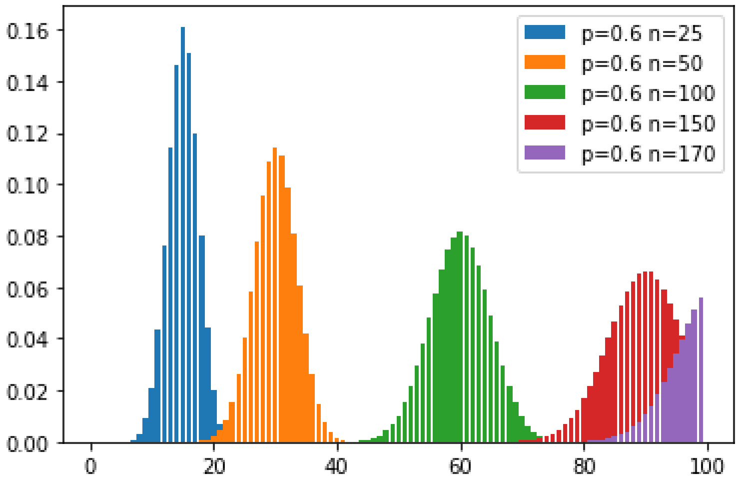
BINOMIAL DISTRIBUTION

In [4]:

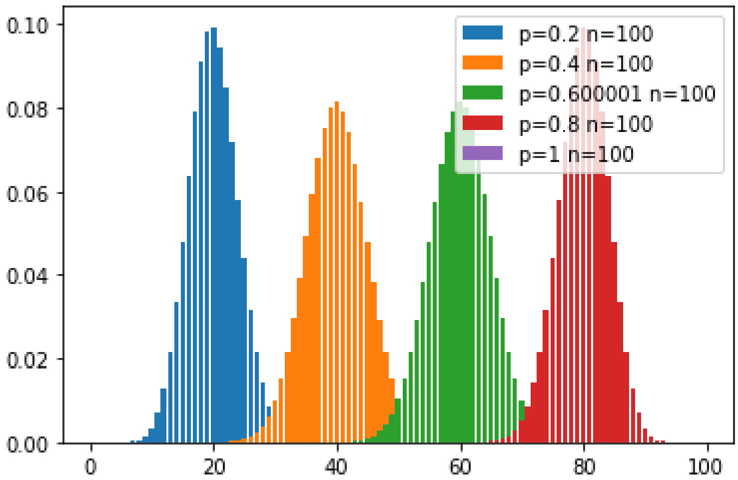
```
print("##### FIRST PART")
def fact(n):
    val=1
    for i in range(1,n+1):
        val=val*i
    return val

import numpy as np
import matplotlib.pyplot as plt
from matplotlib import pyplot
n_var=[25,50,100,150,170]#number of trails
x_var=list(range(1,100))#num of success
p_var=[0.2,0.4,0.600001,0.8,1]#considered varying p values
p=0.6
xPoints=[]#just an empty array to store values and then plot
yPoints=[]#just an empty array to store values and then plot
#since parameters of the distribution are n and p
#Here n is varying calculating and p is fixed the PDF
print("WHEN n VARYING and p fixed THE DISTRIBUTION PLOT IS:")
for i in range(len(n_var)):
    n=n_var[i]
    for j in x_var:
        prob=(fact(n)*(p**j)*(1-p)**(n-j))/(fact(j)*fact(n-j))
        yPoints.append(prob)
    pyplot.bar(x_var, yPoints,label='p=0.6 n={}'.format(n))
    yPoints=[]
plt.legend()
plt.show()

#here n fixed and p is varying
print("WHEN p VARYING and n fixed THE DISTRIBUTION PLOT IS:")
n=100
for i in range(len(p_var)):
    p=p_var[i]
    for j in x_var:
        prob=fact(n)/(fact(j)*fact(n-j))*(p)**j*(1-p)**(n-j)
        xPoints.append(prob)
    pyplot.bar(x_var, xPoints,label='p={} n=100'.format(p))
    xPoints=[]
plt.legend()
plt.show()

print("WHEN n VARYING and p fixed Mean and Variance vary as:")
p=0.6
meanArray=[]
var=[]
for i in range(len(n_var)):
    n=n_var[i]
    meanArray.append(n*p)
    var.append(n*p*(1-p))
plt.plot(n_var,meanArray,label='p fixed n varying, mean graph')
plt.plot(n_var,var,label='p fixed n varying, variance graph')
plt.legend()
plt.show()

print("WHEN p VARYING and n fixed Mean and Variance vary as:")
n=12
meanArray=[]
var=[]
```

```
for i in range(len(p_var)):
    p=p_var[i]
    meanArray.append(n*p)
    var.append(n*p*(1-p))
plt.plot(p_var,meanArray,label='n fixed p varying, mean graph')
plt.plot(p_var,var,label='n fixed p varying, variance graph')
plt.legend()
plt.show()
```

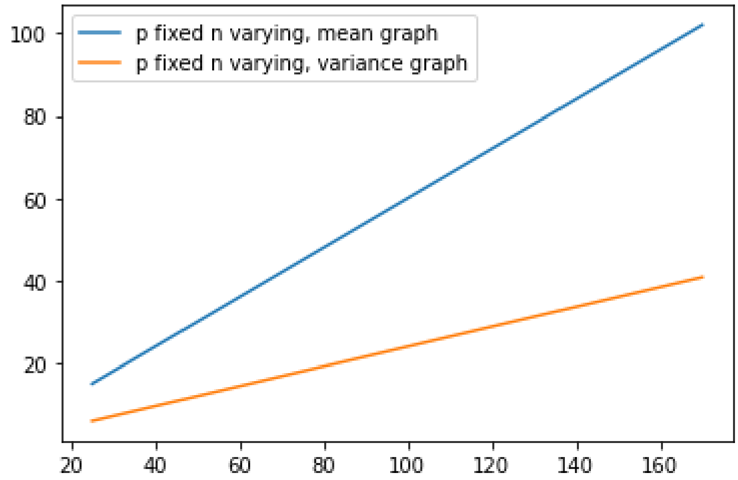
```
##### FIRST PART
WHEN n VARYING and p fixed THE DISTRIBUTION PLOT IS:
```



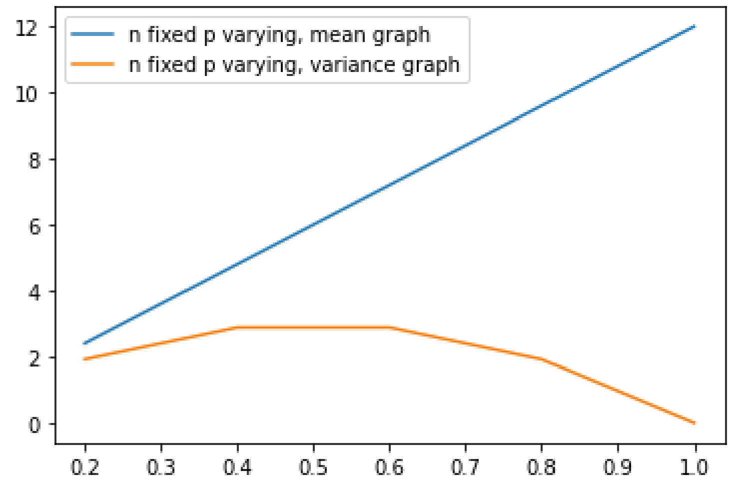

```
WHEN p VARYING and n fixed THE DISTRIBUTION PLOT IS:
```




```
WHEN n VARYING and p fixed Mean and Variance vary as:
```



