Sahakar Maharshi Bhausaheb Santuji Thorat

College Sangamner

DEPARTMENT OF COMPUTER SCIENCE

Sub : Mathematics

Remark
Demonstrator's
Signature
Date:- / /20

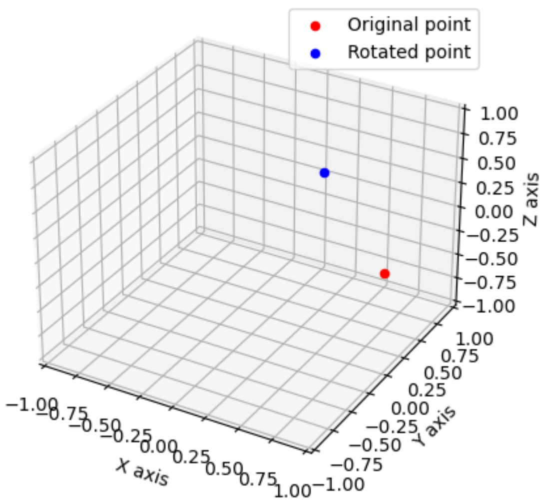
Name:- Gorde Yash Somnath Roll.No:- 21 Date:-
Title of the expt:- Slip no 10 Page.no:- Class:- BCS

Q1) Attempt any TWO of the following

A) Write a python program in 3D to rotate the point(1,0,0) through XY plane in clockwise direction(Rotation through Z-axis by an angle of 90◦)

-→

```
import math
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

p = [1, 0, 0]
theta = math.radians(90)
cos_theta = math.cos(theta)
sin_theta = math.sin(theta
                   )
rotation_matrix = np.array([
    [cos_theta, -sin_theta, 0],
    [sin_theta, cos_theta, 0],
    [0, 0, 1]
])
rotated_p = rotation_matrix.dot(p)
fig = plt.figure()

ax = fig.add_subplot(111, projection='3d')
ax.scatter(p[0], p[1], p[2], c='r', label='Original point')
ax.scatter(rotated_p[0], rotated_p[1], rotated_p[2], c='b', label='Rotated point')
ax.set_xlim([-1, 1])
ax.set_ylim([-1, 1])
ax.set_zlim([-1, 1])
ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')
ax.legend()
plt.show()
```

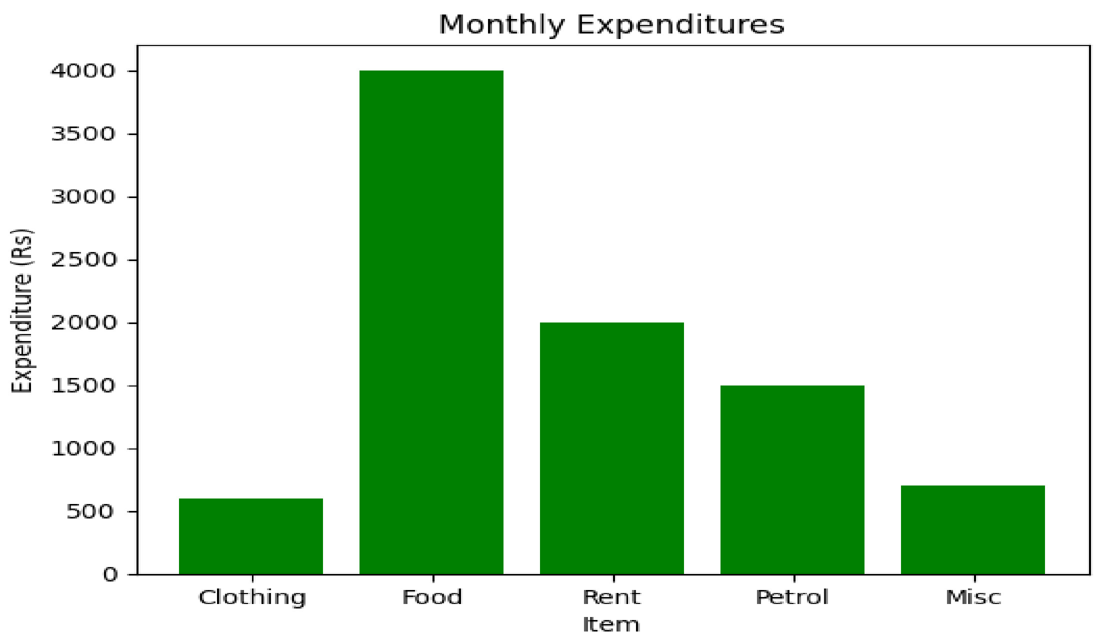


B) Represent the following information using bar graph(in green color)

