Sahakar Maharshi Bhausaheb Santuji Thorat

College Sangamner

DEPARTMENT OF COMPUTER SCIENCE

Sub : Mathematics

| Remark |
| --- |
| **Demonstrator's** |
| **Signature** |
| **Date:- / /20** |

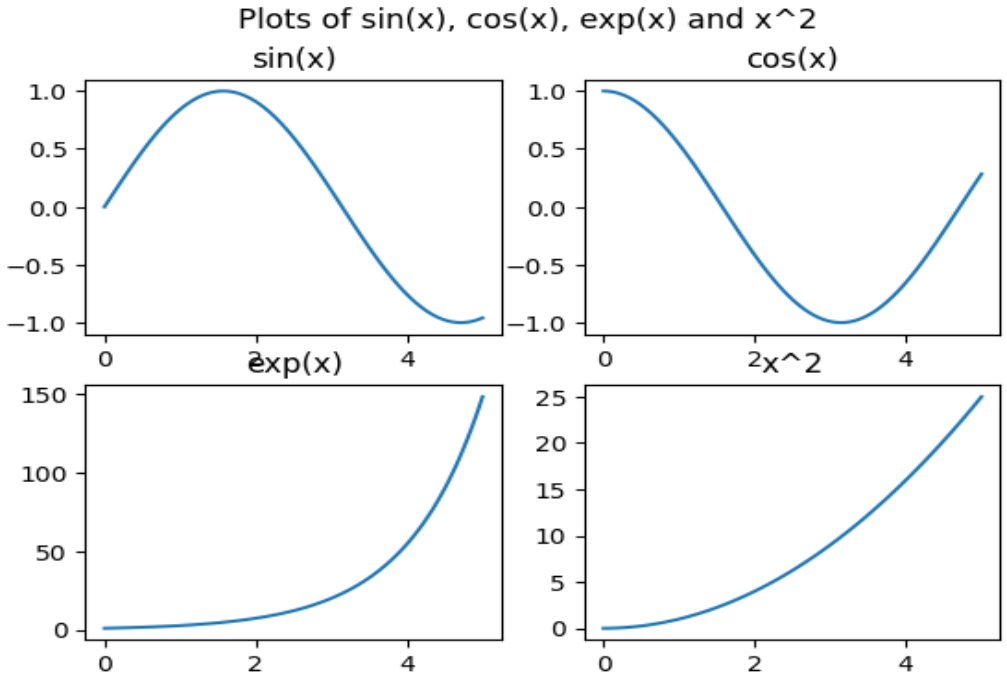
**Name:-__Gorde Yash Somnath_____ Roll.No:-_21__Date:-_____**
**Title of the expt:-__Slip no 8_____Page.no:-_____Class:-_____BCS_____**

