Question 1: List: List is a non-homogeneous data structure which stores the elements in single row and multiple rows and *columns*

```
List = [(1,2,"b","a"), (1,2,3,4)]
print(List)
```

```
[(1, 2, 'b', 'a'), (1, 2, 3, 4)]
```

Tuple: Tuple is immutable & Ordered

```
Tuple = (1,2,"a")
print("Tuple", Tuple)
```

```
Tuple (1, 2, 'a')
```

Dictionary: Dictionary is also a non-homogeneous data structure which stores key value pairs

```
D = {1: "a", 2: "b", 3:"c"}
print(D)
```

```
{1: 'a', 2: 'b', 3: 'c'}
```

Set: Set data structure is also non-homogeneous data structure but stores in single row. Doesn't allow duplicate elements

```
S = set()
S.add(1)
S.add(2)
S.add(2)
print(S)
```

```
{1, 2}
```

Array:

```
import array as arr
Array1 = arr.array('i', [1,2,3,4])
print(Array1)
```

```
array('i', [1, 2, 3, 4])
```

Question 2: Create the students name, roll numbers, subjects in dictionary.

```
Dict1 = {'Name': 'John', 'Roll No.': 1, 'Subjects': 'Algebra'}
print(Dict1)
print(Dict1["Name"])
```

```
{'Name': 'John', 'Roll No.': 1, 'Subjects': 'Algebra'}
John
```

Question 3: Create a list of strings & print the first latter of each string.

```
string = ["india", "China", "USA", "UK"]
first = ([country[0] for country in string])
print(first)
```

```
['i', 'C', 'U', 'U']
```

Question 4: Use the python method- append(), extend(), index(), pop(), count(), remover(), reverse(), len(), update(), add().

```
#Do it yourself
```

Question 5: Execute several mathematical operations in set such as Union, Intersection and Symmetric Difference.

```
s1 = {1,2,3,4,5,6,7,8}
s2 = {1,4,7,'a', 'b'}
s3 = {'a', 'c' , 'e', 5, 6}

U = s1.union(s2)
I = s1.intersection(s2)
Diff = s2.difference(s3)
print(U)
print(I)
print(Diff)
```

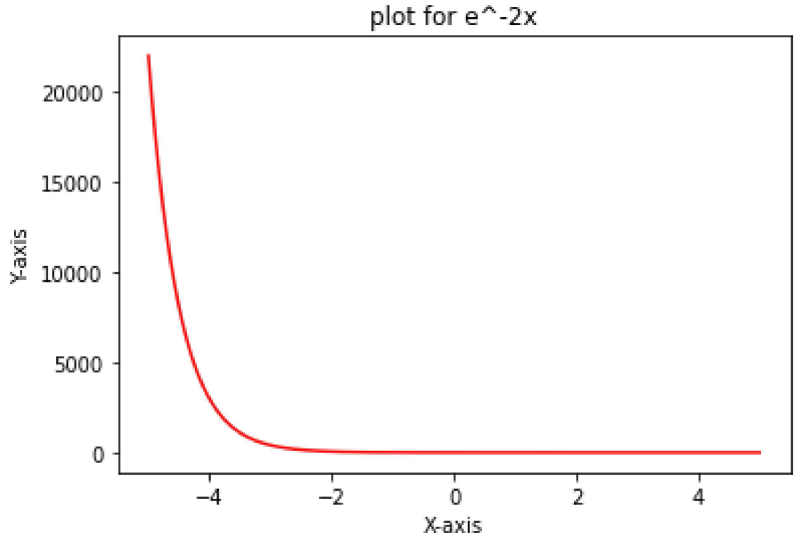
```
{1, 2, 3, 4, 5, 6, 7, 8, 'b', 'a'}
{1, 4, 7}
{1, 'b', 4, 7}
```

Question 6: Plot the functions F(x) = e^(−2x) , G(x) = x^2 .

```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(-5, 5, 1000)
y = np.exp(-2*x)
```

```
plt.plot(x,y,'r')
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("plot for e^-2x")
plt.show
```

```
<function matplotlib.pyplot.show>
```



Question 7: Generate the random variables of following distributions,

(a) Poisson

(b) Normal

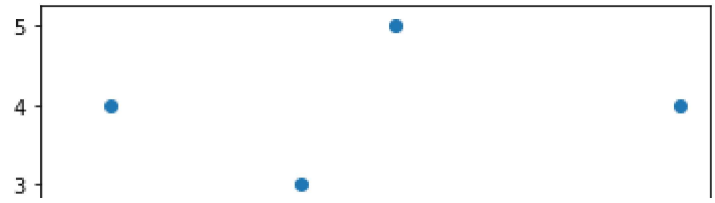
(c) Exponential

(d) Uniform

find the mean, variance, median, mode and also display the histogram, scatter plot of generated random variable.

