Sahakar Maharshi Bhausaheb Santuji Thorat

College Sangamner

DEPARTMENT OF COMPUTER SCIENCE

Sub : Mathematics

Remark
Demonstrator's
Signature
Date:- / /20

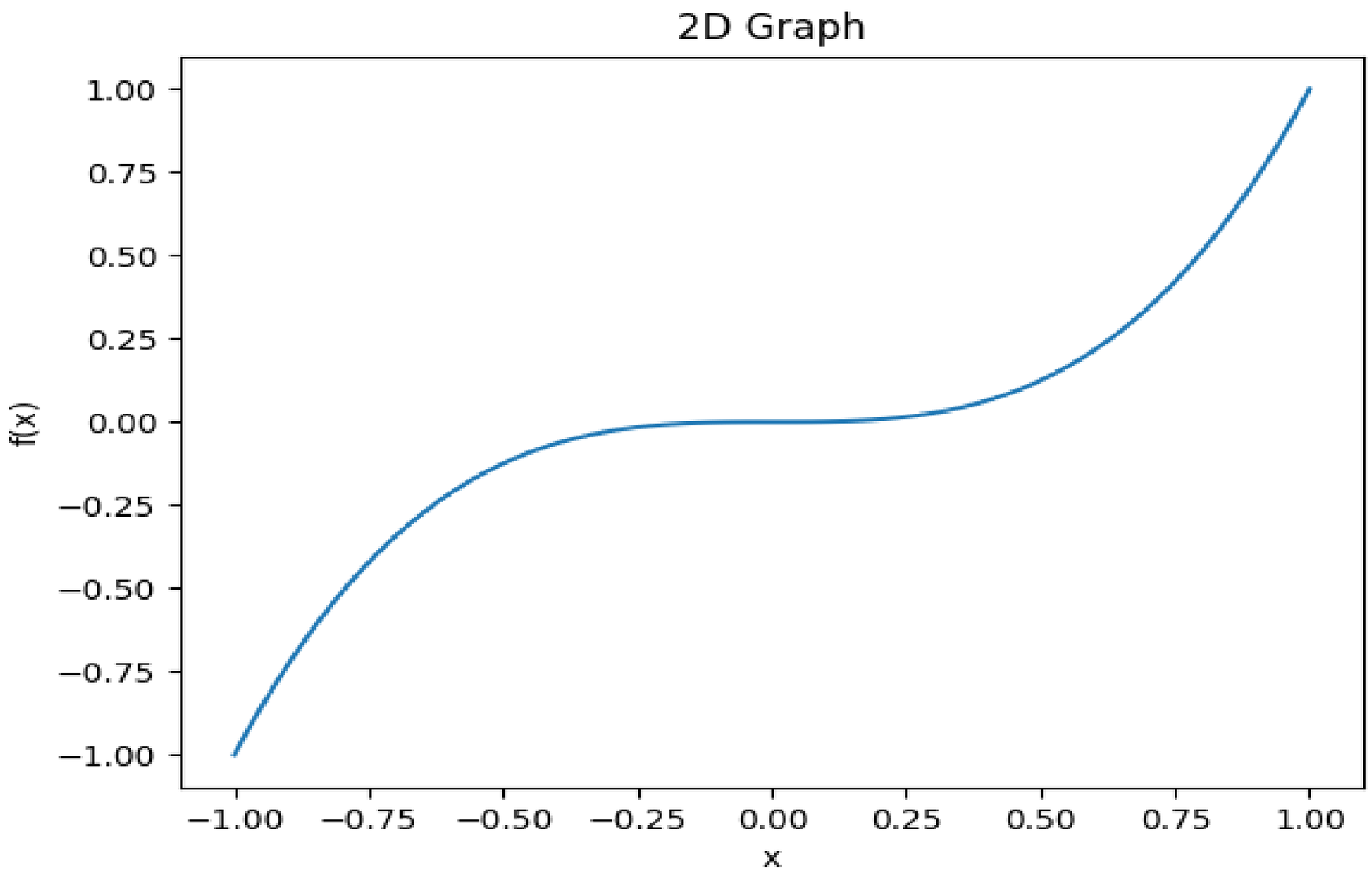
Name:- Gorde Yash Somanath. Roll.No:-_____Date:-_____
Title of the expt:-linear programming problems _Page.no:-____Class:-_____