**Q1 Attempt any TWO of the following**

**A ) Plot the graph of sin x , cos x ,$e^x$ and $x^2$ in [0,5] in one figure with (2x2) subplots.**

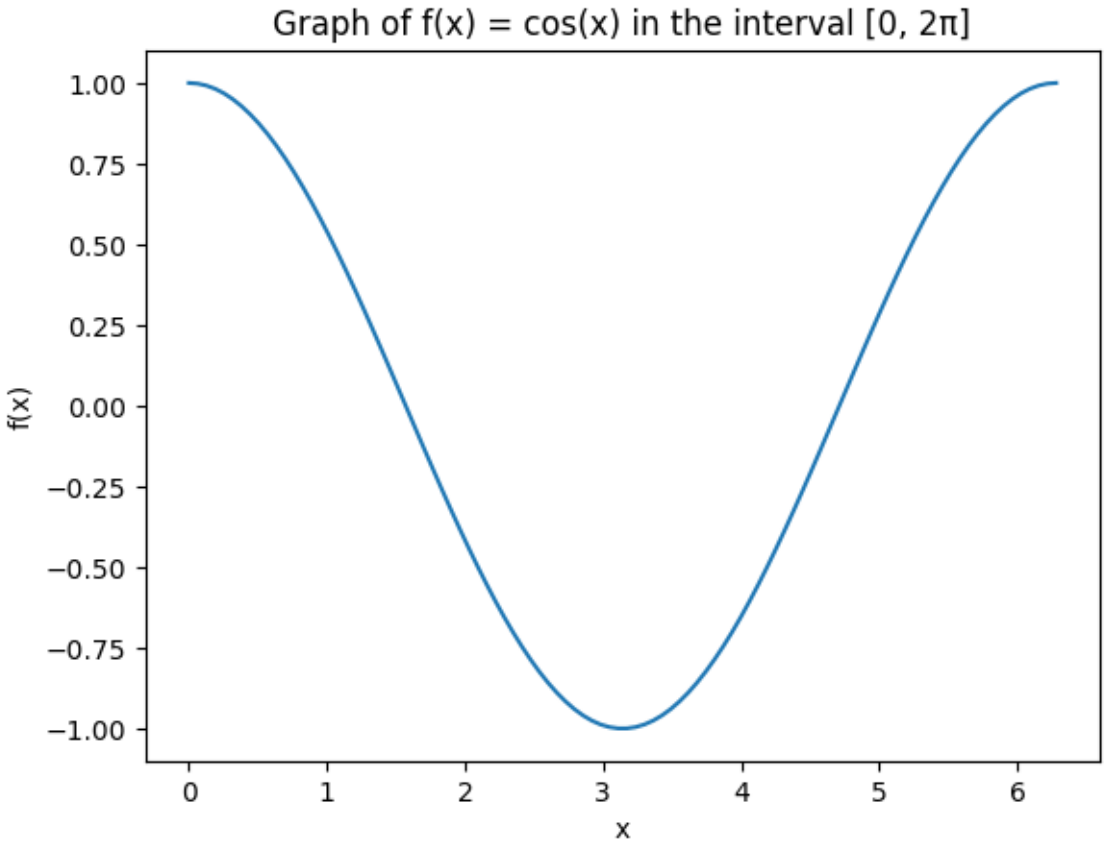
→

```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0, 5, 100)
fig, axs = plt.subplots(2, 2)
axs[0, 0].plot(x, np.sin(x))
axs[0, 0].set_title('sin(x)')
axs[0, 1].plot(x, np.cos(x))
axs[0, 1].set_title('cos(x)')
axs[1, 0].plot(x, np.exp(x))
axs[1, 0].set_title('exp(x)')
axs[1, 1].plot(x, x**2)
axs[1, 1].set_title('x^2')
fig.suptitle('Plots of sin(x), cos(x), exp(x) and x^2')
plt.show()
```



**B ) Using python plot the graph of function f(x)=cos(x) in the interval [0,2π]**
**-→**

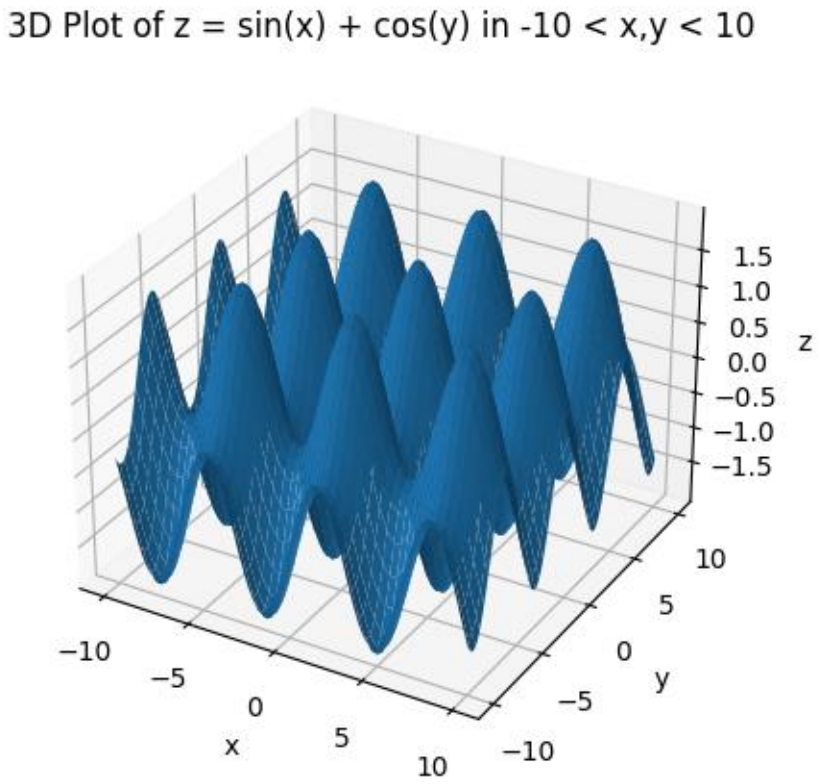
```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0, 2*np.pi, 100)
y = np.cos(x)
plt.plot(x, y)
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Graph of f(x) = cos(x) in the interval [0, 2π]')
plt.show()
```



