**Sahakar Maharshi Bhausaheb Santuji Thorat College Sangamner**

**Remark**

**Demonstrator’s Signature**

**Date:-**

**/**

**/20**

**DEPARTMENT OF COMPUTER SCIENCE**

**Sub : Mathematics**

**Name:-\_Gorde Yash Somnath Roll.No:- 21\_Date:- Title of the expt:- Slip no 9 Page.no:- Class:- BCS**

**Q1 Attempt any TWO of the following**

**A ) Write a python program to Plot 2D X-axis and Y-axis black color and in the same diagram plot green triangle with vertices [5,4],[7,4],[6,6]**

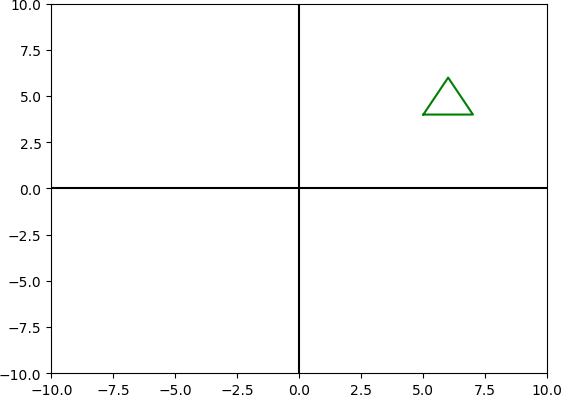
**-**

import matplotlib.pyplot as plt plt.xlim(-10, 10)

plt.ylim(-10, 10) plt.axhline(0, color='black') plt.axvline(0, color='black') x = [5, 7, 6, 5]

y = [4, 4, 6, 4]

plt.plot(x, y, color='green') plt.show()



**B ) Write a program in python to rotate the point through YZ-plane in anticlockwise direction(rotation through Y-axis by an angle of 90⸰)**

**-**

import numpy as np

import matplotlib.pyplot as plt p = np.array([[1], [2], [3]]) theta = np.radians(90)

rot\_matrix = np.array([[np.cos(theta), 0, np.sin(theta)], [0, 1, 0],

[-np.sin(theta), 0, np.cos(theta)]]) p\_rotated = np.dot(rot\_matrix, p)

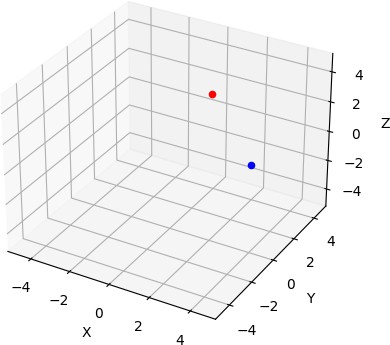
fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d') ax.scatter(p[0], p[1], p[2], c='red', marker='o')

ax.scatter(p\_rotated[0], p\_rotated[1], p\_rotated[2], c='blue', marker='o') ax.set\_xlim(-5, 5)

ax.set\_ylim(-5, 5)

ax.set\_zlim(-5, 5) ax.set\_xlabel('X') ax.set\_ylabel('Y') ax.set\_zlabel('Z') plt.show()



**C ) Using Python plot the graph of function f(x)=cos(x) on the interval [0,2π]**

**-**

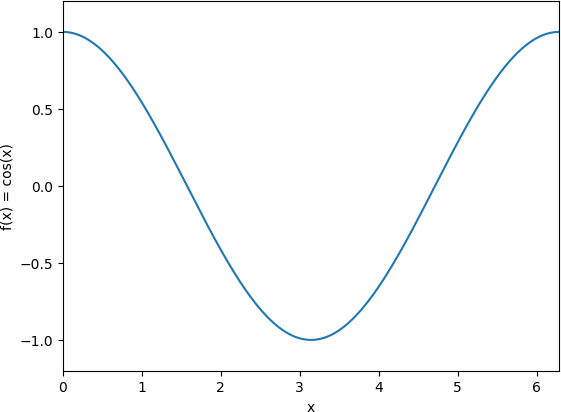
import numpy as np

import matplotlib.pyplot as plt

x = np.linspace(0, 2\*np.pi, 1000) y = np.cos(x)

plt.plot(x, y) plt.xlim(0, 2\*np.pi) plt.ylim(-1.2, 1.2)

plt.xlabel('x') plt.ylabel('f(x) = cos(x)') plt.show()