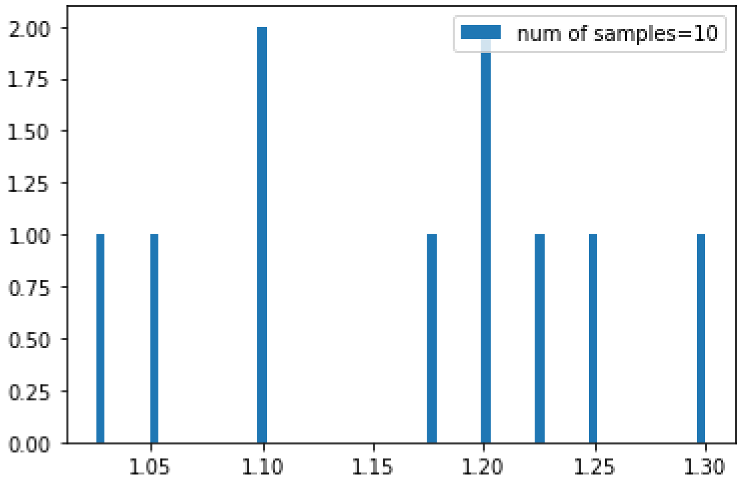

```
WHEN p VARYING and n fixed Mean and Variance vary as:
```

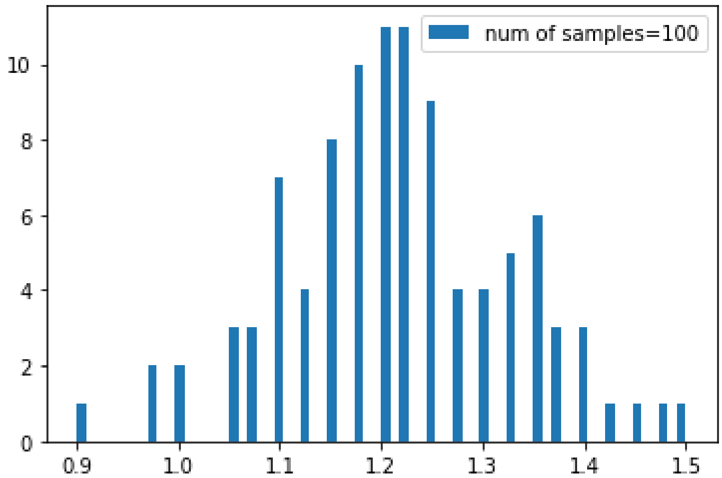


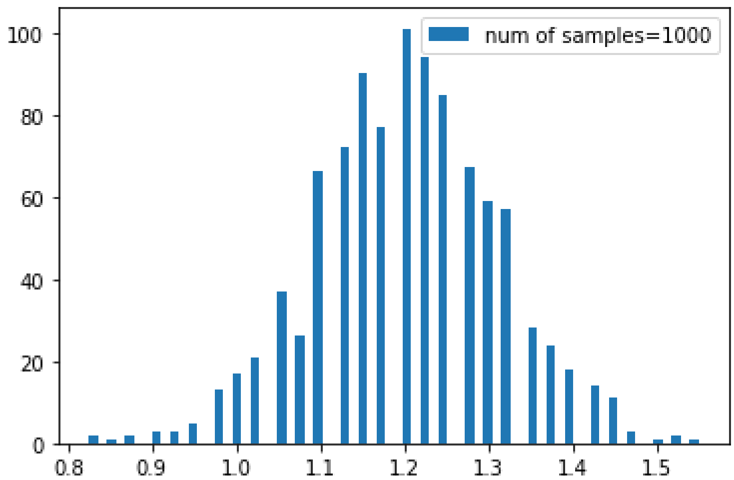
In [6]:

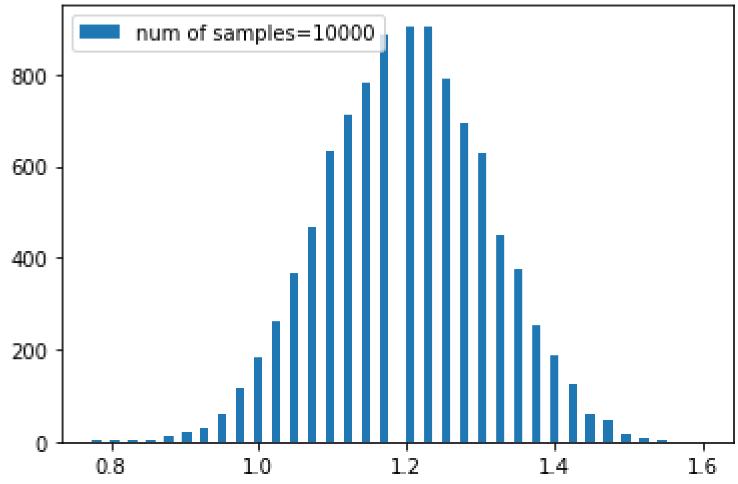
```
print("SECOND PART: To prove CLT")

def fact(n):
    return 1 if (n==1 or n==0) else n * fact(n - 1);

import numpy as np
import matplotlib.pyplot as plt
from matplotlib import pyplot
nt=40#number of trails
x_var=list(range(n + 1))#num of success
p=0.6
ns=[10,100,1000,10000]
xPoints=[]
sampleMean=[]
yPoints=[]
#Here num of times we perform for a sample i.e number of samples is varying
#Generating sample mean plot while varying number of samples
for i in range(4):
    for k in range(ns[i]):
        temp=np.random.binomial(2,p,40)
        sum1=0
        for m in temp:
            sum1=sum1+m
        sampleMean.append(sum1/40)
    plt.hist(sampleMean,68,label="num of samples={}".format(k+1))
    sampleMean=[]
    plt.legend()
    plt.show()
```

SECOND PART: To prove CLT







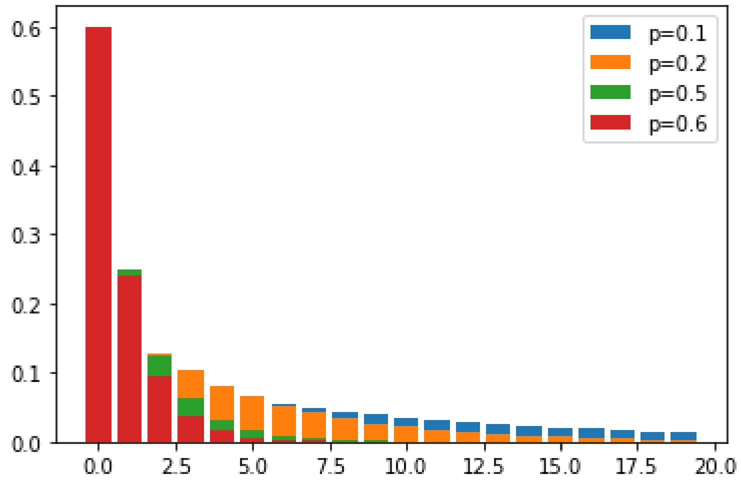


Geometric

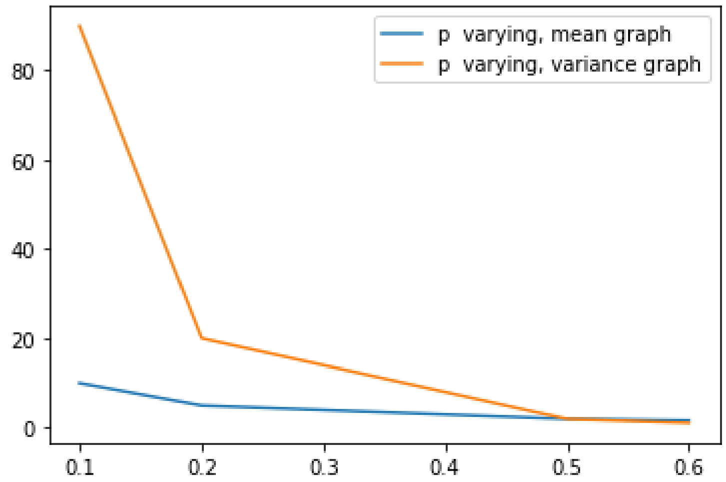
In [8]:

```
print("FIRST PART")
import numpy as np
import matplotlib.pyplot as plt

x_var=list(range(20))
p_var=[0.1,0.2,0.5,0.6]

#Here parameter is p
#When p is varying the distribution will be
for i in range(len(p_var)):
    p=p_var[i]
    for j in x_var:
        prob=((1-p)**(j))*p
        xPoints.append(prob)

    plt.bar(x_var, xPoints,label='p={}'.format(p))
    xPoints=[]
plt.legend()
print('DISTRIBUTION WHEN p IS VARYING:')
plt.show()

#When p is varying mean and variance  will vary as
mean=[]
var=[]
for i in range(len(p_var)):
    p=p_var[i]
    mean.append(1/p)
    var.append((1-p)/(p*p))
plt.plot(p_var,mean,label='p  varying, mean graph')
plt.plot(p_var,var,label='p  varying, variance graph')
plt.legend()
print('MEAN AND VARIANCE GRAPHS WHEN PARAMETER p VARYING:')
plt.show()
```

