_1

Vector matrix multiplication of a rby c matrix with an c-vector u.

**CODE:-**

import numpy as np x=np.array([1,4,6]) y=np.array([[2,3],[3,4],[4,5]]) print(np.dot(x,y))

**Output**

# Practical 4\_2

Find the matrix-matrix product of M with a c by p matrix N

**CODE:-**

import numpy as np A=np.array([[3,2,2],[4,1,5],[1,2,3]]) print("matrix A is ",A) B=np.array([[1,2,3],[1,1,1],[2,2,2]]) print("matrix B is ",B)

print("multiplication of two matrices A & B is ") M=([[0,0,0],[0,0,0],[0,0,0]]) for i in range(len(A)):

for j in range(len(B[0])):

for k in range(len(B)):

M[i][j]+=A[i][k]\*B[k][] for r in M: print(r)

**OUTPUT:-**

matrix A is [[3 2 2]

[4 1 5]

[1 2 3]] matrix B is [[1 2 3]

1. 1 1]
2. 2 2]] multiplication of two matrices A & B is [3, 0, 0]

[0, 0, 0]

[0, 0, 0]

[5, 0, 0]

[0, 0, 0]

[0, 0, 0]

[9, 0, 0]

[0, 0, 0]

[0, 0, 0]

[9, 6, 0]

[0, 0, 0]

[0, 0, 0]

[9, 8, 0]

[0, 0, 0]

[0, 0, 0]

[9, 12, 0]

[0, 0, 0]

[0, 0, 0]

[9, 12, 9]

[0, 0, 0]

[0, 0, 0]

[9, 12, 11]

[0, 0, 0]

[0, 0, 0]

[9, 12, 15]

[0, 0, 0]

[0, 0, 0]

[9, 12, 15]

[4, 0, 0]

[0, 0, 0]

[9, 12, 15]

[5, 0, 0]

[0, 0, 0]

[9, 12, 15]

[15, 0, 0]

[0, 0, 0]

[9, 12, 15]

[15, 8, 0]

[0, 0, 0]

[9, 12, 15]

[15, 9, 0]

[0, 0, 0]

[9, 12, 15]

[15, 19, 0]

[0, 0, 0] [9, 12, 15]

[15, 19, 12]

[0, 0, 0] [9, 12, 15]

[15, 19, 13]

[0, 0, 0] [9, 12, 15]

[15, 19, 23]

[0, 0, 0] [9, 12, 15]

[15, 19, 23]

[1, 0, 0] [9, 12, 15]

[15, 19, 23]

[3, 0, 0] [9, 12, 15]

[15, 19, 23]

[9, 0, 0] [9, 12, 15]

[15, 19, 23]

[9, 2, 0] [9, 12, 15]

[15, 19, 23]

[9, 4, 0] [9, 12, 15]

[15, 19, 23]

[9, 10, 0] [9, 12, 15]

[15, 19, 23]

[9, 10, 3] [9, 12, 15]

[15, 19, 23]

[9, 10, 5] [9, 12, 15]

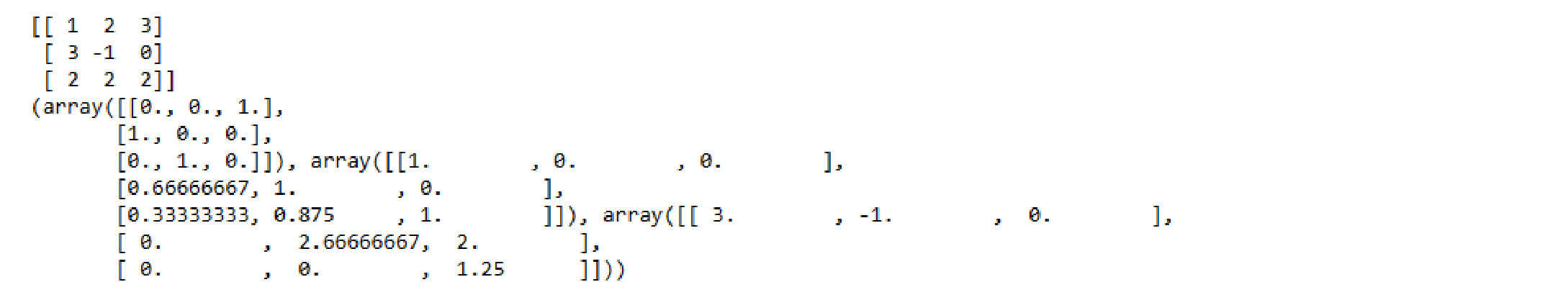
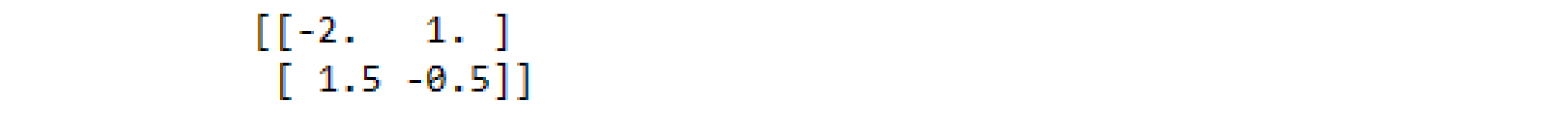
[15, 19, 23]