Q1.Attempt any Two of the following

a) Write a Python program to plot 2D graph of the function $f(x)=x^3$ in [-1,1]

->

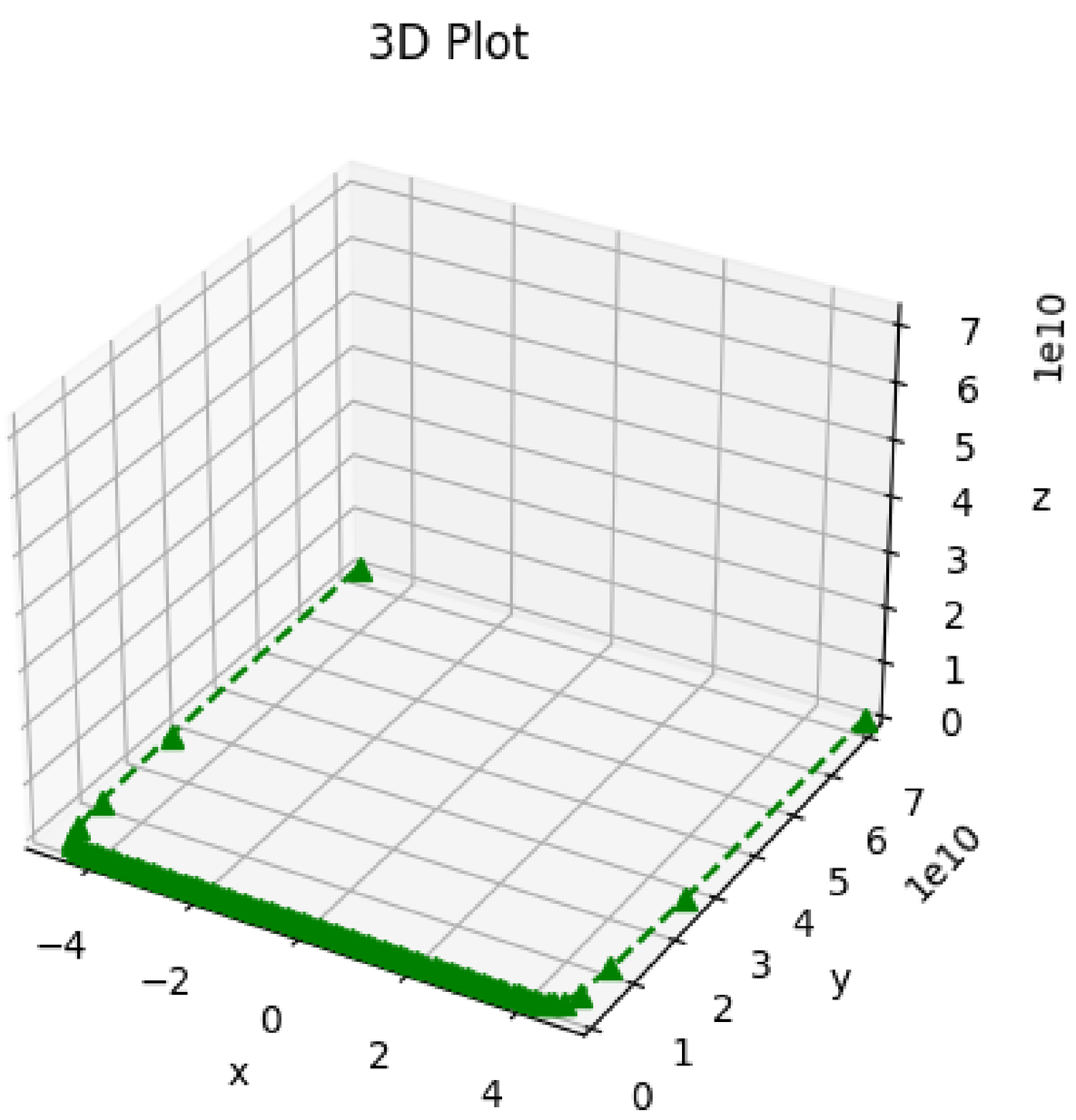
```
import matplotlib.pyplot as plt
import numpy as np
def f(x):
  return x**3
x=np.linspace(-1,1,100)
y=f(x)
plt.plot(x,y)
plt.title("2D Graph")
plt.xlabel("x")
plt.ylabel("f(x)")
plt.show()
```