FIRST PART
DISTRIBUTION WHEN p IS VARYING:



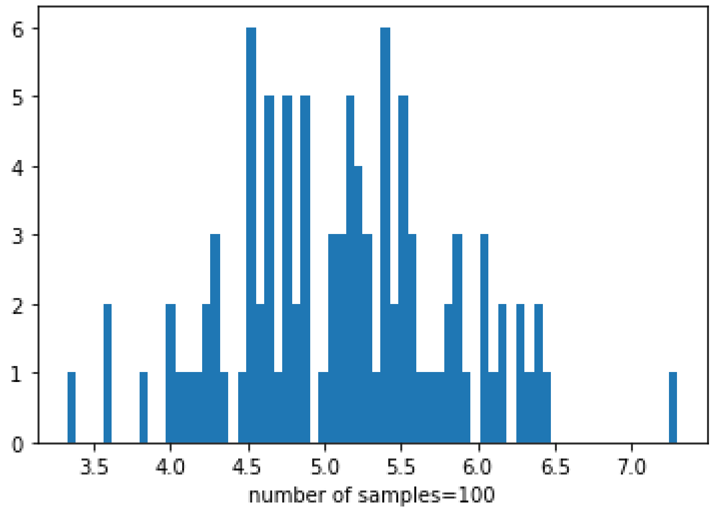
MEAN AND VARIANCE GRAPHS WHEN PARAMETER p VARYING:

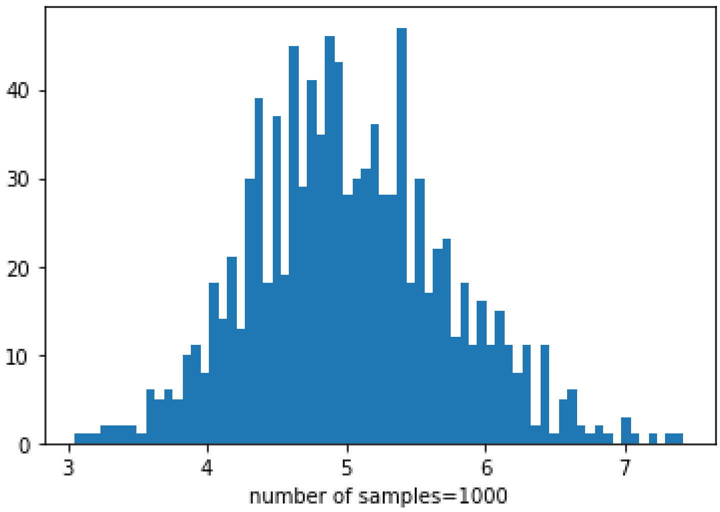


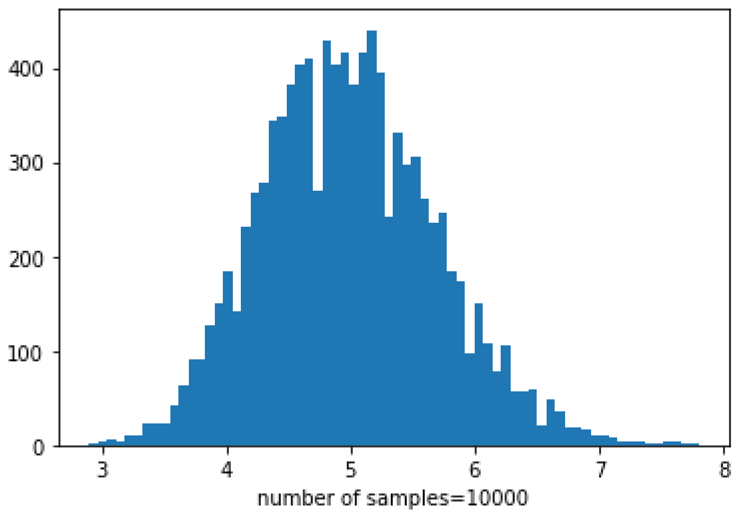
In [9]:

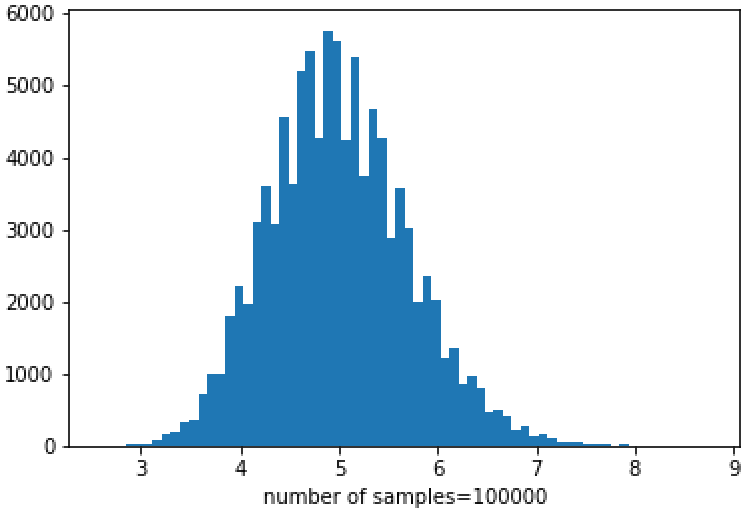
```
print("SECOND PART")
import numpy as np
import matplotlib.pyplot as plt
#calculate sample mean with different number of samples
p=0.6
n=40
print("when p=0.6 and n=40")
ns=10
for i in range(4):
    ns=ns*10
    samplemean=[]
    for j in range(ns):
        sum=0
        x = np.random.geometric(0.2,n)
        for k in x:
            sum=sum+k
        samplemean.append(sum/n)
    plt.hist(samplemean,68)
    plt.xlabel('number of samples={}'.format(ns))
    plt.show()
```

```
SECOND PART
when p=0.6 and n=40
```







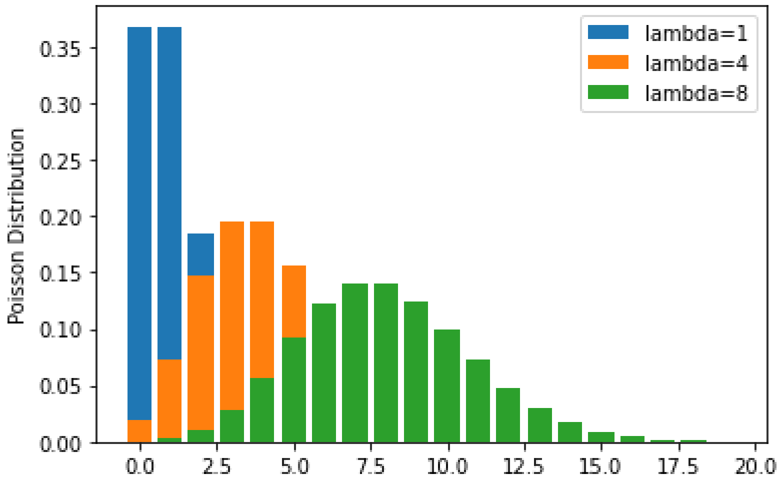


POISSON DISTRIBUTION

In [10]:

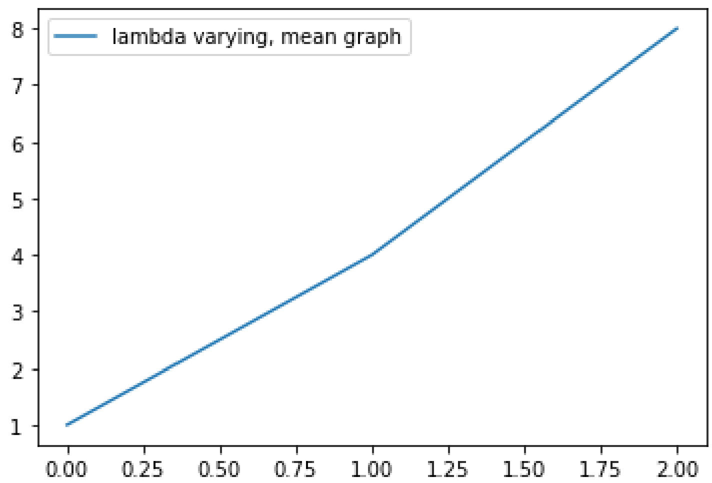
```
print("FIRST PART")
#factorial function
def fact(n):
    return 1 if (n==1 or n==0) else n * fact(n - 1);

import numpy as np
import matplotlib.pyplot as plt

x_var=list(range(20))
lam_var=[1,4,8]
xPoints=[]
for i in range(len(lam_var)):
    lam=lam_var[i]
    for j in x_var:
        #pdf of poisson distribution
        prob=(np.exp(-lam))*((np.power(lam,j))/fact(j))
        xPoints.append(prob)

    plt.bar(x_var, xPoints,label='lambda={}'.format(lam))
    xPoints=[]

plt.ylabel('Poisson Distribution')
plt.legend()
print('Poisson distribution when lambda is varying:')
plt.show()

mean=[]
var=[]
print('In poisson distribution mean=var=lambda')
for i in range(len(lam_var)):
    lam=lam_var[i]
    mean.append(lam)
    var.append(lam)
plt.plot(mean,label='lambda varying, mean graph')
plt.legend()
print('\nMean and variance graph when lambda is varying:')
plt.show()
```

FIRST PART
Poisson distribution when lambda is varying:



In poisson distribution mean=var=lambda

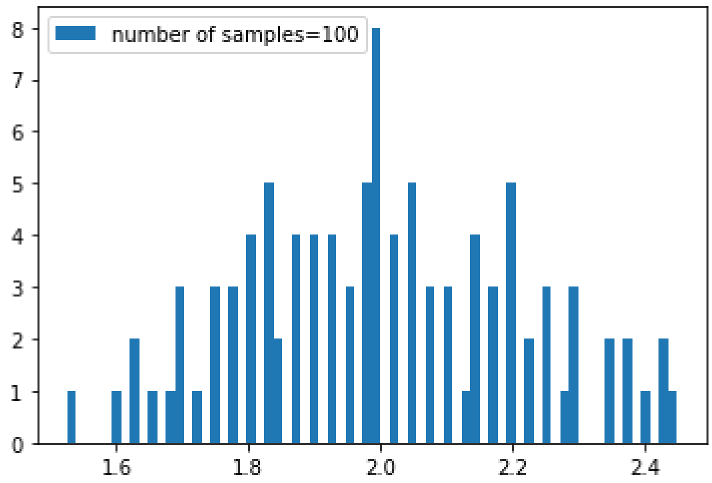
Mean and variance graph when lambda is varying:

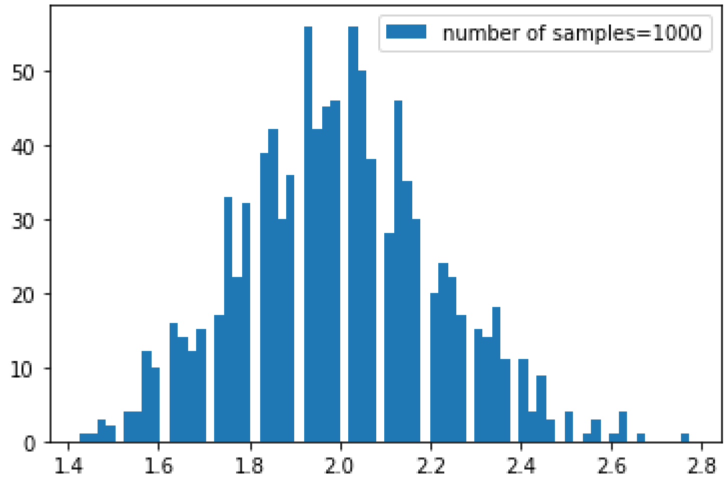


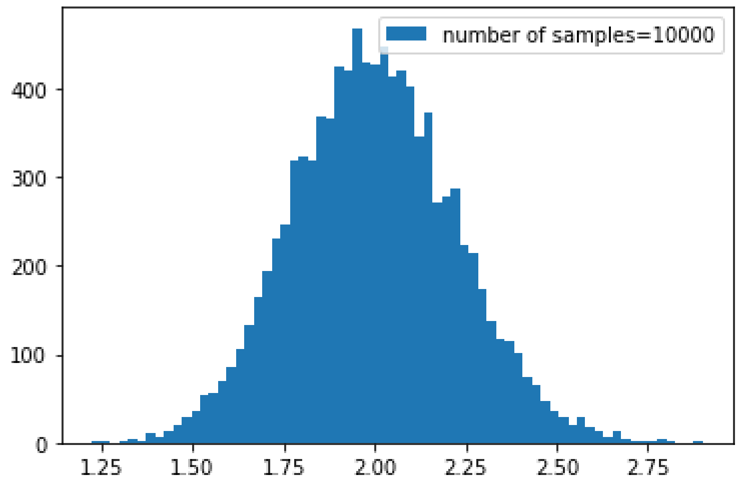
In [11]:

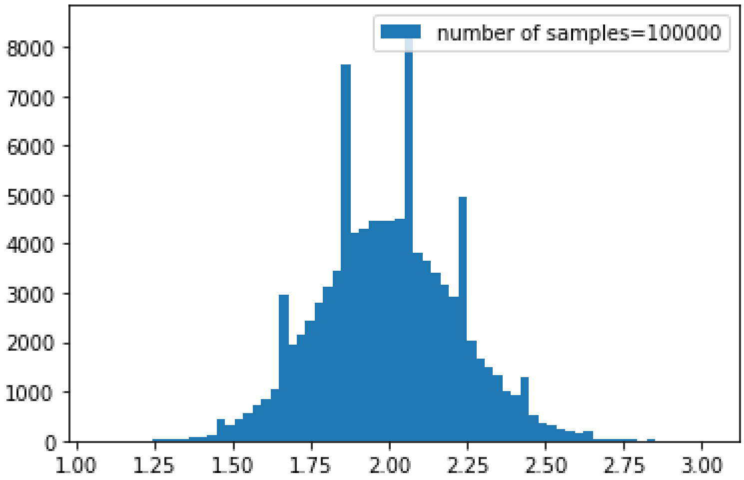
```
print("SECOND PART")
import numpy as np
import matplotlib.pyplot as plt
#calculate sample mean with different number of samples
lam=2
n=40
print("when lam=2 and n=40")
ns=10
for i in range(4):
    ns=ns*10
    samplemean=[]
    for j in range(ns):
        sum=0
        x = np.random.poisson(2,n)
        for k in x:
            sum=sum+k
        samplemean.append(sum/n)
    plt.hist(samplemean,68,label='number of samples={}'.format(ns))
    plt.legend()
    plt.show()
```

```
SECOND PART
when lam=2 and n=40
```