Item	**Clothing**	**Food**	**rent**	**Petrol**	**Misc**
Expenditure in RS	**600**	**4000**	**2000**	**1500**	**700**

-→

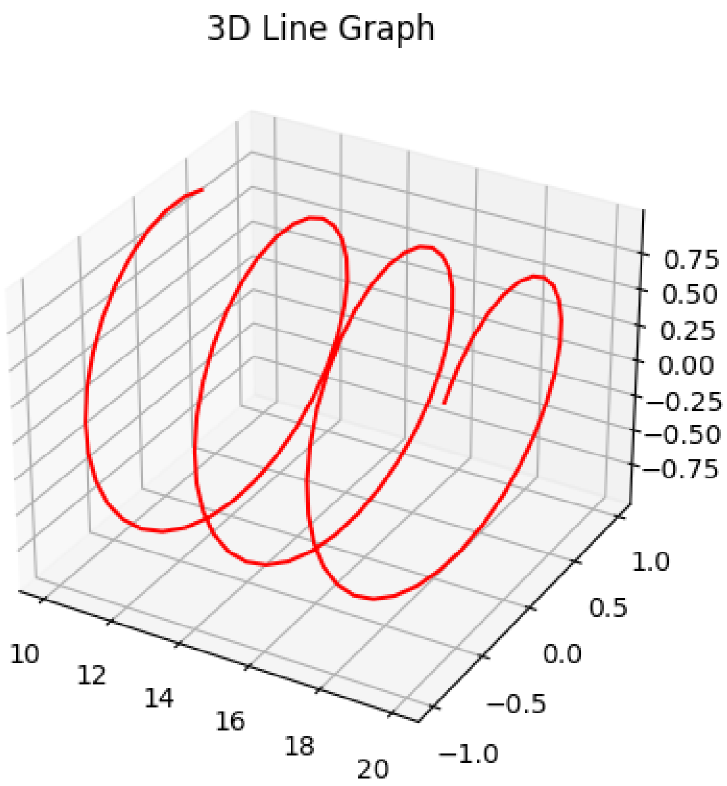
```
import matplotlib.pyplot as plt
items = ['Clothing', 'Food', 'Rent', 'Petrol', 'Misc']
expenditures = [600, 4000, 2000, 1500, 700]
plt.bar(items, expenditures, color='green')
plt.title('Monthly Expenditures')
plt.xlabel('Item')
plt.ylabel('Expenditure (Rs)')
plt.show()
```



C) Write a python program to plot a 3D line graph whose parametric equation is (cos(2x),sin(2x),x) for 10≤x≤20 (in red color) , with titile to the graph

-🡪

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
def parametric_eq(x):
   return (np.cos(2*x), np.sin(2*x), x)

