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 18_____Page.no:-_____Class:-_____BCS_____

Q1) Attempt any TWO of the following

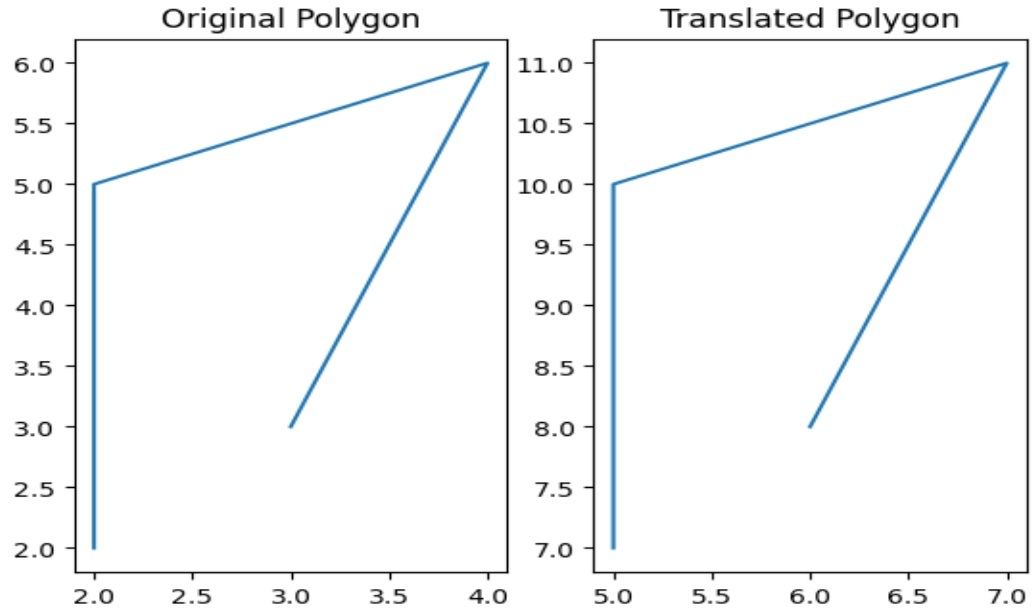
A) Write a python program to draw polygon with vertices [3,3],[4,6],[2,5],[2,2] and its trasnslation in X and Y direction using the factors 3,5 respectively -→

```
import matplotlib.pyplot as plt
import numpy as np
vertices = np.array([[3, 3], [4, 6], [2, 5], [2, 2]])
tx = 3
ty = 5
fig, axs = plt.subplots(1, 2)

axs[0].plot(vertices[:, 0], vertices[:, 1])
axs[0].set_title('Original Polygon')

translation_matrix = np.array([[1, 0, tx], [0, 1, ty], [0, 0, 1]])
translated_vertices = np.dot(translation_matrix, np.hstack((vertices, np.ones((len(vertices),
1))))).T).T[:, :-1]

axs[1].plot(translated_vertices[:, 0], translated_vertices[:, 1])
axs[1].set_title('Translated Polygon')
plt.show()
```