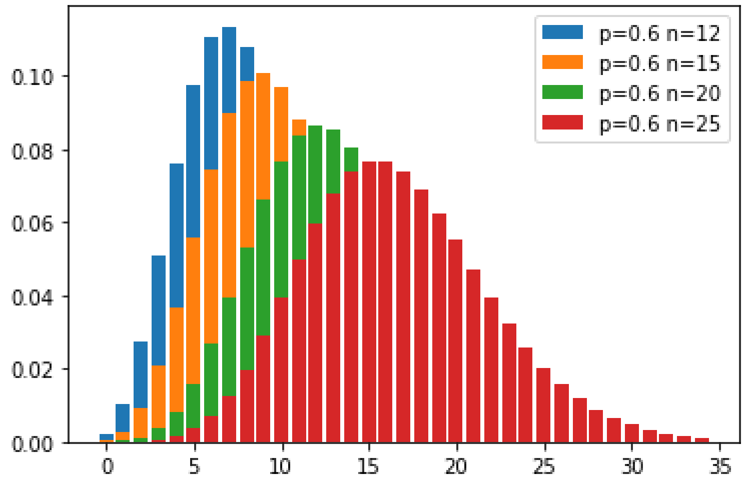
NEGATIVE BINOMIAL

In [13]:

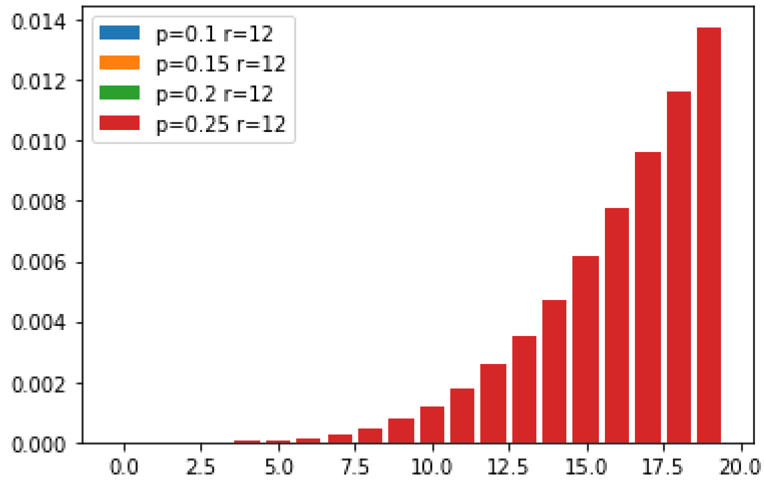
```
print("FIRST PART")
def fact(n):
    return 1 if (n==1 or n==0) else n * fact(n - 1);

import numpy as np
import matplotlib.pyplot as plt
from matplotlib import pyplot
r_var=[12,15,20,25]#number of success
k_var=list(range(35))#num of failures
p_var=[0.1,0.15,0.2,0.25]
p=0.6
xPoints=[]
yPoints=[]
#Here r is varying
for i in range(len(r_var)):
    r=r_var[i]
    for k in k_var:
        prob=(fact(k+r-1)/(fact(k)*fact(r-1)))*(p**r)*((1-p)**(k))
        yPoints.append(prob)
    plt.bar(k_var, yPoints,label='p=0.6 n={}'.format(r))
    yPoints=[]
plt.legend()
print('DISTRIBUTION WHEN r IS VARYING:')
plt.show()

#here r fixed and p is varying
r=12
for i in range(len(p_var)):
    p=p_var[i]
    for k in x_var:
        prob=(fact(k+r-1)/(fact(k)*fact(r-1)))*(p**r)*((1-p)**(k))
        xPoints.append(prob)
    plt.bar(x_var, xPoints,label='p={} r=12'.format(p))
    xPoints=[]
plt.legend()
print('DISTRIBUTION WHEN P IS VARYING:')
plt.show()

#when p fixed and r varying plot for mean and variance vary
print("when p fixed and r varying mean and variance vary as")
p=0.6
mean=[]
var=[]
for i in range(len(r_var)):
    r=r_var[i]
    mean.append((r*p)/(1-p))
    var.append((r*p)/(1-p)**2)
plt.plot(r_var,mean,label='p fixed r varying, mean graph')
plt.plot(r_var,var,label='p fixed r varying, variance graph')
plt.legend()
plt.show()

print("when r fixed and p varying mean and variance vary as")
r=12
mean=[]
var=[]
for i in range(len(p_var)):
    p=p_var[i]
    mean.append((r*p)/(1-p))
```

```
    var.append((r*p)/(1-p)**2)
plt.plot(p_var,mean,label='r fixed p varying, mean graph')
plt.plot(p_var,var,label='r fixed p varying, variance graph')
plt.legend()
plt.show()
```

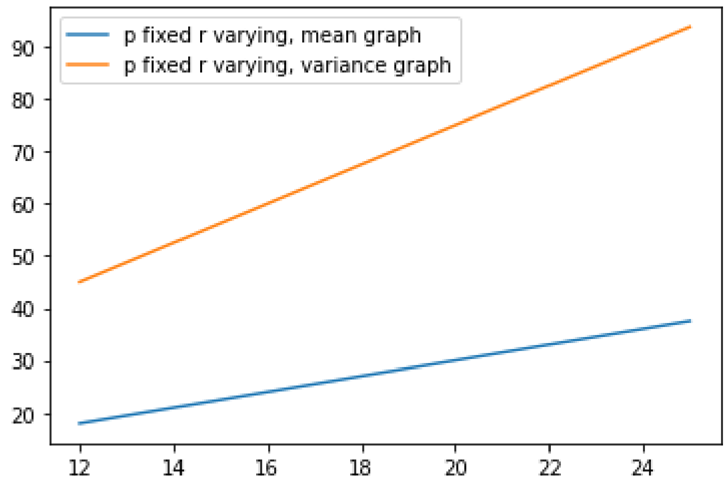
```
FIRST PART
DISTRIBUTION WHEN r IS VARYING:
```



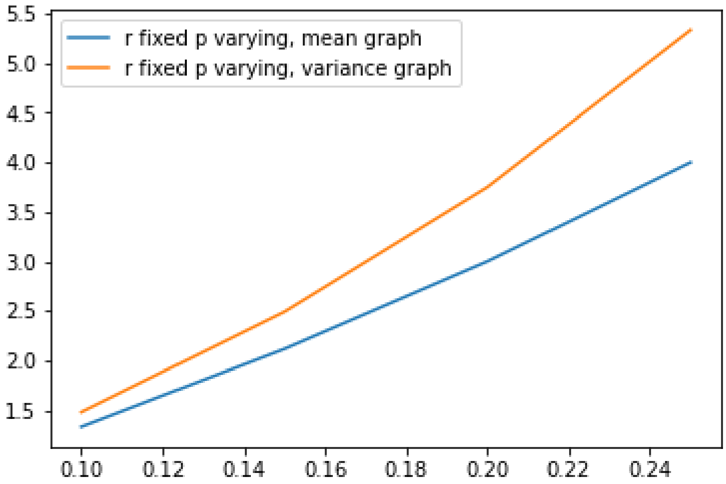

```
DISTRIBUTION WHEN P IS VARYING:
```




```
when p fixed and r varying mean and variance vary as
```



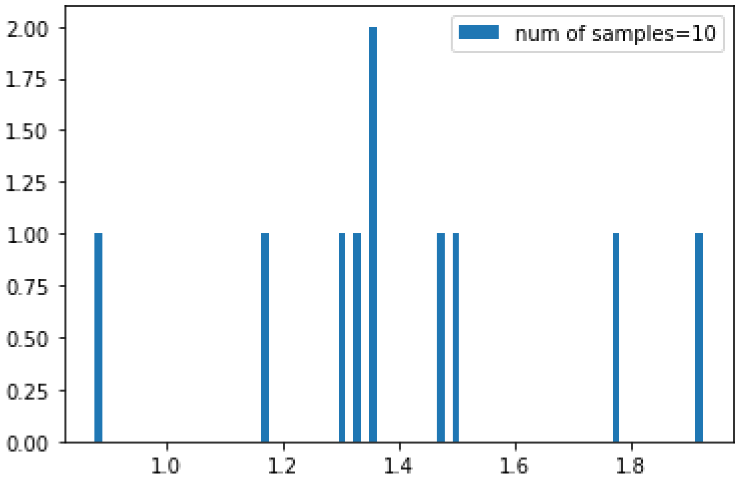

```
when r fixed and p varying mean and variance vary as
```