b) Write a Python program to plot 3D graph of the function $f(x)=e^{-x2}$ in [-5,5] with green dashed points line with upward pointing triangle

->

```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
def f(x):
   return np.exp(x**2)
x=np.linspace(-5,5,100)
y=f(x)
fig=plt.figure()
ax=fig.add_subplot(111,projection='3d')
ax.plot(x,y,zs=0,zdir='z',color='green',linestyle='dashed',marker='^')
ax.set_xlim(-5,5)
ax.set_ylim(0,np.exp(25))
ax.set_zlim(0,np.exp(25))

ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')
ax.set_title("3D Plot")
plt.show()
```

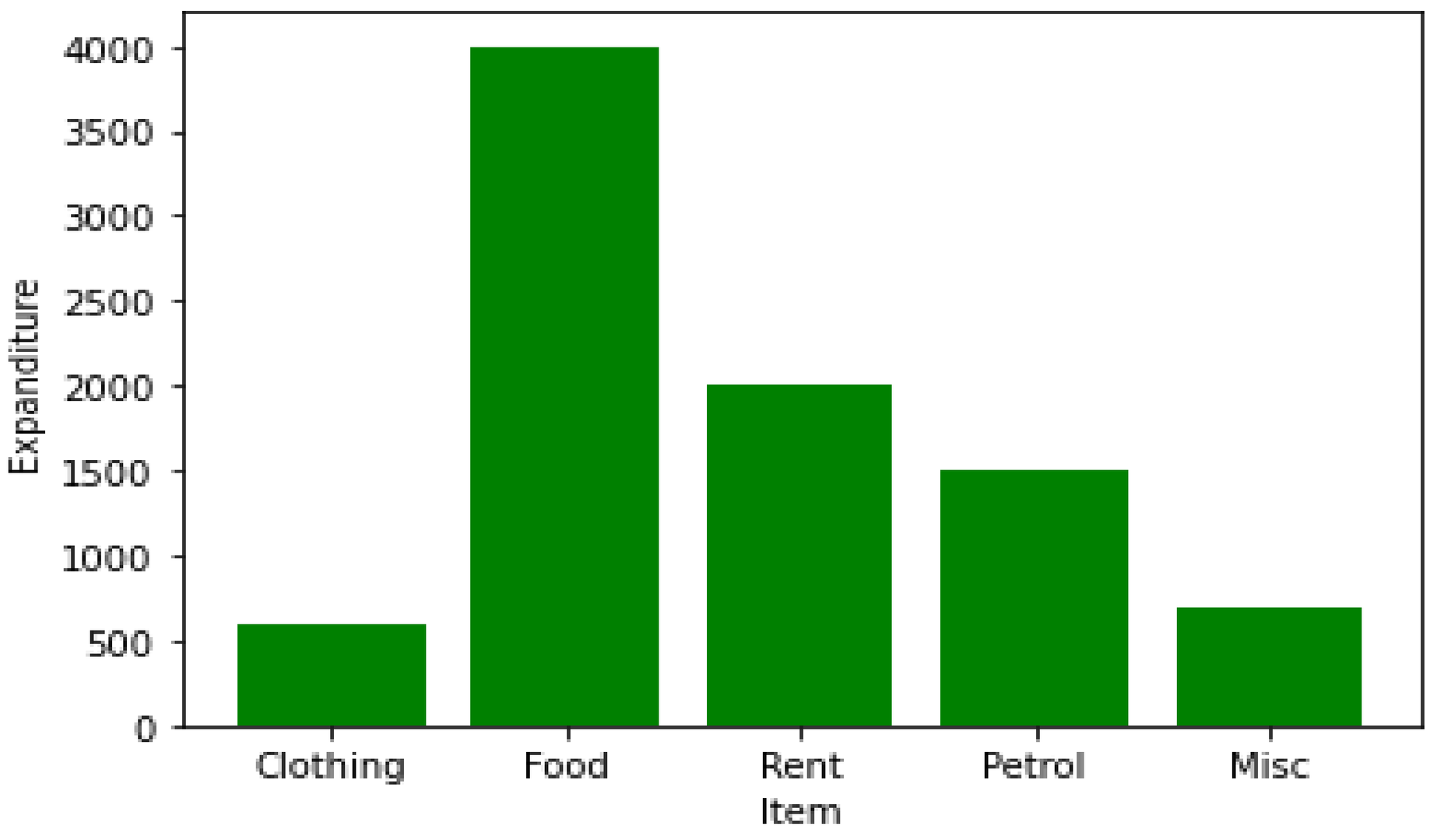


c) Using Python, Represent the following information using bar graph

Item	Clothing	Food	rent	Petrol	Misc.
Expenditure in Rs	600	4000	2000	1500	700

->

```
import matplotlib.pyplot as plt
left=[1,2,3,4,5]
height=[600,4000,2000,1500,700]
tick_label=['Clothing','Food','Rent','Petrol','Misc']
plt.bar(left,height,tick_label=tick_label,
width=0.8,color=['green'])
plt.xlabel('Item')
plt.ylabel('Expanditure')
plt.title("")
plt.show()
```



Q2) Attempt any TWO Of the Following

a) Write a python program to reflect the line segment joining the points A[5,3] and B[1,4] through the line Y=X+1