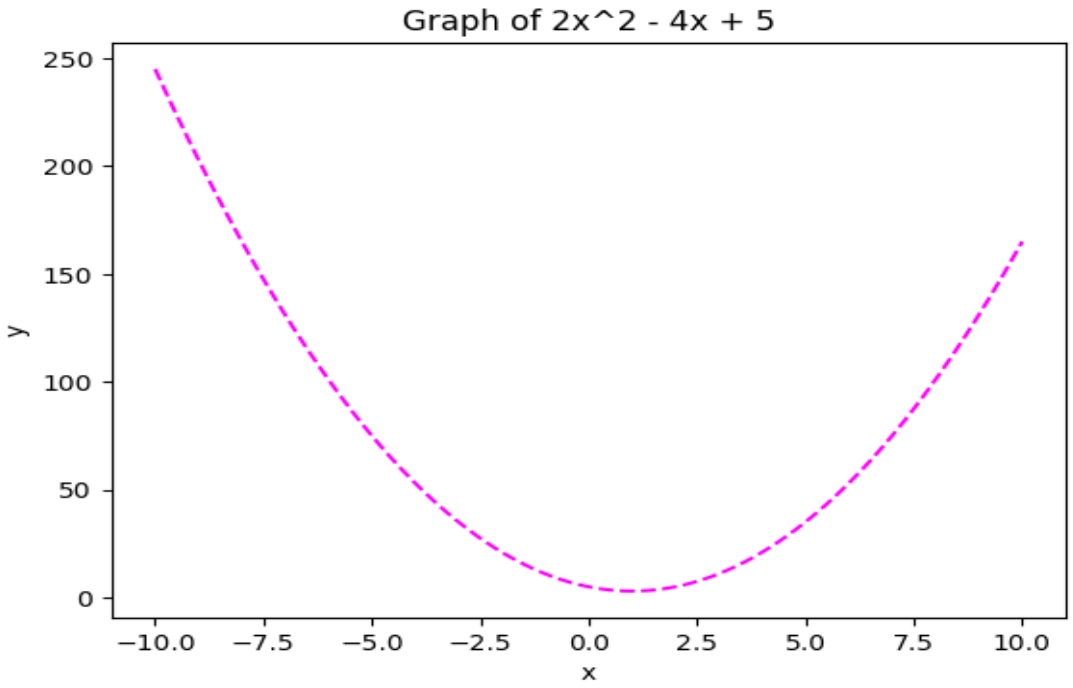
B) Write a python program to plot the graph $2x^2-4x+5$ in [-10,10] in magneta colored dashed pattern

-→

```
import matplotlib.pyplot as plt
import numpy as np
def f(x):
    return 2*x**2 - 4*x + 5

x = np.linspace(-10, 10, 1000)
y = f(x)

plt.plot(x, y, color='magenta', linestyle='--')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Graph of 2x^2 - 4x + 5')
plt.show()
```



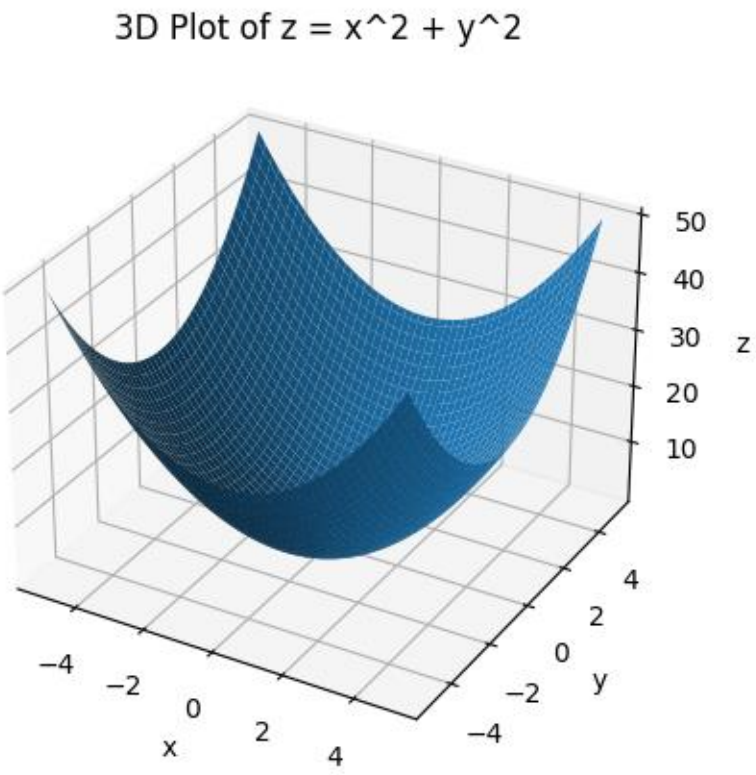
C) Write a python program to generate 3D plot of the function $z=x^2+y^2$ in $-5<x,y<5$

-→

```
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
import numpy as np
def f(x, y):
    return x**2 + y**2

x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
X, Y = np.meshgrid(x, y)
Z = f(X, Y)
```

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z)
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')
ax.set_title('3D Plot of z = x^2 + y^2')
plt.show()
```



Q2) Attempt any TWO of the following

A) Write a python program to generate vector x in the interval [-22,22] using numpy package with 80 subintervals

-→

