**Name – Ujjawal Mandhani**

**Batch – F6**

**Enroll No.-9918103237**

**Lab 6 OSS**

**Question 1:**Import the essential library scipy with i/o packageand Numpy. Create a 4 x 4 dimensionalone’s array. Store array in test.text file. Get datafrom test.text file and print the output

**import numpy as np**

**import scipy**

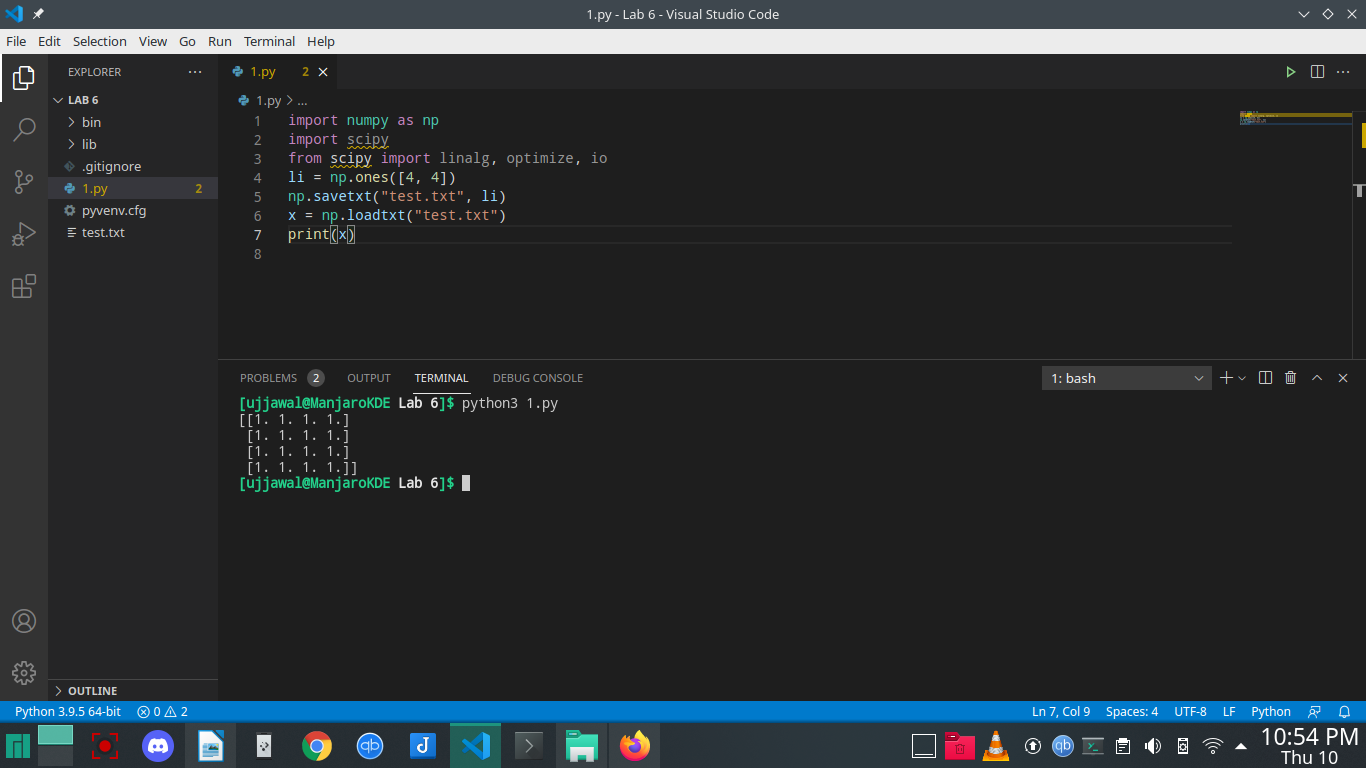
**from scipy import linalg, optimize, io**