**C ) Write a python program to generate 3D plot of the function z=sin x+ cos y in -10 <x,y<10**

**-→**

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(-10, 10, 100)
y = np.linspace(-10, 10, 100)
X, Y = np.meshgrid(x, y)
Z = np.sin(X) + np.cos(Y)

ax.plot_surface(X, Y, Z)
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')

plt.title('3D Plot of z = sin(x) + cos(y) in -10 < x,y < 10')
plt.show()
```

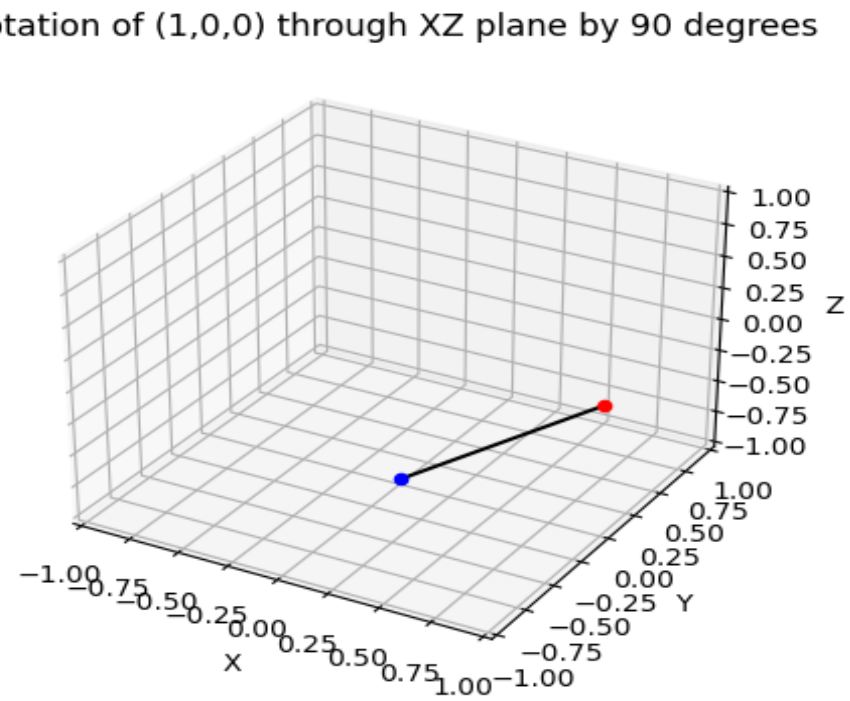


**Q2 ) Attempt any TWO of the following**

a) **Write a python program in 3D to rotate the point (1,0,0) through XZ plane in anticlockwise direction(Rotation through Y axis by an angle of 90◦**

**-→**

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()

ax = fig.add_subplot(111, projection='3d')
ax.set_xlim([-1, 1])
ax.set_ylim([-1, 1])
ax.set_zlim([-1, 1])

p = np.array([1, 0, 0, 1])
theta = np.pi/2
R = np.array([[np.cos(theta), 0, np.sin(theta), 0],
        [0, 1, 0, 0],
        [-np.sin(theta), 0, np.cos(theta), 0],
        [0, 0, 0, 1]])
q = R @ p

ax.scatter(p[0], p[1], p[2], color='red')
ax.scatter(q[0], q[1], q[2], color='blue')
ax.plot([p[0], q[0]], [p[1], q[1]], [p[2], q[2]], color='black')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

plt.title('Rotation of (1,0,0) through XZ plane by 90 degrees')
plt.show()
```