**Q2 ) Attempt any TWO of the following**

**A ) Write a python program to rotate the ray by 90⸰ having starting point (1,0) and (2,-1)**

**-**

import math

import matplotlib.pyplot as plt

start = [1, 0]

end = [2, -1]

theta = math.atan2(end[1]-start[1], end[0]-start[0])

new\_end = [start[0] + math.cos(theta + math.radians(90)), start[1] + math.sin(theta + math.radians(90))]

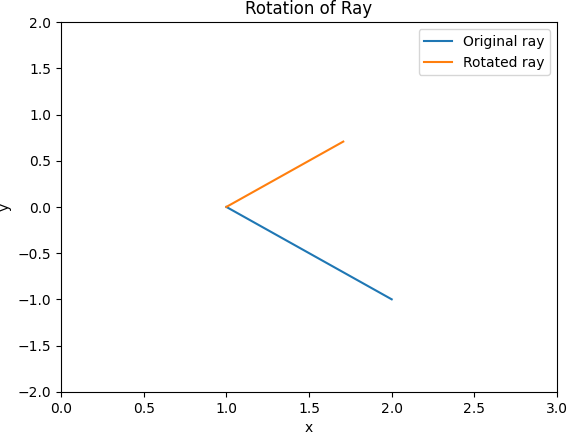
plt.plot([start[0], end[0]], [start[1], end[1]], label='Original ray') plt.plot([start[0], new\_end[0]], [start[1], new\_end[1]], label='Rotated ray') plt.xlim(0, 3)

plt.ylim(-2, 2)

plt.xlabel('x')

plt.ylabel('y') plt.title('Rotation of Ray') plt.legend()

plt.show()



**B ) Using sympy declare the points A(0,7), B(5,2). Declare the line segment passing through them Find length and midpoint of the line ssegment passing through points A and B**

**-**

from sympy import Point from sympy import Segment

A = Point(0, 7)

B = Point(5, 2)

AB = Segment(A, B) length = AB.length

print("Length of AB:", length) midpoint = AB.midpoint print("Midpoint of AB:", midpoint)

output :

Length of AB: 5\*sqrt(2)

Midpoint of AB: Point2D(5/2, 9/2)

**c) Write a python program to find the area and perimeter of ∆ABC where A(0,0),B(5,0),C(3,3)**

**-**

from math import sqrt

A = (0, 0)

B = (5, 0)

C = (3, 3)

AB = sqrt((B[0]-A[0])\*\*2 + (B[1]-A[1])\*\*2)

BC = sqrt((C[0]-B[0])\*\*2 + (C[1]-B[1])\*\*2)

CA = sqrt((A[0]-C[0])\*\*2 + (A[1]-C[1])\*\*2)

perimeter = AB + BC + CA s = perimeter / 2

area = sqrt(s\*(s-AB)\*(s-BC)\*(s-CA)) print("Perimeter of triangle ABC:", perimeter) print("Area of triangle ABC:", area)

output :

Perimeter of triangle ABC: 12.848191962583275 Area of triangle ABC: 7.5000000000000036

**Q 3) Attempt the following**

**A ) Attempt any ONE of the Following**

**i) Write A python program to solve the following LPP : MAX Z=150x+75y**

**Subject to 4x+6y≤24**

**5x+3y≤15 X,y≥0**

**-**

from scipy.optimize import linprog c = [-150, -75]

A = [[4, 6], [5, 3]]

b = [24, 15]

x\_bounds = (0, None) y\_bounds = (0, None)

result = linprog(c, A\_ub=A, b\_ub=b, bounds=[x\_bounds, y\_bounds])

print("Solution status:", result.message) print("Optimal value of x:", result.x[0]) print("Optimal value of y:", result.x[1]) print("Optimal value of Z:", -result.fun)

output :