**li = np.ones([4, 4])**

**np.savetxt("test.txt", li)**

**x = np.loadtxt("test.txt")**

**print(x)**



**Question 2:**Find cubic root of 27, 64, 891 using sciPy specialpackage

**import numpy as np**

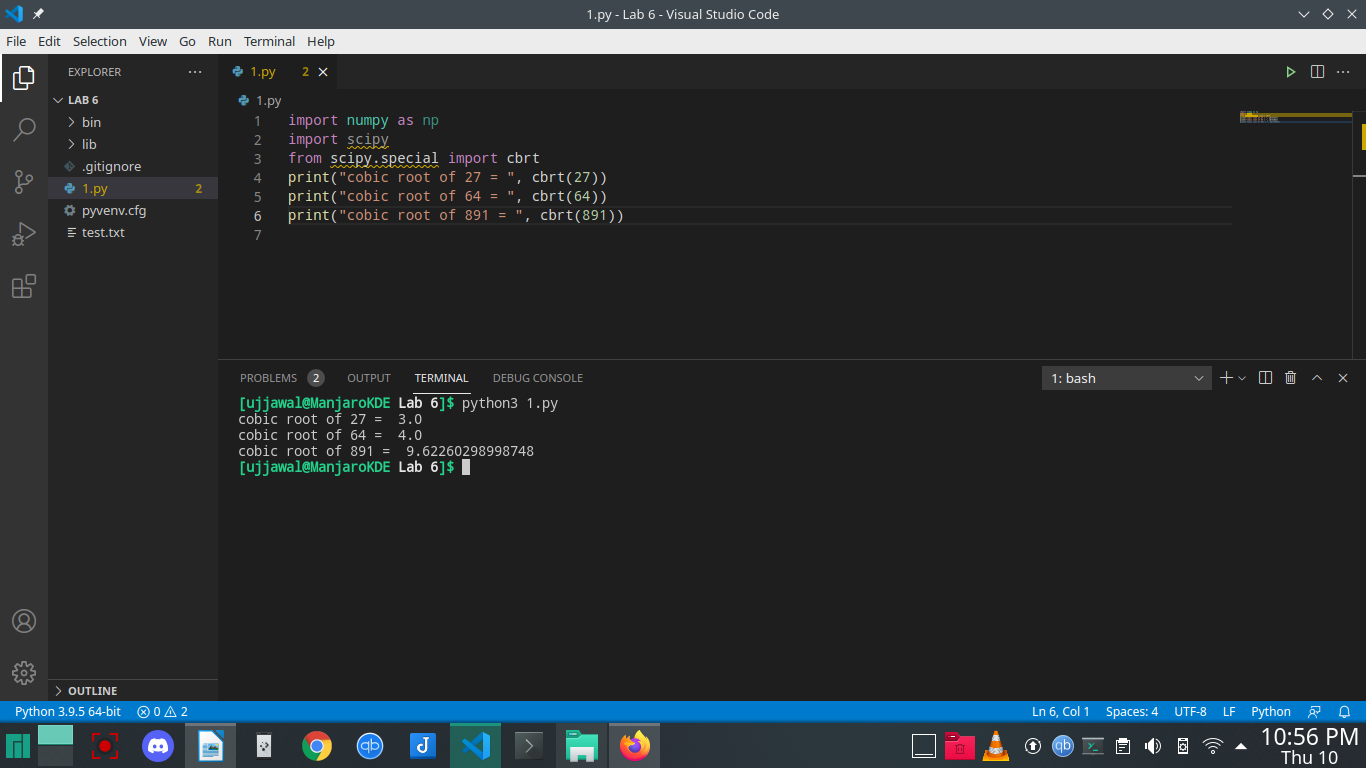
**import scipy**

**from scipy.special import cbrt**

**print("cobic root of 27 = ", cbrt(27))**

**print("cobic root of 64 = ", cbrt(64))**

**print("cobic root of 891 = ", cbrt(891))**



**Question 3:**Create two matrices with 2x2 dimensions. Initialize them with values [4,5], [3,2]. Calculatedeterminant of a two-dimensional matrix using scipy.linalg.Calculate the inverse of a matrixin 3.