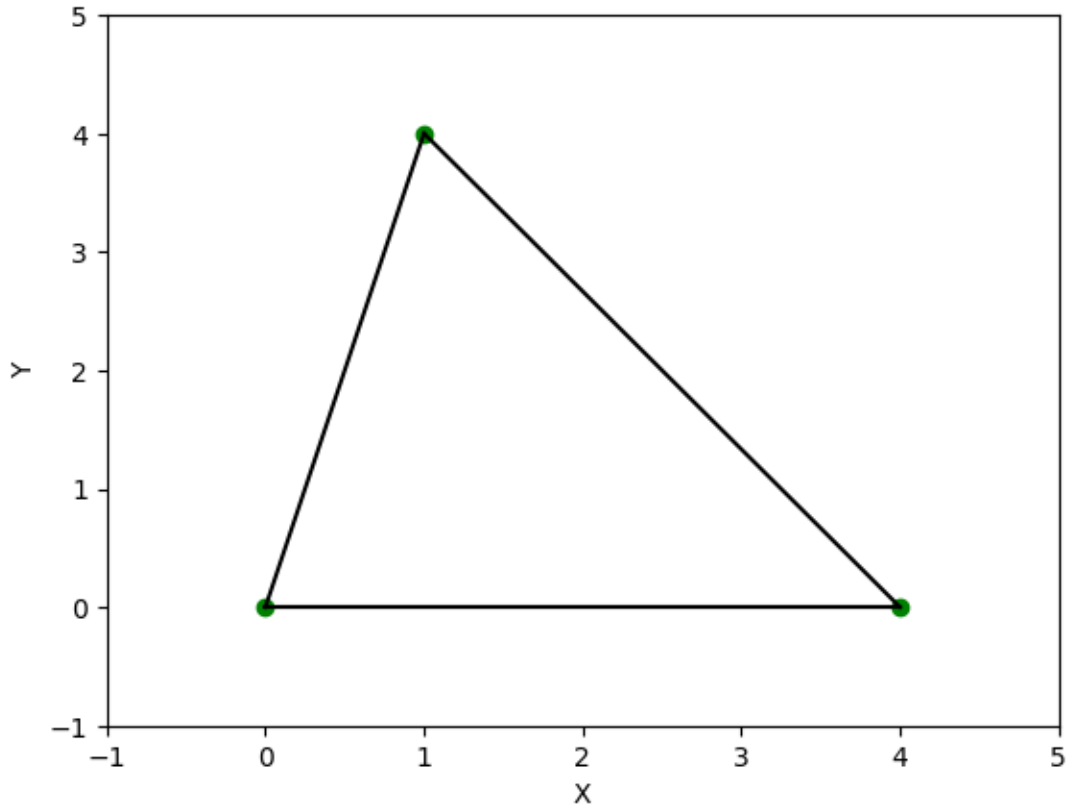



**B ) Using python generate tringle with vertices (0,0),(4,0),(1,4) check whether the tringle is Scalene triangle**

**-→**

```
import math
import matplotlib.pyplot as plt
A = [0, 0]
B = [4, 0]
C = [1, 4]

a = math.sqrt((B[0]-C[0])**2 + (B[1]-C[1])**2)
b = math.sqrt((C[0]-A[0])**2 + (C[1]-A[1])**2)
c = math.sqrt((A[0]-B[0])**2 + (A[1]-B[1])**2)

if a != b and b != c and c != a:
    print("The triangle is a Scalene triangle.")
    color = 'green'
else:
    print("The triangle is not a Scalene triangle.")
    color = 'red'

fig, ax = plt.subplots()
ax.plot([A[0], B[0]], [A[1], B[1]], color='black')
ax.plot([B[0], C[0]], [B[1], C[1]], color='black')
```

```
ax.plot([C[0], A[0]], [C[1], A[1]], color='black')
ax.scatter([A[0], B[0], C[0]], [A[1], B[1], C[1]], color=color)
ax.set_xlim([-1, 5])
ax.set_ylim([-1, 5])
ax.set_xlabel('X')
ax.set_ylabel('Y')

plt.show()
```