[9, 10, 11]

# Practical 5

import numpy as np from numpy.linalg import inv a=np.array([[1,2],[3,4]]) b=inv(a) print(b) Output

# Practical 6

from scipy.linalg import lu import numpy as np M=np.array([[1,2,3],[3,-1,0],[2,2,2]]) u=lu(M) print(M) print(u) Output

# Practical 7\_1

N=54

print(N)

a=9

b=6

print("factors of N are a and b",b) x=(a+b)/2

y=(a-b)/2

print("x and y is " ,,y)

a1=x\*x

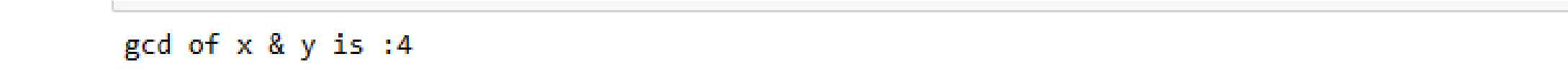
b1=y\*y

print("a1 and b1 is " ,a1,b1) N=a1-b1

print(N)

Output

# Practical 7\_2

import math print("gcd of x & y is :",end="") print(math.gcd(12,16)) Output

# Practical 8

import numpy as np

def oprojection (of\_vec,on\_vec):

x1=np.array(of\_vec)

x2=np.array(of\_vec)

scal=np.dot(x2,x1)/np.dot(x1,x2)

vec=scal\*2

return round(scal,10),np.around(vec,decimals=10) print(oprojection([2.0,2.0],[1.0,0.0])) print(oprojection([2.0,2.0],[6.0,2.0]))

**Output**

# 

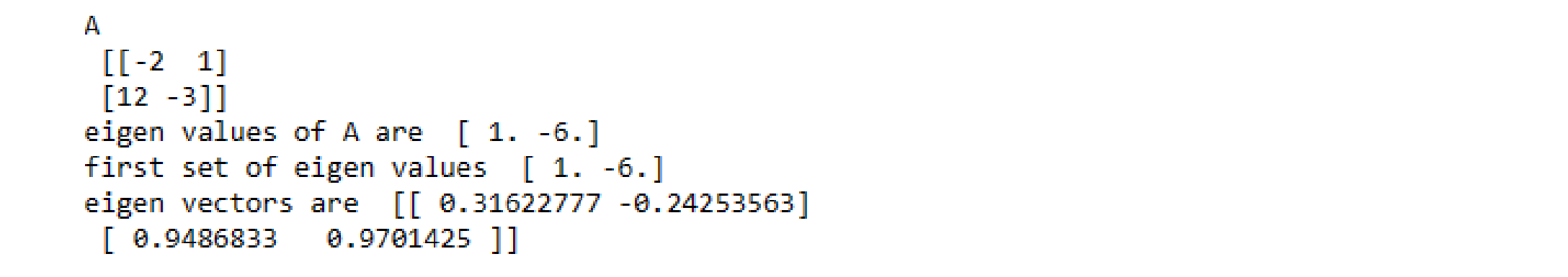
# Practical 9

import numpy as np

A=np.mat("-2 1;12 -3")

print("A \n",A)

print("eigen values of A are",np.linalg.eigvals(A)) eigenvalues,eigenvectors=np.linalg.eig(A) print("first set of eigen values ",eigenvalues) print("eigen vectors are ",eigenvectors)

**Output:-**