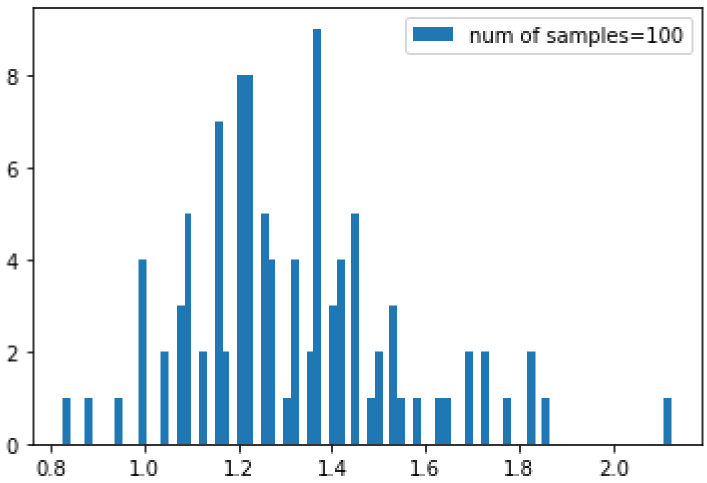

r fixed p varying, mean graph
r fixed p varying, variance graph
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
0.10
0.12
0.14
0.16
0.18
0.20
0.22
0.24

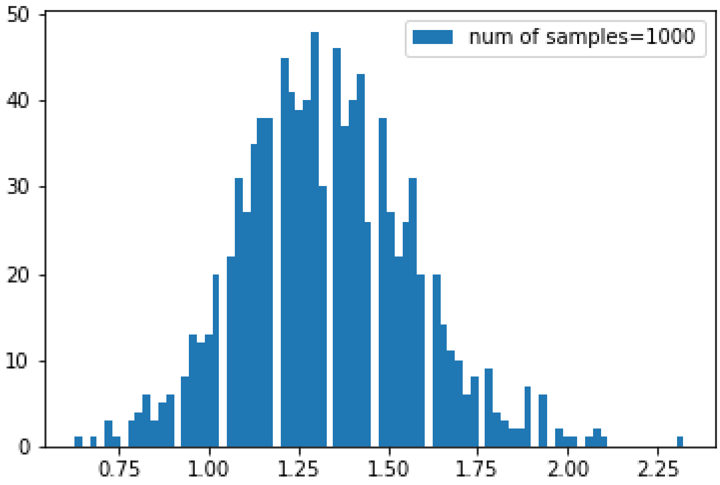
In [14]:

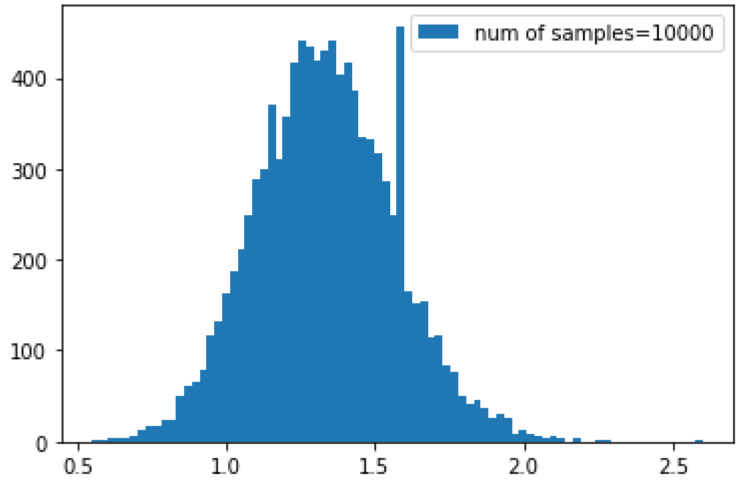
```
print("SECOND PART")
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import pyplot
nt=40#number of trails
r_var=list(range(n + 1))#num of success
p=0.6
ns=[10,100,1000,10000]
xPoints=[]
sampleMean=[]
yPoints=[]
#Here n is varying
for i in range(4):
    for k in range(ns[i]):
        temp=np.random.negative_binomial(2,p,40)
        sum1=0
        for m in temp:
            sum1=sum1+m
        sampleMean.append(sum1/40)
    plt.hist(sampleMean,80,label="num of samples={}".format(k+1))
    sampleMean=[]
    plt.legend()
    plt.show()
```

SECOND PART


num of samples=10
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
1.0
1.2
1.4
1.6
1.8


num of samples=100
8
6
4
2
0
0.8
1.0
1.2
1.4
1.6
1.8
2.0


num of samples=1000
50
40
30
20
10
0
0.75
1.00
1.25
1.50
1.75
2.00
2.25



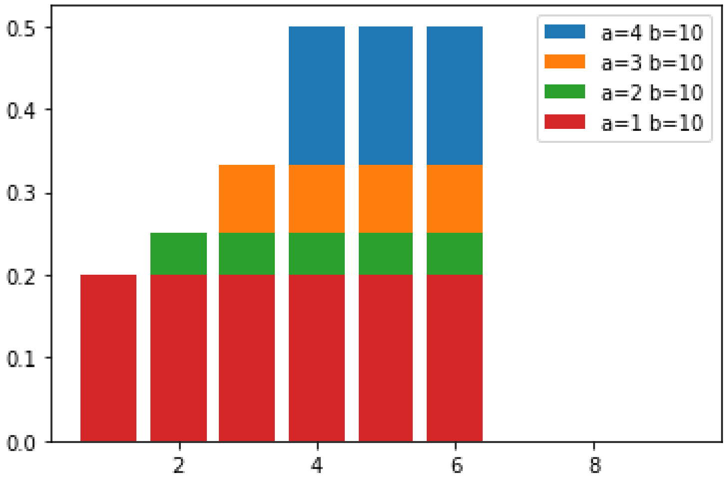
DISCRETE UNIFORM

In [15]:

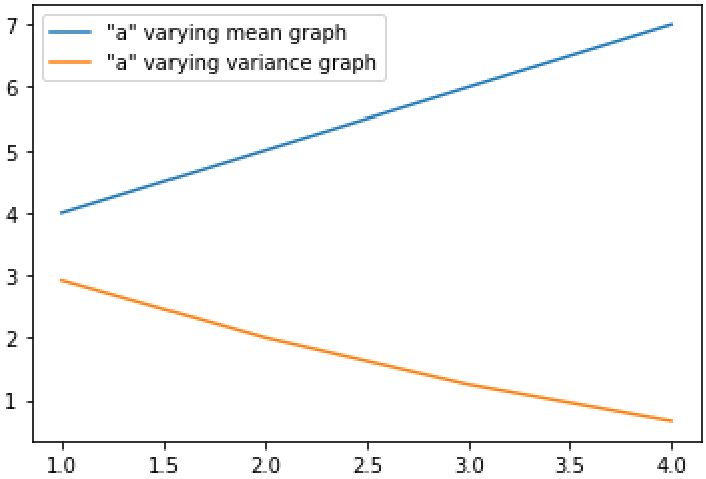
```
print("FIRST PART")
#Discrete Distribution
import numpy as np
import matplotlib.pyplot as plt
x_var= []
for i in range(1,10):
    x_var.append(i)
a_var=[4,3,2,1]
n=10
k=1
x=[]
y=[]
means=[]
var=[]
#if a varies
b=6
for j in a_var:
    for i in range(len(x_var)):
        if x_var[i]<=b and x_var[i]>=j:
            prob=1/(b-j)
            y.append(prob)
        else:
            y.append(0)
    var.append((((b-j+1)**2) - 1)/12)
    means.append(j+b/2)

    #plt.scatter(x_var,y)
    plt.bar(x_var,y,label='a={} b=10'.format(j))
    y=[]
plt.legend(loc='best')
print("When parameter 'a' varies distribution plot is:")
plt.show()
print("When parameter 'a' varies mean and variance vary as:")
plt.plot(a_var,means,label='"a" varying mean graph')
plt.plot(a_var,var,label='"a" varying variance graph')
plt.legend()
plt.show()

#if b varies
a=2
means=[]
var=[]
b_var=[6,7,8,9]

for j in b_var:
    for i in range(len(x_var)):
        if x_var[i]<=j and x_var[i]>=a:
            prob=1/(j-a)
            y.append(prob)
        else:
            y.append(0)
        #y.append(prob)
    var.append((((j-a+1)**2) - 1)/12)
    means.append(a+j/2)

    #plt.scatter(x_var,y)
    plt.bar(x_var,y,label='a=2 b={}'.format(j))
    y=[]
plt.legend(loc='best')
```

```
print("When parameter 'b' varies distribution plot is:")
plt.show()
print("When parameter 'b' varies mean and variance vary as:")
plt.plot(b_var,means,label='"b" varying mean graph')
plt.plot(b_var,var,label='"b" varying variance graph')
plt.legend()
plt.show()
```