**C ) Write a python program to find the area and perimeter of the ΔABC , where A[0,0],B[6,0],C[4,4]**

**-→**

```
import math
A = [0, 0]
B = [6, 0]
C = [4, 4]

a = math.sqrt((B[0]-C[0])**2 + (B[1]-C[1])**2)
b = math.sqrt((C[0]-A[0])**2 + (C[1]-A[1])**2)
c = math.sqrt((A[0]-B[0])**2 + (A[1]-B[1])**2)
perimeter = a + b + c
s = perimeter / 2
```

```
area = math.sqrt(s * (s - a) * (s - b) * (s - c))
print("The area of the triangle is:", area)
print("The perimeter of the triangle is:", perimeter)
```

output

```
The area of the triangle is: 11.999999999999996
The perimeter of the triangle is: 16.12899020449196
```

**Q3) 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
obj = [-150, -75]
lhs = [[4, 6], [5, 3]]
rhs = [24, 15]
bounds = [(0, None), (0, None)]
res = linprog(c=obj, A_ub=lhs, b_ub=rhs, bounds=bounds, method='simplex')
print("The maximum value of Z is:", -res.fun)
print("The values of x and y that maximize Z are:", res.x)
```

output :

```
The maximum value of Z is: 450.0
The values of x and y that maximize Z are: [3. 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=3x+5y+4z**
**Subject to 2x+3y≤8**
**2y+5z≤10**
**3x+2y+4z≤15**
**X,y,z≥0**

**-→**

```
import pulp
prob = pulp.LpProblem('LPP', pulp.LpMaximize)

x = pulp.LpVariable('x', lowBound=0)
y = pulp.LpVariable('y', lowBound=0)
z = pulp.LpVariable('z', lowBound=0)

prob += 3*x + 5*y + 4*z
prob += 2*x + 3*y <= 8
prob += 2*y + 5*z <= 10
prob += 3*x + 2*y + 4*z <= 15
prob.solve()

print('Optimal Solution:')
print('Z =', pulp.value(prob.objective))
print('x =', pulp.value(x))
print('y =', pulp.value(y))
print('z =', pulp.value(z))
```

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

**I ) Apply Python program in each of the following transformation on the point P[4,-2]**
**A ) Refection trough Y-axis**
**B ) Scaling in X-coordinate by factor 5**
**C ) Rotation about origin through an angle $\frac{\pi}{2}$**
**D ) Shering in X direction by $\frac{7}{2}$ units**

**-→**

```
import numpy as np
# reflection through Y-axis
p = np.array([4, -2])
R = np.array([[-1, 0],
              [0, 1]])
q = R @ p
print(q)

# scaling in X-coordinate by factor 5
```

```
p = np.array([4, -2])
S = np.array([[5, 0],
              [0, 1]])
q = S @ p
print(q)

# rotation about origin through an angle π/2
p = np.array([4, -2])
theta = np.pi/2
R = np.array([[np.cos(theta), -np.sin(theta)],
              [np.sin(theta), np.cos(theta)]])
q = R @ p
print(q)

# shearing in X direction by 7/2 units
p = np.array([4, -2])
a = 7/2
S = np.array([[1, a],
              [0, 1]])
q = S @ p
print(q)
```

output :

```
[-4 -2]
[20 -2]
[2. 4.]
[-3. -2.]
```

**II) Find the combined transformation of the line segment between the point A[7,-2] & B[6,2] by using python program for the following sequence of transformation**

**A ) Rotation about origin through an angle $\frac{\pi}{3}$**

**B ) Scaling in X-cordinate by 7 units**

**C ) Uniform scaling by -4 units**

**D ) reflection through the line X-axis**

-→

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

A = np.array([7,-2])
```

```
B = np.array([6,2])
theta = np.pi/3
Sx = 7
Sy = -4
R = np.array([[np.cos(theta), -np.sin(theta)], [np.sin(theta),
np.cos(theta)]])
S = np.array([[Sx, 0], [0, Sy]])
T = np.array([[1, 0], [0, -1]])
A = T @ S @ R @ A
B = T @ S @ R @ B

plt.plot([A[0], B[0]], [A[1], B[1]], color='blue')
plt.xlim(-30, 30)
plt.ylim(-30, 30)
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Transformed Line Segment')
plt.show()
```