```python
# ------------------------------------------------------------
# -----------------------Probablity --------------------------
# ------------------------------------------------------------
import matplotlib.pyplot as plt
import numpy as np
import math
import random
# ------------------------------------------------------------
random_number = random.random()
print(random_number)

outcome = random.randint(1,6)
print(outcome)

a1 =[ random.randint(1, 6) for _ in range(10) ]
print(a1)

print(random.sample(range(1, 50), 6))

a = np.random.random((1000))
# ------------------------------------------------------------

plt.figure(1)
plt.hist(a, bins=10)

mu, sigma = 0, 0.1
s = np.random.normal(mu, sigma, 1000)

plt.figure(2)
plt.hist(s, bins=10,histtype='bar')


s1 = np.random.randn( 1000)

plt.figure(3)
plt.hist(s, bins=10,histtype='bar')
plt.show()
# ------------------------------------------------------------
# -------------Linear Algebra --------------------------------
# ------------------------------------------------------------

V1 = np.array([1,2])
print(V1)
print(V1.shape)

V2 = np.array([1,2])
print(V2)
print(V2.shape)

V3 = np.dot(V1, V2)

V4 = np.multiply(V1,V2)
print(V4)

V5 = np.arange(15).reshape(3, 5)
print(V5)
print(V5.ndim)

V6 = np.array([[1, 2, 3, 4, 5],
               [1, 2, 3, 4, 5],
               [1, 2, 3, 4, 5]])

V7 = np.dot(V5, np.transpose(V6))
V8 = np.matmul(V5,np.transpose(V6))

V9 = np.array([1, 2, 3, 4])
V10 = V9.reshape((-1, 1)) # Trick for 1 dim

V11 = np.array([(1, 2, 3, 4, 5),
                (1, 2, 3, 4, 5),
                (1, 2, 3, 4, 5)])

print(V7)

# ------------------------------------------------------------
# -------------Symbolic  -------------------------------------
# ------------------------------------------------------------

from sympy import *
x = Symbol('x')
y = Symbol('y')

f1 = (x+y)**2
print(f1)

f2 = expand((x+y)**3)
print(f2)

f3 = simplify((x+x*y)/x)
print(f3)


l1 = limit(x, x, oo)
print(l1)

l2 = limit(1/x, x, oo)
print(l2)

l3 = limit(x**x, x, 0)
print(l3)

l4 =limit((tan(x+y)-tan(x))/y, y, 0)
print(l4)


```

```python
d1 = diff(sin(x), x)
print(d1)

d2 = diff(sin(2*x), x)
print(d2)



s1 = series(cos(x), x)
print(s1)

s2 = series(1/cos(x), x)
print(s2)

i1 = integrate(6*x**5, x)
print(i1)

i2 = integrate(sin(x), x)
print(i2)

i3 = integrate(2*x + sinh(x), x)
print(i3)

i4 = integrate(x**3, (x, -1, 1))
print(i4)

i5= integrate(sin(x), (x, 0, pi/2))
print(i5)


i6 = integrate(cos(x), (x, -pi/2, pi/2))
print(i6)

e1 = solve(x**4 - 1, x)
print(e1)
```

```python

e2 = solve([x + 5*y - 2, -3*x + 6*y - 15], [x, y])
print(e2)

f = x**4 - 3*x**2 + 1
factor(f)



from sympy import Matrix

M1 = Matrix([[1,0], [0,1]])
print(M1)

M2 = Matrix([[1,x], [y,1]])
print(M2)
# ------------------------------------------------------------
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter
import numpy as np


fig = plt.figure(4)
ax = fig.gca(projection='3d')

# Make data.
X = np.arange(-5, 5, 0.25)
Y = np.arange(-5, 5, 0.25)

X, Y = np.meshgrid(X, Y)
R = np.sqrt(X**2 + Y**2)

Z = np.sin(R)
```

```

surf = ax.plot_surface(X, Y, Z)

# Plot the surface.
# surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,  linewidth=0, antialiased=False)

# Customize the z axis.
# ax.set_zlim(-1.01, 1.01)
# ax.zaxis.set_major_locator(LinearLocator(10))
# ax.zaxis.set_major_formatter(FormatStrFormatter('%.02f'))

# Add a color bar which maps values to colors.
# fig.colorbar(surf, shrink=0.5, aspect=5)

plt.show()
```