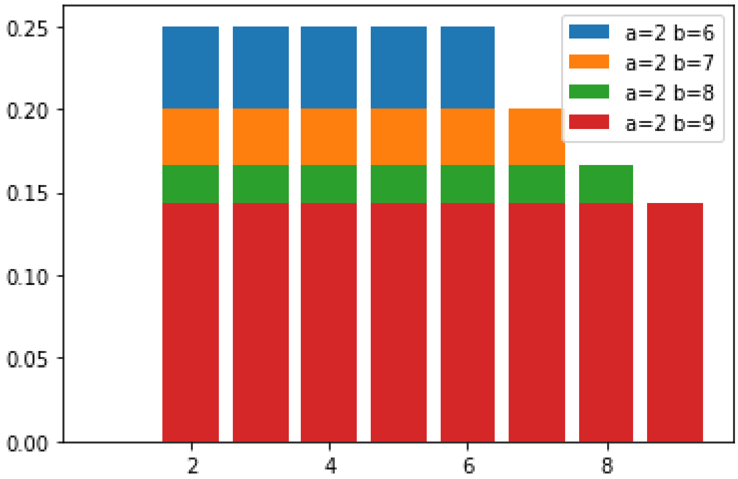
```
FIRST PART
When parameter 'a' varies distribution plot is:
```



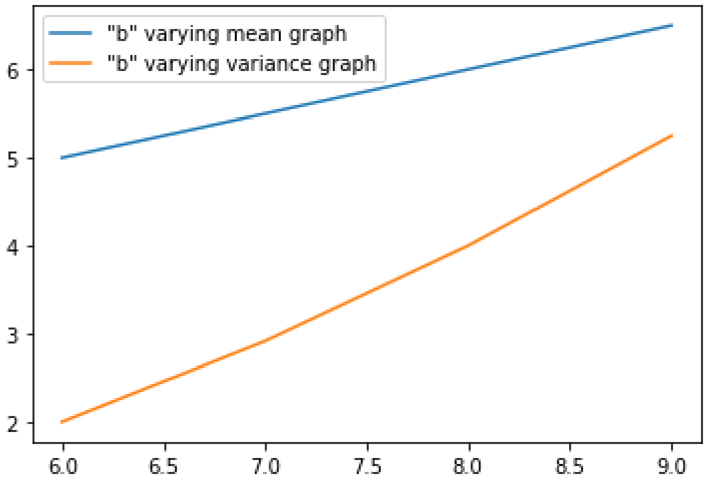

```
When parameter 'a' varies mean and variance vary as:
```




```
When parameter 'b' varies distribution plot is:
```



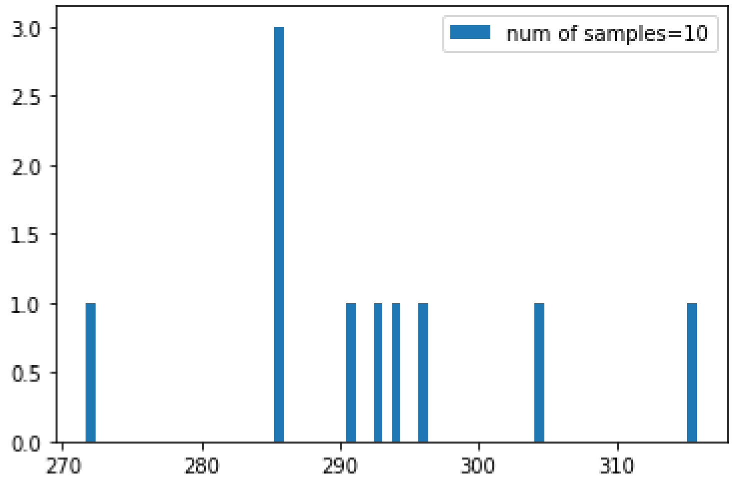
When parameter 'b' varies mean and variance vary as:

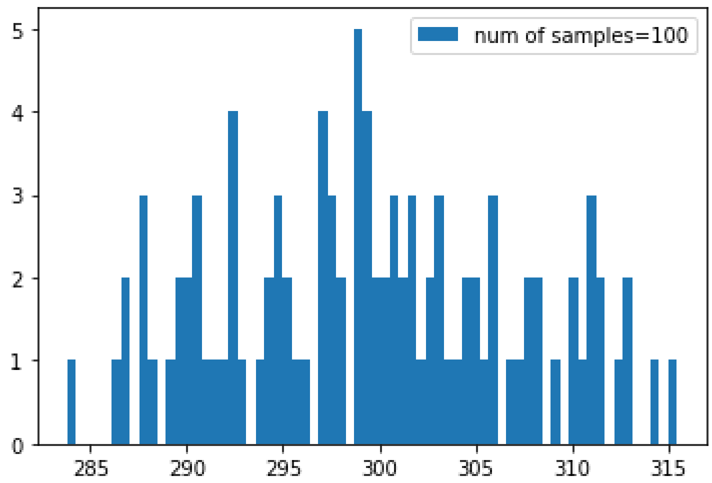


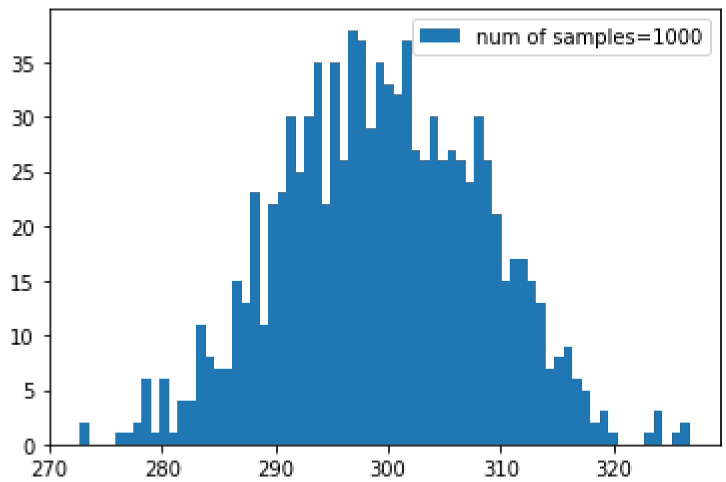
In [16]:

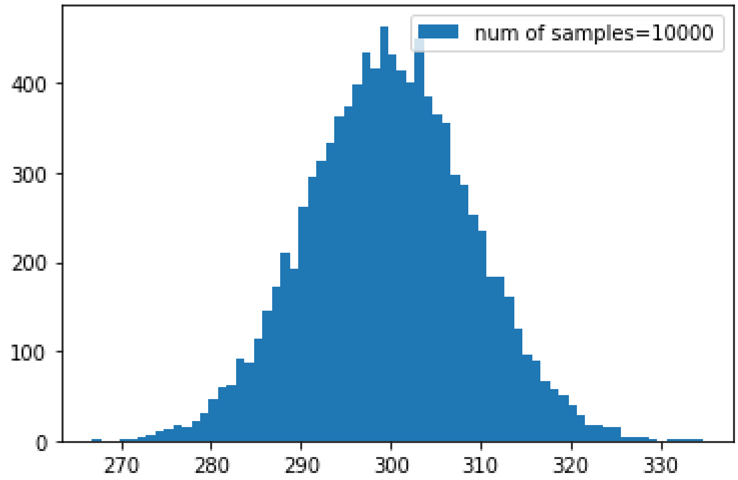
```
print("SECOND PART:To prove CLT")
#to prove CLT
from numpy import random
import matplotlib.pyplot as plt
n=[10,100,1000,10000]
sampleMean=[]
for i in n:
    for j in range(i):
        sum1=0
        x = random.uniform(5,10,40)
        for k in x:
            sum1=sum1+k
        sampleMean.append(sum1)
    plt.hist(sampleMean,68,label="num of samples={}".format(i))
    plt.legend()
    plt.show()
    sampleMean=[]
```

