**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 7 Page.no:- Class:- BCS**

# Q1 ) Attempt any TWO of the following

**A ) Plot the graph of f(x)=x4 in [0,5] with red dashed line with circle marked**

**-**

import numpy as np

import matplotlib.pyplot as plt def f(x):

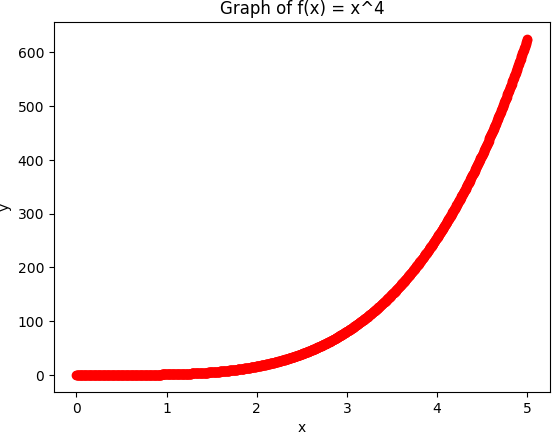
return x\*\*4

x\_values = np.linspace(0, 5, num=1000) y\_values = f(x\_values)

plt.plot(x\_values, y\_values, 'r--', marker='o') plt.title('Graph of f(x) = x^4')

plt.xlabel('x')

plt.ylabel('y') plt.show()



# B ) Using Python program generate 3D surface plot for the function f(x)=sin(x2+y2) in the interval [0,10]

**-**

import numpy as np

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d import Axes3D def f(x, y):

return np.sin(x\*\*2 + y\*\*2)

x\_values = np.linspace(0, 10, 100)

y\_values = np.linspace(0, 10, 100)

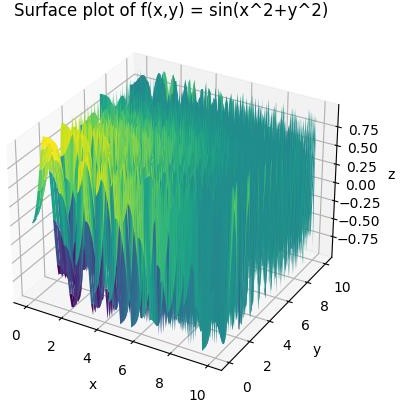
x\_mesh, y\_mesh = np.meshgrid(x\_values, y\_values) z\_values = f(x\_mesh, y\_mesh)

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d') ax.plot\_surface(x\_mesh, y\_mesh, z\_values, cmap='viridis') ax.set\_xlabel('x')

ax.set\_ylabel('y') ax.set\_zlabel('z')

ax.set\_title('Surface plot of f(x,y) = sin(x^2+y^2)') plt.show()



# C ) Write a python program to draw rectangle with vertices [1,0],[2,1],[1,2] and [0,1] its rotation about the origin by 𝝅 radians

𝟐

**-**

import numpy as np

import matplotlib.pyplot as plt

vertices = np.array([[1, 0], [2, 1], [1, 2], [0, 1], [1, 0]]) theta = np.pi/2

rotation\_matrix = np.array([[np.cos(theta), -np.sin(theta)],

[np.sin(theta), np.cos(theta)]]) rotated\_vertices = vertices @ rotation\_matrix

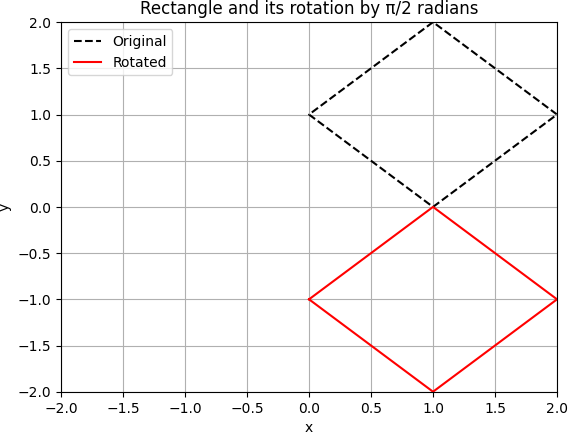
fig, ax = plt.subplots()

ax.plot(vertices[:,0], vertices[:,1], 'k--', label='Original') ax.plot(rotated\_vertices[:,0], rotated\_vertices[:,1], 'r-', label='Rotated') ax.legend()

ax.set\_xlim(-2, 2)

ax.set\_ylim(-2, 2) ax.grid() ax.set\_xlabel('x') ax.set\_ylabel('y')

ax.set\_title('Rectangle and its rotation by π/2 radians') plt.show()