```
import numpy as np
n = 80
x = np.linspace(-22, 22, n+1)
print(x)
```

outuput :

```
[-22.   -21.45 -20.9  -20.35 -19.8  -19.25 -18.7  -18.15 -17.6  -17.05
 -16.5  -15.95 -15.4  -14.85 -14.3  -13.75 -13.2  -12.65 -12.1  -11.55
 -11.   -10.45  -9.9   -9.35  -8.8   -8.25  -7.7   -7.15  -6.6   -6.05
  -5.5   -4.95  -4.4   -3.85  -3.3   -2.75  -2.2   -1.65  -1.1   -0.55
```

```
 0.    0.55  1.1   1.65  2.2   2.75  3.3   3.85  4.4   4.95
 5.5   6.05  6.6   7.15  7.7   8.25  8.8   9.35  9.9   10.45
11.   11.55 12.1  12.65 13.2  13.75 14.3  14.85 15.4  15.95
16.5  17.05 17.6  18.15 18.7  19.25 19.8  20.35 20.9  21.45
22. ]
```

B) Write a python program to rotate the triangle ABC by 90 degree where A[1,2],B[2,-2] and C[-1,2]

-→

```
import numpy as np
A = np.array([1, 2])
B = np.array([2, -2])
C = np.array([-1, 2])
theta = np.pi/2
rot_mat = np.array([[np.cos(theta), -np.sin(theta)],
                    [np.sin(theta), np.cos(theta)]])

A_rot = np.dot(rot_mat, A)
B_rot = np.dot(rot_mat, B)
C_rot = np.dot(rot_mat, C)

print("Rotated triangle:")
print("A':", A_rot)
print("B':", B_rot)
print("C':", C_rot)
```

output :

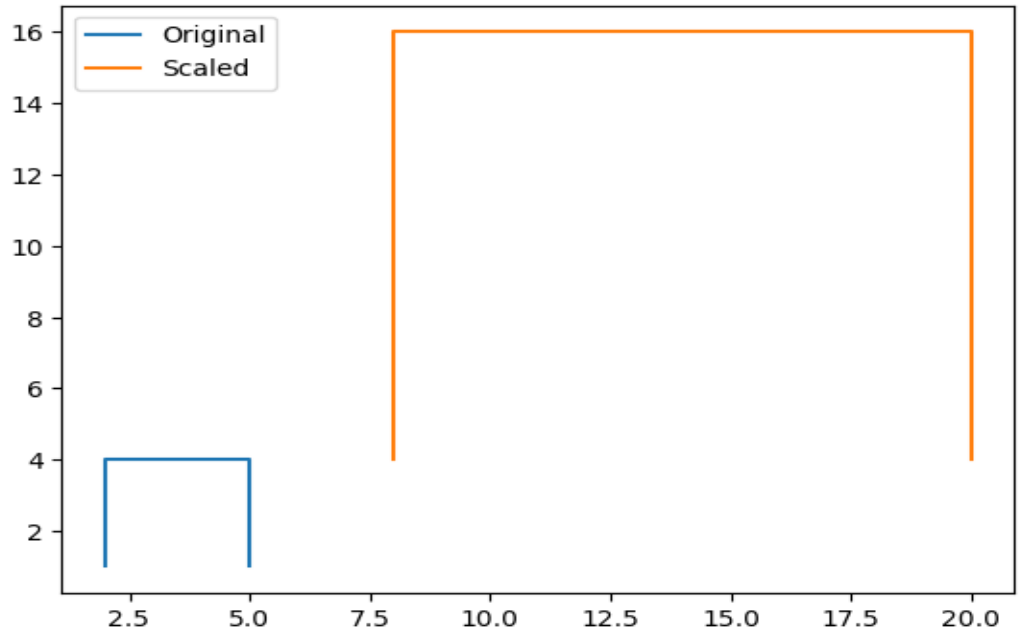
```
Rotated triangle:
A': [-2.  1.]
B': [2. 2.]
C': [-2. -1.]
```

C) Write a python program to plot the rectangle with vertices at [2,1],[2,4],[5,4],[5,1] and its uniform expansion by factor 4

-→