**import numpy as np**

**import scipy**

**from scipy import linalg**

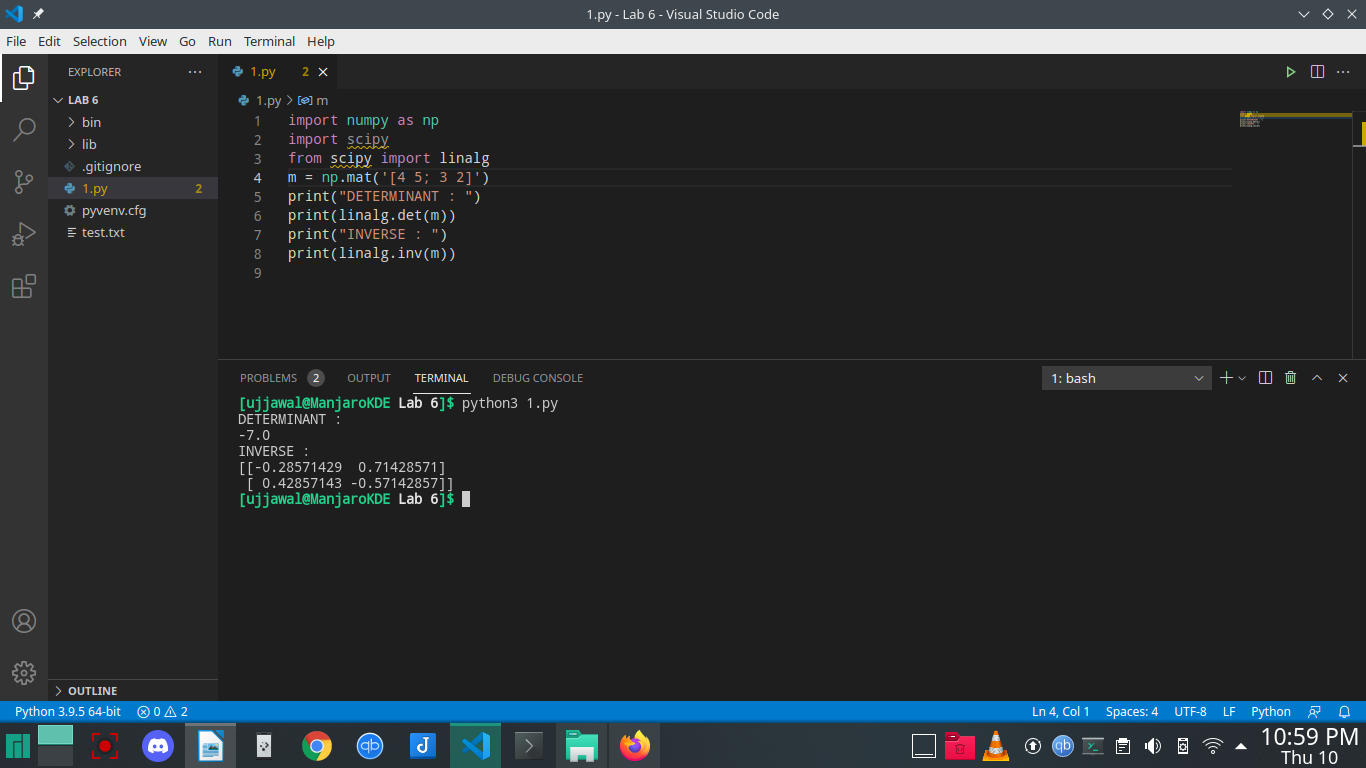
**m = np.mat('[4 5; 3 2]')**

**print("DETERMINANT : ")**

**print(linalg.det(m))**

**print("INVERSE : ")**

**print(linalg.inv(m))**



**Question 4:**Define two-dimensional array with values {(5,4),(6,3)}.Output eigen values andeigenvectors of the matrix

**import numpy as np**

**import scipy**

**from scipy import linalg**

**m = np.array([[5, 4], [6, 3]])**

**print("EIGEN VALUES : ")**

**l, v = linalg.eig(m)**

**l1, l2 = l**

**print(l1, l2)**

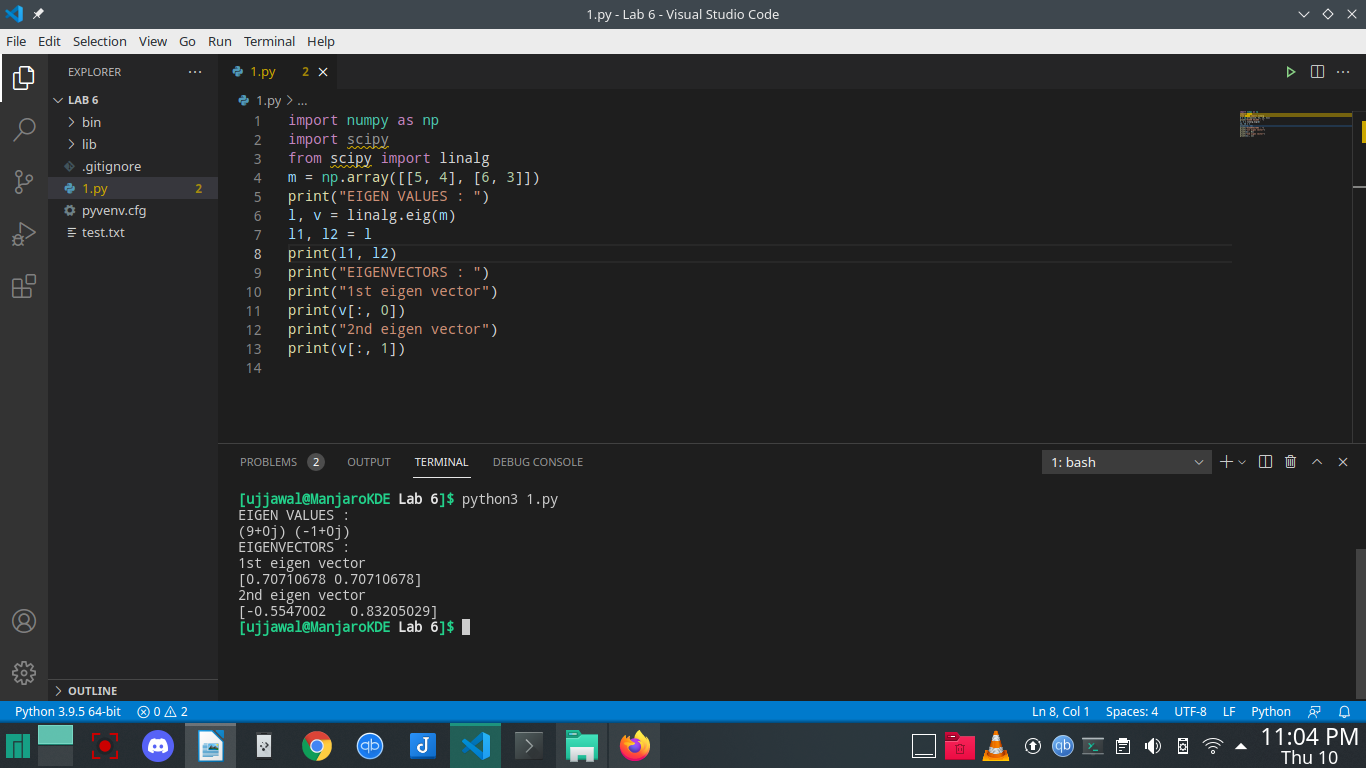
**print("EIGENVECTORS : ")**

**print("1st eigen vector")**

**print(v[:, 0])**

**print("2nd eigen vector")**

**print(v[:, 1])**



**Question 5:**Create Sparse matrices A and B and analyze various functions of sciPy sparse package

**import numpy as np**

**from numpy.core.numeric import count\_nonzero**

**import scipy**

**from scipy import linalg**

**from scipy.sparse import