# Q 2) Attempt any TWO of the following

1. **Write a python program to reflect the line segment joining the points A[5,3] & B[1,4] through the line y=x+1**

**-**

import numpy as np

import matplotlib.pyplot as plt

A = np.array([5, 3])

B = np.array([1, 4]) def reflection\_line(x):

return x - 1

m = 1

b = 1

A\_image = np.array([(1-m\*\*2)\*A[0] + 2\*m\*A[1] - 2\*m\*b, (1-m\*\*2)\*A[1] + 2\*m\*A[0] - 2\*b])

B\_image = np.array([(1-m\*\*2)\*B[0] + 2\*m\*B[1] - 2\*m\*b, (1-m\*\*2)\*B[1] + 2\*m\*B[0] - 2\*b])

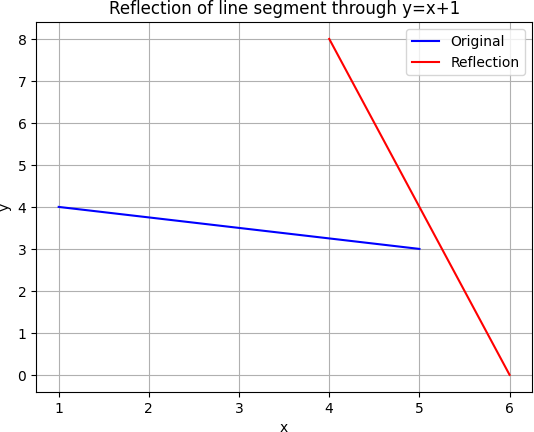
fig, ax = plt.subplots()

ax.plot([A[0], B[0]], [A[1], B[1]], 'b-', label='Original')

ax.plot([A\_image[0], B\_image[0]], [A\_image[1], B\_image[1]], 'r-', label='Reflection') ax.legend()

ax.grid() ax.set\_xlabel('x') ax.set\_ylabel('y')

ax.set\_title('Reflection of line segment through y=x+1') plt.show()



1. **Using python declare the points P(5,2),Q(5,-2),R(5,0) check whether these points are collinear.Declare the ray passing through the points P and Q find the length of this ray between P and Q. Also find slope of this ray**

**-**

import math

# Define the coordinates of the points P, Q, and R

P = (5, 2)

Q = (5, -2)

R = (5, 0)

# Check if the points P, Q, and R are collinear

if (Q[1]-P[1])\*(R[0]-Q[0]) == (R[1]-Q[1])\*(Q[0]-P[0]):

print("The points P, Q, and R are collinear.") else:

print("The points P, Q, and R are not collinear.")

# Define the ray passing through P and Q ray\_direction = (Q[0]-P[0], Q[1]-P[1])

# Find the length of the ray between P and Q

ray\_length = math.sqrt(ray\_direction[0]\*\*2 + ray\_direction[1]\*\*2)

# Find the slope of the ray if ray\_direction[0] != 0:

slope = ray\_direction[1] / ray\_direction[0] else:

slope = float('inf')

# Print the length and slope of the ray

print("The length of the ray between P and Q is", ray\_length) print("The slope of the ray is", slope)

Output :

The points P, Q, and R are collinear.