x_vals = np.linspace(10, 20, 100)
y_vals, z_vals, _ = parametric_eq(x_vals)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot(x_vals, y_vals, z_vals, color='red')
ax.set_title('3D Line Graph')
plt.show()
```

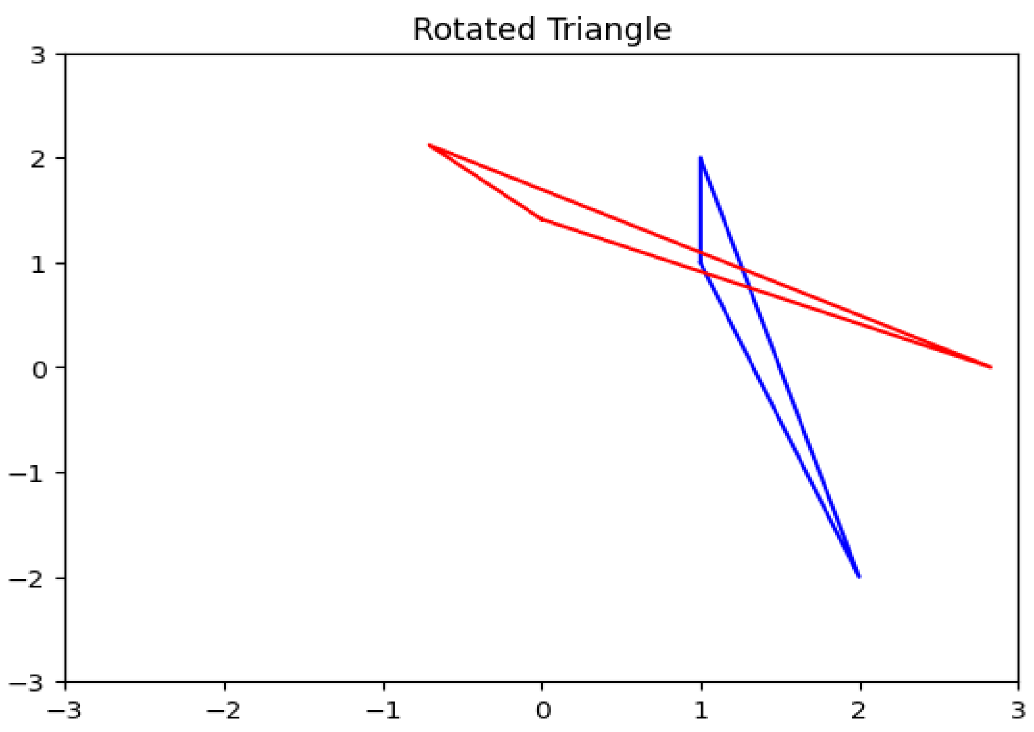


Q2) Attempt any TWO of the following

A) Write a python program to rotate the ΔABC,where A(1,1),B(2,-2),C(1,2)

-→

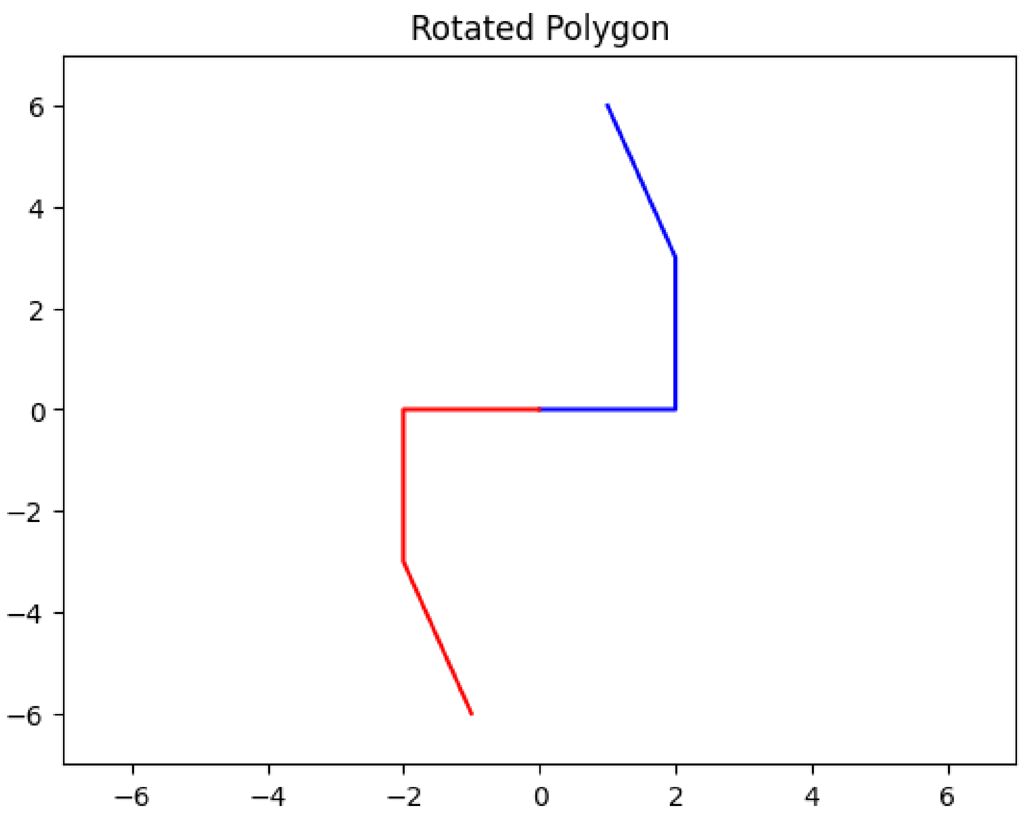
```
import numpy as np
import matplotlib.pyplot as plt
A = [1, 1]
B = [2, -2]
C = [1, 2]