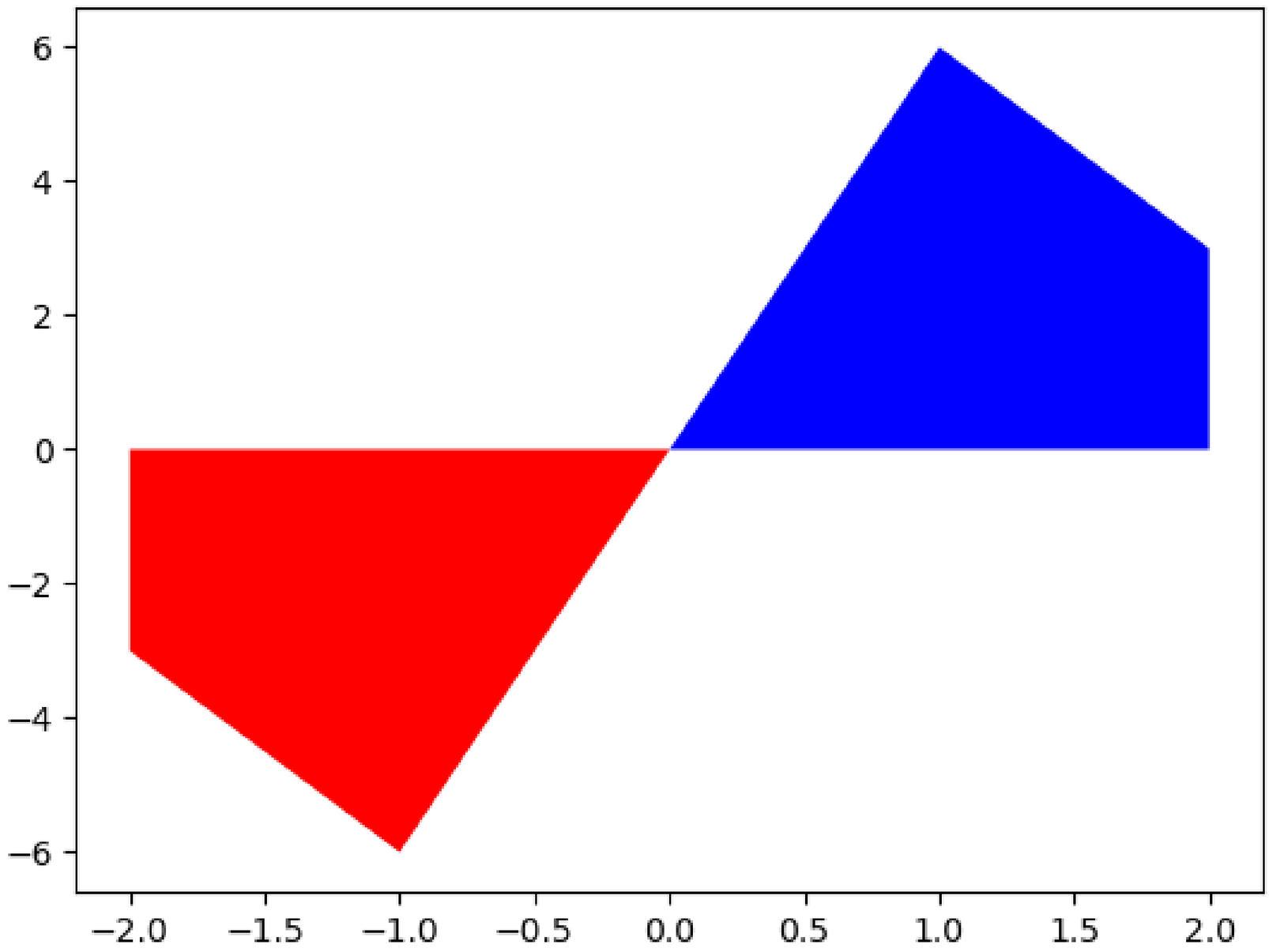
->

```
def reflect(point,line):
    x,y=point
    a,b,c=line
    x_reflected=(b**2-a**2)*x-2*a*b*y-2*a*c
    y_reflected=-2*a*b*x+(a**2-b**2)*y-2*b*c
    denominator=a**2+b**2
    x_reflected /= denominator
    y_reflected /=denominator
    return (x_reflected,y_reflected)

a,b,c=1,-1,-1
A=(5,3)
B=(1,4)
A_reflacted=reflect(A,(a,b,c))
B_reflacted=reflect(B,(a,b,c))

print("Reflectd point A :",A_reflacted)
print("Reflected point B :",B_reflacted)
```

B)Write a python program to draw a polygon with vertices (0,0),(2,0),(2,3) and(1,6) and rotate it by 180°

->

```
import numpy as np
import matplotlib.pyplot as plt
vertices=np.array([[0,0],[2,0],[2,3],[1,6]])
fig,ax=plt.subplots()
ax.fill(vertices[:,0],vertices[:,1],'blue')
theta=np.pi
rotation_matrix=np.array([[np.cos(theta),-np.sin(theta)],[np.sin(theta),np.cos(
theta)]])
rotated_vertices=np.matmul(rotation_matrix,np.transpose(vertices))
ax.fill(rotated_vertices[0],rotated_vertices[1],'red')
plt.show()
```



C)Write a python program to find the area and perimeter of the Δ ABC ,Where A[0,0],B[5,0],C[3,3]