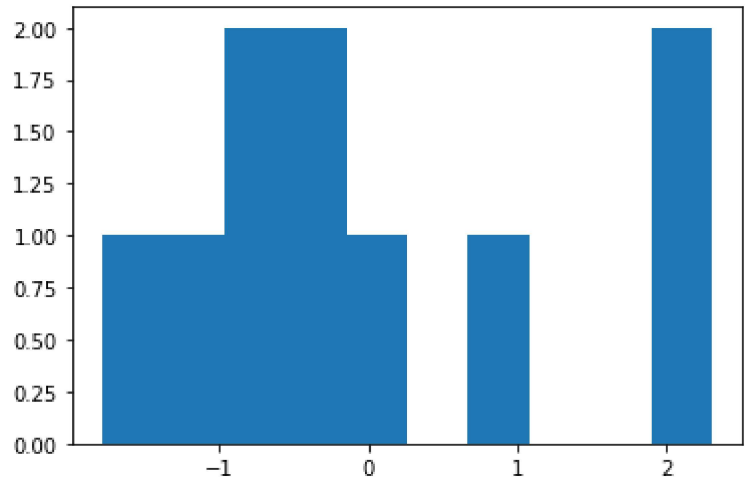
```
Z1=np.random.normal(0,1,10)
z2=np.random.poisson(2, 10)
print(Z1)
print(z2)
print("the scatter plot is the following:")
plt.scatter(Z1,z2) ## scatter plot
```

```
[ 0.28746332  0.6650343   0.81367096 -0.78022825 -1.00430067  2.41561512
  0.92212575  0.47188687  0.93008989 -0.7176145 ]
[3 2 5 4 0 4 1 1 2 2]
the scatter plot is the following:
<matplotlib.collections.PathCollection at 0x7ff216c32650>
```




```
z3= np.random.exponential(3,10)
print("the histogram is the following:")
plt.hist(Z1)
```

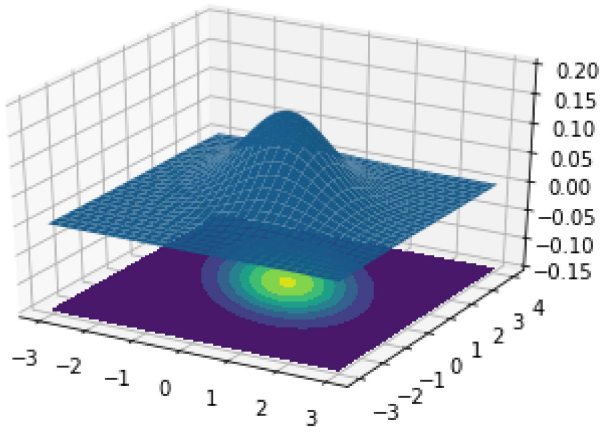
```
the histogram is the following:
(array([1., 1., 2., 2., 1., 0., 1., 0., 0., 2.]),
 array([-1.78660612, -1.37701011, -0.96741411, -0.5578181 , -0.14822209,
         0.26137392,  0.67096993,  1.08056593,  1.49016194,  1.89975795,
         2.30935396]),
 <a list of 10 Patch objects>)
```



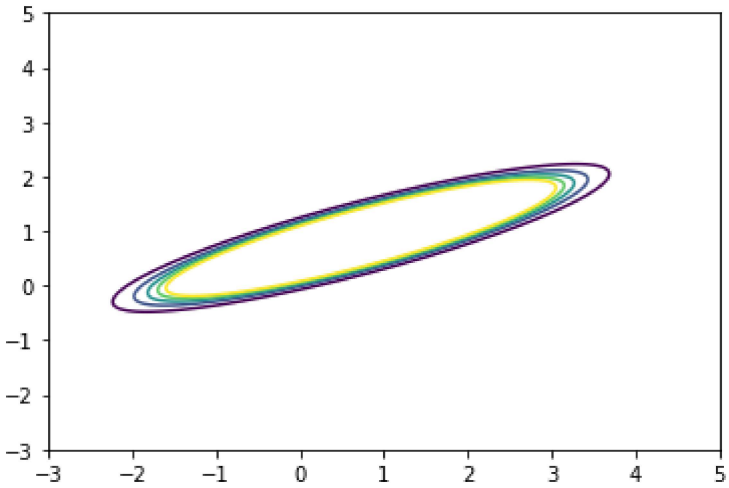
Question 8: Plot a contour and 3D plot of the bi-variate Gaussian distribution.

```
 # Create a surface plot (3-D mesh plot) and projected filled contour plot under it
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
N = 60 #for D=2 dimensional variable X and Y for bi-variate gaussian distribution
X = np.linspace(-3, 3, N)
Y = np.linspace(-3, 4, N)
X, Y = np.meshgrid(X, Y)
mu = np.array([0., 1.]) # for mean vector and covarience matrix
Sigma = np.array([[ 1. , -0.5], [-0.5, 1.5]])
com = np.empty(X.shape + (2,)) # combine X and Y
com[:, :, 0] = X
com[:, :, 1] = Y
```

```
def multivariate_gaussian(com, mu, Sigma): # the bi-variate Gaussian distribution .
  n = mu.shape[0]
  Sigma_det = np.linalg.det(Sigma)
  Sigma_inv = np.linalg.inv(Sigma)
  N = np.sqrt((2*np.pi)**n * Sigma_det)
  fac = np.einsum('...k,kl,...l->...', com-mu, Sigma_inv, com-mu) #einsum call calculates (x-
  return np.exp(-fac / 2) / N
Z = multivariate_gaussian(com, mu, Sigma) #The distribution on the variables X, Y
fig = plt.figure()
ax = plt.axes(projection='3d')
ax.plot_surface(X, Y, Z)
ax.contourf(X, Y, Z, offset=-0.15)
ax.set_zlim(-0.15,0.2) #Adjust the limits
plt.show()
```




```
 # Plot a contour
import numpy as np
from scipy.stats import multivariate_normal as mvn
import matplotlib.pyplot as plt
D = 2
x = np.random.rand(D)
mu = np.random.rand(D)
A = np.random.rand(D,D)
# random symmetric matrix
cov = A.T.dot(A)
# Generate grid points
x, y = np.meshgrid(np.linspace(-3,5,1000),np.linspace(-3,5,1000))
xy = np.column_stack([x.flat, y.flat])
# density values at the grid points
Z = mvn.pdf(xy, mu, cov).reshape(x.shape)
# arbitrary contour levels
contour_level = [.01, .02, .03, .04, .05]
fig = plt.contour(x, y, Z, levels = contour_level)
```


5
4
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3
4
5