```
import matplotlib.pyplot as plt
import numpy as np
rectangle = np.array([[2, 1], [2, 4], [5, 4], [5, 1]])
scale_factor = 4
scale_mat = np.array([[scale_factor, 0],
                      [0, scale_factor]])
rectangle_scaled = np.dot(scale_mat, rectangle.T).T
```

```
fig, ax = plt.subplots()
ax.plot(rectangle[:, 0], rectangle[:, 1], label='Original')
ax.plot(rectangle_scaled[:, 0], rectangle_scaled[:, 1], label='Scaled')
ax.legend()
plt.show()
```



Q3) Attempt the following

A) Attempt any ONE of the following

I) Write a python program to solve the following LPP :

Min Z=x+y
Subjcet to x≥6
y≥6
x+y≤11
x,y≥0

-→

```
from scipy.optimize import linprog
c = [1, 1]
A = [[-1, 0], [0, -1], [1, 1]]
b = [-6, -6, 11]

x_bounds = (0, None)
y_bounds = (0, None)
res = linprog(c=c, A_ub=A, b_ub=b, bounds=[x_bounds, y_bounds])

if res.success:
    # Print the results
```

```
    print('Optimal value:', res.fun)
    print('x:', res.x[0])
    print('y:', res.x[1])
else:
    print('No solution found.')
```

output :

No solution found.

II) Write a python program to solve the following LPP

Max Z=2x+3y

Subject to 5x-y≥0

X+y≥6

X,y≥0

-→

```
from scipy.optimize import linprog
c = [-2, -3]
A = [[5, -1], [-1, 1]]
b = [0, 6]
x_bounds = (0, None)
y_bounds = (0, None)
res = linprog(c=c, A_ub=A, b_ub=b, bounds=[x_bounds, y_bounds])

print('Optimal value:', -res.fun)
print('x:', res.x[0])
print('y:', res.x[1])
```

output :

Optimal value: 25.5
x: 1.5
y: 7.5

B) Attempt any ONE of the following

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

A) Eirst rotation about origin through an angle π/6

B) Followed by scalin in y coordination by 4 units respectively

C) Followed by reflection through the origin

-→

```
import numpy as np
A = np.array([3, 2])
B = np.array([2, -3])

theta = np.pi/6
R = np.array([[np.cos(theta), -np.sin(theta)], [np.sin(theta), np.cos(theta)]])

S = np.array([[1, 0], [0, 4]])
F = np.array([[-1, 0], [0, -1]])
T = F @ S @ R
A_transformed = T @ A
B_transformed = T @ B

print(f"A': {A_transformed}")
print(f"B': {B_transformed}")
```

output :

```
A': [ -1.59807621 -12.92820323]
B': [-3.23205081  6.39230485]
```

II) Apply each of the following transformation on the point P[3,-1]

A) Reflection through Y-axis

B) Scaling in X and Y direction by ½ and 3 units respectively

C) Sheriang in both X and Y direction by -2 and 4 units respectively

D) Rotation about origin by an angle 60 degrees

-→

```
import numpy as np
P = np.array([3, -1])
F = np.array([[-1, 0], [0, 1]])
P_transformed = F @ P
print("After Reflection through Y-axis: ", P_transformed)

S = np.array([[0.5, 0], [0, 3]])
P_transformed = S @ P
print("After Scaling in X and Y direction by ½ and 3 units respectively: ",
P_transformed)
```

```
H = np.array([[1, -2], [4, 1]])
P_transformed = H @ P
print("After Shearing in both X and Y direction by -2 and 4 units respectively: ",
P_transformed)

theta = np.pi/3 # 60 degrees in radians
R = np.array([[np.cos(theta), -np.sin(theta)], [np.sin(theta), np.cos(theta)]])
P_transformed = R @ P

print("After Rotation about origin by an angle 60 degrees: ", P_transformed)
```

output :

```
After Reflection through Y-axis:  [-3 -1]
After Scaling in X and Y direction by ½ and 3 units respectively:  [ 1.5 -3. ]
After Shearing in both X and Y direction by -2 and 4 units respectively:  [ 5 11]
After Rotation about origin by an angle 60 degrees:  [2.3660254  2.09807621]
```