The length of the ray between P and Q is 4.0 The slope of the ray is inf

**C ) write a python program in 3D to rotate the point (1,0,0) through X plane in anticlock wise direction(rotation through Z axis) by angle of 90⸰**

**-**

import numpy as np point = np.array([1, 0, 0]) theta = np.radians(90)

rotation\_matrix = np.array([[np.cos(theta), -np.sin(theta), 0], [np.sin(theta), np.cos(theta), 0],

[0, 0, 1]])

new\_point = np.dot(rotation\_matrix, point)

print("The original point was:", point) print("The rotated point is:", new\_point)

output:

The original point was: [1 0 0]

The rotated point is: [6.123234e-17 1.000000e+00 0.000000e+00]

**Q 3 ) Attempt the following**

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

**I ) Write a python program to solve the following LPP : Min Z=3.5x+2y**

**Subject to x+y≥5**

**x≥4 y≤2 x,y≥0**

-

from pulp import \*

prob = LpProblem("LP Problem", LpMinimize) x = LpVariable("x", lowBound=0)

y = LpVariable("y", lowBound=0) prob += 3.5\*x + 2\*y

prob += x + y >= 5 prob += x >= 4 prob += y <= 2 prob.solve()

print("Status: ", LpStatus[prob.status]) print("Optimal values:")

for v in prob.variables(): print(v.name, "=", v.varValue)

print("Optimal objective value: ", value(prob.objective))

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

**Max Z=x+2y+z Subject to x+2y+2z≤1**

**3x+2y+z≥8 X,y,x≥0**

-

from pulp import \*

prob = LpProblem("LP Problem", LpMaximize) x = LpVariable("x", lowBound=0)

y = LpVariable("y", lowBound=0)

z = LpVariable("z", lowBound=0)

prob += x + 2\*y + z

prob += x + 2\*y + 2\*z <= 1 prob += 3\*x + 2\*y + z >= 8 prob.solve()

print("Status: ", LpStatus[prob.status]) print("Optimal values:")

for v in prob.variables(): print(v.name, "=", v.varValue)

print("Optimal objective value: ", value(prob.objective))

B ) Attempt any ONE of the following

**I ) Apply Python program in each of the following transformation on the point P[4,-2]**

**A ) Reflection through the Y-axis**

* 1. **Scaling in X-coordinate by factor 3**
  2. **Rotation about origin through an angle π**
  3. **Shearing in both X and Y direction by -2 and 4 units respectively**

**-**

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

P\_reflect\_y = np.array([-4, 2])

print("Reflection through Y-axis:", P\_reflect\_y)

P\_scale\_x = np.array([12, -2])

print("Scaling in X-coordinate by factor 3:", P\_scale\_x)

theta = np.pi

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

P\_rotate = rot\_matrix.dot(P)

print("Rotation about origin through an angle π:", P\_rotate)

shear\_matrix = np.array([[1, -2], [4, 1]]) P\_shear = shear\_matrix.dot(P)

print("Shearing in both X and Y direction by -2 and 4 units respectively:", P\_shear)

Output :

Reflection through Y-axis: [-4 2]

Scaling in X-coordinate by factor 3: [12 -2] Rotation about origin through an angle π: [-4. 2.]

Shearing in both X and Y direction by -2 and 4 units respectivel y: [ 8 14]

**II ) Find the combined transformation of the line segment between the points A[4,-1] & B[3,2] by using python program for the following sequence of transformation**

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

𝟒

1. **Shering in Y direction by 4 units**
2. **Scaling in X-coordinate by 5 units**
3. **reflection through Y-axis**

**-**

import numpy as np

A = np.array([4, -1])

B = np.array([3, 2]) theta = np.pi/4

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

A\_rot = rot\_matrix.dot(A) B\_rot = rot\_matrix.dot(B)

shear\_matrix = np.array([[1, 0], [4, 1]]) A\_shear = shear\_matrix.dot(A\_rot) B\_shear = shear\_matrix.dot(B\_rot)

scale\_matrix = np.array([[5, 0], [0, 1]]) A\_scale = scale\_matrix.dot(A\_shear) B\_scale = scale\_matrix.dot(B\_shear)

A\_reflect\_y = np.array([-A\_scale[0], A\_scale[1]]) B\_reflect\_y = np.array([-B\_scale[0], B\_scale[1]])

print("Transformed point A:", A\_reflect\_y) print("Transformed point B:", B\_reflect\_y)

output :

Transformed point A: [-17.67766953 16.26345597]

Transformed point B: [-3.53553391 6.36396103]