csr\_matrix**

**from scipy.sparse.csc import csc\_matrix**

**A = np.array([[1, 0, 0, 1, 0, 0], [0, 0, 2, 0, 0, 1], [0, 0, 0, 2, 0, 0]])**

**B = np.array([[0, 0, 2, 1, 0, 0], [0, 4, 0, 4, 0, 1], [0, 0, 0, 2, 0, 9]])**

**print("convert into sparse matrix using CSR")**

**S = csr\_matrix(A)**

**print(S)**

**print("DENSE MATRIX")**

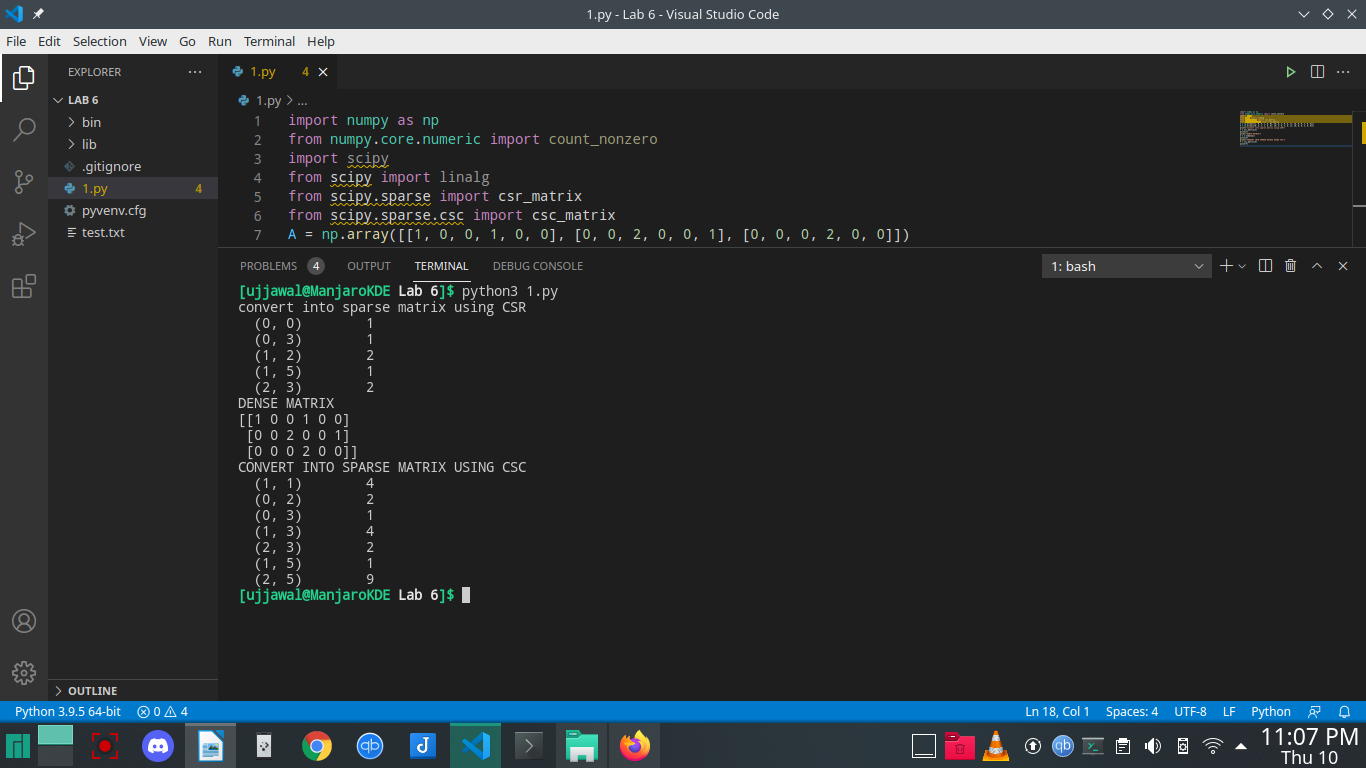
**Q = S.todense()**

**print(Q)**

**print("CONVERT INTO SPARSE MATRIX USING CSC")**

**P = csc\_matrix(B)**

**print(P)**

****