theta = np.pi/4
R = np.array([[np.cos(theta), -np.sin(theta)], [np.sin(theta), np.cos(theta)]])
A_rot = R.dot(A)
B_rot = R.dot(B)
C_rot = R.dot(C)
fig, ax = plt.subplots()

ax.plot([A[0], B[0], C[0], A[0]], [A[1], B[1], C[1], A[1]], 'b')
ax.plot([A_rot[0], B_rot[0], C_rot[0], A_rot[0]], [A_rot[1], B_rot[1], C_rot[1],
A_rot[1]], 'r')
ax.set_xlim([-3, 3])
ax.set_ylim([-3, 3])
ax.set_title('Rotated Triangle')
plt.show()
```



B) Draw a polygon with vertices (0,0),(2,0),(2,3),(1,6) , write a python program t rotate the polygon by 180◦

-→

```
import numpy as np
import matplotlib.pyplot as plt
vertices = np.array([(0, 0), (2, 0), (2, 3), (1, 6)])
theta = np.pi
R = np.array([[np.cos(theta), -np.sin(theta)], [np.sin(theta), np.cos(theta)]])
vertices_rot = np.dot(vertices, R)
fig, ax = plt.subplots()
ax.plot(vertices[:, 0], vertices[:, 1], 'b')
ax.plot(vertices_rot[:, 0], vertices_rot[:, 1], 'r')
ax.set_xlim([-7, 7])
ax.set_ylim([-7, 7])
ax.set_title('Rotated Polygon')
plt.show()
```



C) Find the area and perimeter of the ∆ABC where A[0,0],B[5,0],C[3,3]

-→

```
import numpy as np
A = np.array([0, 0])
```

```
B = np.array([5, 0])
C = np.array([3, 3])

AB = np.linalg.norm(B - A)
BC = np.linalg.norm(C - B)
AC = np.linalg.norm(C - A)

perimeter = AB + BC + AC
s = perimeter / 2
area = np.sqrt(s * (s - AB) * (s - BC) * (s - AC))

print("Area:", area)
print("Perimeter:", perimeter)
```

output :

```
Area: 7.5000000000000036
Perimeter: 12.848191962583275
```

Q3) Attempt the following

A) Attempt any ONE of the following

I) Solve the following LPP :

Max $Z=x+y$

Subject to $x-y\geq1$

$X+y\geq2$

$X,y\geq0$

-→

```
from scipy.optimize import linprog
c = [1, 1]
A = [[-1, 1], [1, 1]]
b = [-1, 2]
bounds = [(0, None), (0, None)]
result = linprog(c, A_ub=A, b_ub=b, bounds=bounds, method='simplex')

if result.success:
    optimal_value = result.fun
    optimal_point = result.x
    print("Optimal value:", optimal_value)
    print("Optimal point:", optimal_point)
else:
```

```
    print("No solution found.")
```

output ;

```
Optimal value: 1.0
Optimal point: [1. 0.]
```

II) Write a python program to display the following LPP by using pulp module and sim plex
Method .Find its optimal solution if exist

Max z=3x+2y+5z
Subject to x+2y+z≤430
3x+4z≤460
X+4y≤120
X,y,z≥0

-→