->

```
import math
A=[0,0]
B=[5,0]
C=[3,3]

AB=math.sqrt((B[0]-A[0])**2 + (B[1]-A[1])**2)
BC=math.sqrt((C[0]-B[0])**2 +(C[1]-B[1])**2)
CA=math.sqrt((A[0]-C[0])**2 +(A[1]-C[1])**2)
perimeter=AB+BC+CA
s=perimeter /2
area=math.sqrt(s*(s-AB)*(s-BC)*(s-CA))

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

Q3) Attempt the Following

a) Attempt any one of the following

1) Write a Python program to solve the following LPP :

Max Z =150X+75Y

Subject to 4x+6y**≤ 24**

5x+3y≤15

X ≥ 0, y ≥ 0

->

```
from scipy.optimize import linprog
obj=[150,75]
lhs_eq=[[4,6],[5,3]]
rhs_eq=[24,15]
bnd=[(0,float("inf")),(0,float("inf"))]
opt=linprog(c=obj,A_eq=lhs_eq,b_eq=rhs_eq,bounds=bnd,method="highs")

print("Optimal Solution :\n",opt.x)
print("Objective function value :\n",opt.fun)
```

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

Min Z=x + y

Subject to x≥6

y≥6

x + y≤11

x≥0,y≥0

->

```
from scipy.optimize import linprog
obj=[1,1]
lhs_eq=[[1,0],[0,1],[1,1]]
rhs_eq=[6,6,11]
bnd=[(0,float("inf")),(0,float("inf"))]
opt=linprog(c=obj,A_eq=lhs_eq,b_eq=rhs_eq,bounds=bnd,method="highs")

print("Optimal Solution :\n",opt.x)
print("Objective function value :\n",opt.fun)
```

b)Attempt any ONE of the following

1)Apply python program in each of the following transformation on the point p[3,-1]
1) reflection through the X-axis
2) Scalling in x-cordinate by factor 2
3)Scalling in Y-cordinate by factor 1.5
4) reflection through the line y=x

->

```
p = [3, -1]

p_reflected = [p[0], -p[1]]
print("Point p after reflection through X-axis: ", p_reflected)

p_scaled = [2*p[0], p[1]]
print("Point p after scaling in x-coordinate by factor 2: ", p_scaled)

p_scaled = [p[0], 1.5*p[1]]
print("Point p after scaling in Y-coordinate by factor 1.5: ", p_scaled)

p_reflected = [p[1], p[0]]
print("Point p after reflection through the line y=x: ", p_reflected)
```

output :

```
 Point p after reflection through X-axis:  [3, 1]
Point p after scaling in x-coordinate by factor 2:  [6, -1]
Point p after scaling in Y-coordinate by factor 1.5:  [3, -1.5]
Point p after reflection through the line y=x:  [-1, 3]
```

2)find the combined transformation of the line segment between the point A[5,-2] & B[4,3] by using Python program for the following sequence of transformation

i) Rotation about origin through an angle π

ii) Scaling in X-cordinate by 2 units

iii) reflection through the line y=-x

iv) Shering in X direction by 4 units

->

```
import numpy as np
import matplotlib.pyplot as plt

# Define the two points as numpy arrays
A = np.array([5, -2])
B = np.array([4, 3])

# Define the transformation matrices
R = np.array([[-1, 0],
              [0, -1]])
Sx = np.array([[2, 0],
               [0, 1]])
Ref = np.array([[-1/2, 1],
                [1/2, 1]])
Shx = np.array([[1, 4],
                [0, 1]])

# Apply the transformations to the two points
A = Shx @ Ref @ Sx @ R @ A
B = Shx @ Ref @ Sx @ R @ B

# Plot the original and transformed line segments
plt.plot([5, 4], [-2, 3], 'b-', label='Original')
plt.plot([A[0], B[0]], [A[1], B[1]], 'r-', label='Transformed')
plt.legend()
plt.show()
```