Solution status: Optimization terminated successfully. (HiGHS Status 7: Optimal) Optimal value of x: 3.0

Optimal value of y: 0.0 Optimal value of Z: 450.0

**II) Write a python program to display the following LPP by using pulp module and simplex method Find its optimal solution if exist**

**Max Z=4x+y+3z+5w Subject to 4x+6y-4z+w≥20**

**-3x-2y+4z+w≤10**

**-8x-3y+3z+2w≤20 X,y,z,w≥0**

**-**

from scipy.optimize import linprog c = [-4, -1, -3, -5]

A = [[4, 6, -4, 1], [-3, -2, 4, 1], [-8, -3, 3, 2]]

b = [20, 10, 20]

x\_bounds = (0, None) y\_bounds = (0, None) z\_bounds = (0, None) w\_bounds = (0, None)

result = linprog(c, A\_ub=A, b\_ub=b, bounds=[x\_bounds, y\_bounds, z\_bounds, w\_bounds])

print("Solution status:", result.message) print("Optimal value of x:", result.x[0]) print("Optimal value of y:", result.x[1]) print("Optimal value of z:", result.x[2]) print("Optimal value of w:", result.x[3]) print("Optimal value of Z:", -result.fun)

output :

Solution status: Optimization terminated successfully. (HiGHS Status 7: Optimal) Optimal value of x: 30.0

Optimal value of y: 0.0 Optimal value of z: 25.0 Optimal value of w: 0.0 Optimal value of Z: 195.0

**B ) Attempt any ONE of the following**

**I ) Write a python program to apply the following transformation on the point (-2,4) :**

**A ) Shering in Y direction by 7 units**

**B ) Scaling in X and Y direction by** 𝟕 **and & units respectively**

𝟐

**C ) Shering in X and Y direction by 4 and 7 units respectively**

**D) Rotation about origin by an angle 60⸰**



import numpy as np point = np.array([-2, 4])

A = np.array([[1, 0], [7, 1]])

B = np.array([[7/2, 0], [0, 5]])

C = np.array([[1, 4], [7, 1]])

theta = np.radians(60)

D = np.array([[np.cos(theta), -np.sin(theta)], [np.sin(theta), np.cos(theta)]])

transformed\_point\_a = np.dot(A, point) transformed\_point\_b = np.dot(B, point) transformed\_point\_c = np.dot(C, point) transformed\_point\_d = np.dot(D, point)

print("Original point: ", point)

print("Transformed point after A: ", transformed\_point\_a) print("Transformed point after B: ", transformed\_point\_b) print("Transformed point after C: ", transformed\_point\_c) print("Transformed point after D: ", transformed\_point\_d)

output :

Original point: [-2 4]

Transformed point after A: [ -2 -10] Transformed point after B: [-7. 20.] Transformed point after C: [ 14 -10]

Transformed point after D: [-4.46410162 0.26794919]

**II ) Write a python program to find combined transformation of the line segment between the points A[5,3] & B[1,4] for the following sequence of transformation**

1. **Rotation about origin through an angle** 𝝅

𝟐

1. **Uniform scaling by -3.5 units**
2. **Scaling in Y-axis by 5 units**
3. **Shering in X and Y direction by 3 and 4 units respectively**



import numpy as np

line\_segment = np.array([[5, 3], [1, 4]])

theta\_a = np.pi/2

A = np.array([[np.cos(theta\_a), -np.sin(theta\_a)], [np.sin(theta\_a), np.cos(theta\_a)]])

B = np.array([[-3.5, 0], [0, -3.5]])

C = np.array([[1, 0], [0, 5]])

D = np.array([[1, 3], [4, 1]])

combined\_transform = np.dot(D, np.dot(C, np.dot(B, A))) transformed\_line\_segment = np.dot(combined\_transform, line\_segment.T).T

print("Original line segment: ") print(line\_segment)

print("Transformed line segment after A, B, C, and D: ") print(transformed\_line\_segment)

output :

Original line segment:

[[5 3]

[1 4]]

Transformed line segment after A, B, C, and D: [[-252. -45.5]

[ -38.5 38.5]]