```
from pulp import *
prob = LpProblem("LP Problem", LpMaximize)
x = LpVariable('x', lowBound=0, cat='Continuous')
y = LpVariable('y', lowBound=0, cat='Continuous')
z = LpVariable('z', lowBound=0, cat='Continuous')

prob += 3*x + 2*y + 5*z
prob += x + 2*y + z <= 430
prob += 3*x + 4*z <= 460
prob += x + 4*y <= 120
prob.solve(solvers.PULP_CBC_CMD(msg=False))

print("Status:", LpStatus[prob.status])
if LpStatus[prob.status] == 'Optimal':
    print("Optimal value:", value(prob.objective))
    print("Optimal point: x = %s, y = %s, z = %s" % (value(x), value(y), value(z)))
```

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 $\frac{3}{2}$ and 4 unit respectively
C) Shering in X and Y direction by 2 and 4 units respectively
D) Rotation about origin an angle 45◦

-→

```
import math
```

```
point = (-2, 4)
print("Original point:", point)

shear_y = lambda x, y: (x, y+7)
point = shear_y(*point)
print("After shearing in Y direction:", point)

scale = lambda x, y: (3/2*x, 4*y)
point = scale(*point)
print("After scaling:", point)

shear_x = lambda x, y: (x+2*y, y+4*x)
point = shear_x(*point)
print("After shearing in X and Y direction:", point)
rotate = lambda x, y: (x*math.cos(math.pi/4) - y*math.sin(math.pi/4),
                       x*math.sin(math.pi/4) + y*math.cos(math.pi/4))
point = rotate(*point)
print("After rotation:", point)
```

Output :

```
  Original point: (-2, 4)
After shearing in Y direction: (-2, 11)
After scaling: (-3.0, 44)
After shearing in X and Y direction: (85.0, 32.0)
After rotation: (37.47665940288702, 82.73149339882606)
```

II)Write a python program to find the combined transformation of the line segment between the points A[3,2] & B[2,-3] for the following sequence of transformation

A) Rotation about origin through an angle $\frac{\pi}{6}$

B) Scaling in Y-coordinate by -4 units

C) Uniform scaling by -6.4 units

D) Shering in Y-direction by 5 units

-→

```
import math
import numpy as np
A = np.array([3, 2])
B = np.array([2, -3])
print("Original points:\nA =", A, "\nB =", B)
```

```
rotate = lambda point, angle: np.dot(np.array([[math.cos(angle), -math.sin(angle)],
                                               [math.sin(angle), math.cos(angle)]]), point)
A = rotate(A, math.pi/6)
B = rotate(B, math.pi/6)
print("After rotation:\nA =", A, "\nB =", B)

scale_y = lambda point, factor: np.array([point[0], point[1]*factor])
A = scale_y(A, -4)
B = scale_y(B, -4)
print("After scaling in Y-coordinate:\nA =", A, "\nB =", B)

uniform_scale = lambda point, factor: np.array([point[0]*factor, point[1]*factor])
A = uniform_scale(A, -6.4)
B = uniform_scale(B, -6.4)
print("After uniform scaling:\nA =", A, "\nB =", B)

shear_y = lambda point, factor: np.dot(np.array([[1, 0],
                                                 [factor, 1]]), point)
A = shear_y(A, 5)
B = shear_y(B, 5)
print("After shearing in Y-direction:\nA =", A, "\nB =", B)
```

Output :

```
  Original points:
A = [3 2]
B = [ 2 -3]
After rotation:
A = [1.59807621 3.23205081]
B = [ 3.23205081 -1.59807621]
After scaling in Y-coordinate:
A = [  1.59807621 -12.92820323]
B = [3.23205081 6.39230485]
After uniform scaling:
A = [-10.22768775  82.74050067]
B = [-20.68512517 -40.91075101]
After shearing in Y-direction:
A = [-10.22768775  31.60206191]
B = [ -20.68512517 -144.33637685]
```