SECOND PART:To prove CLT









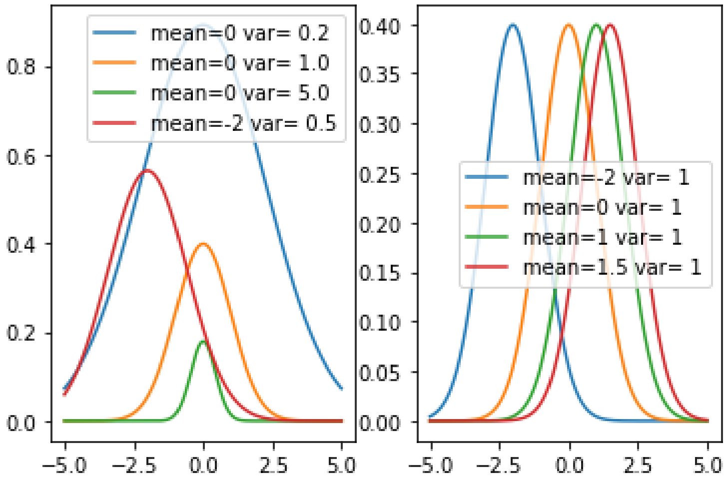
CONTINUOUS DISTRIBUTIONS

NORMAL DISTRIBUTION

In [17]:

```
print("FIRST PART")
import numpy as np
import matplotlib.pyplot as plt
print("NORMAL DISTRIBUTION")
mean = 0;

x = np.linspace(-5,5,1000)
Var=[0.2,1.0,5.0,0.5]
Mu=[-2,0,1,1.5]
#When parameter mean is fixed and variance is varying(Since mean and variance are param
eters in normal distribution)
fig,(ax1,ax2)=plt.subplots(1,2)
print("Normal distribution graphs:")
print("First graph shows when parameter variance is varying and second graph shows when
parameter mean is varying ")
for i in range(4):
    #let it be initial variance

    mean=0
    if(i==3):
        mean=-2
    var=Var[i]
    f = (1/np.sqrt(2*np.pi*var))* np.exp((-1/2*var)*np.square(x-mean))
    #f = np.exp(-np.square(x-mean)/2*var)/(np.sqrt(2*np.pi*var))
    mean=0
    if(i==3):
        mean=-2
    ax1.plot(x,f,label='mean={} var= {}'.format(mean,var))
    ax1.legend()

    mean=Mu[i]
    var=1
    f = (1/np.sqrt(2*np.pi*var))* np.exp((-1/2*var)*np.square(x-mean))
    #f = np.exp(-np.square(x-mean)/2*var)/(np.sqrt(2*np.pi*var))

    ax2.plot(x,f,label='mean={} var= 1'.format(mean))
    ax2.legend()
```

```
FIRST PART
NORMAL DISTRIBUTION
Normal distribution graphs:
First graph shows when parameter variance is varying and second graph show
s when parameter mean is varying
```



In [18]:

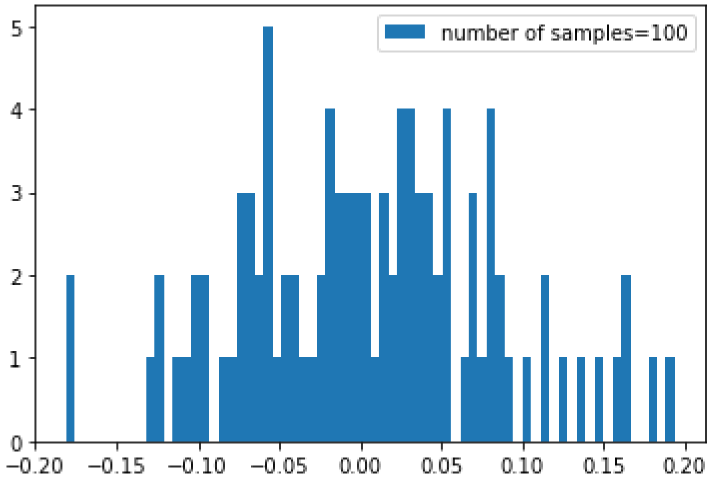
```
print("Since in normal distribution parameters are mean and variance itself, we cannot
 show their varying as they are independent")
```

```
Since in normal distribution parameters are mean and variance itself, we c
annot show their varying as they are independent
```

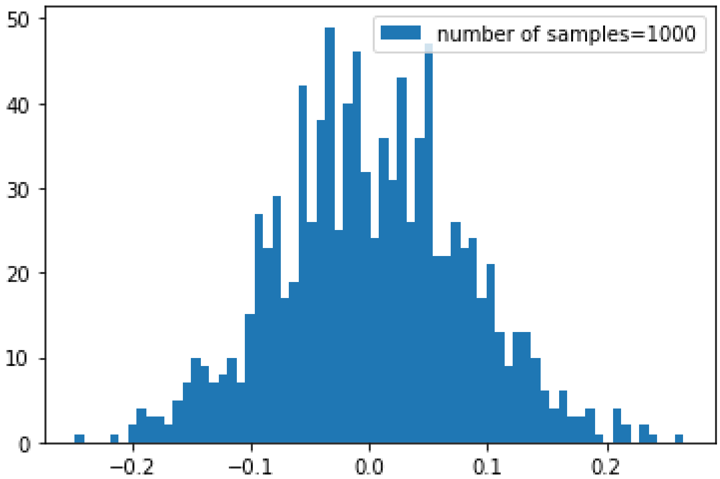
In [19]:

```
print("SECOND PART")
import numpy as np
from numpy.random import seed
from numpy.random import randint
from numpy import mean

import matplotlib.pyplot as plt
seed(1)

l=[100,1000,10000,100000]
#print(x)
#calculate mean of 40 weights 1000 times
for j in l:
    print("WHEN n=40")
    print("PLOT FOR SAMPLE MEANS IS")
    samplemean=[]
    for i in range(j):
        sum1=0
        x = np.random.normal(0,0.5,40)
        for k in x:
            sum1=sum1+k
        samplemean.append(sum1/40)
    plt.hist(samplemean,68,label="number of samples={}".format(j))
    plt.legend()
    plt.show()
```

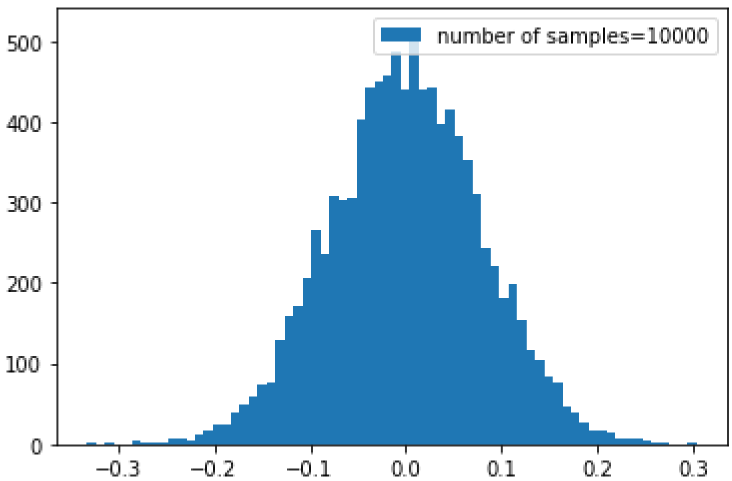
```
SECOND PART
WHEN n=40
PLOT FOR SAMPLE MEANS IS
```



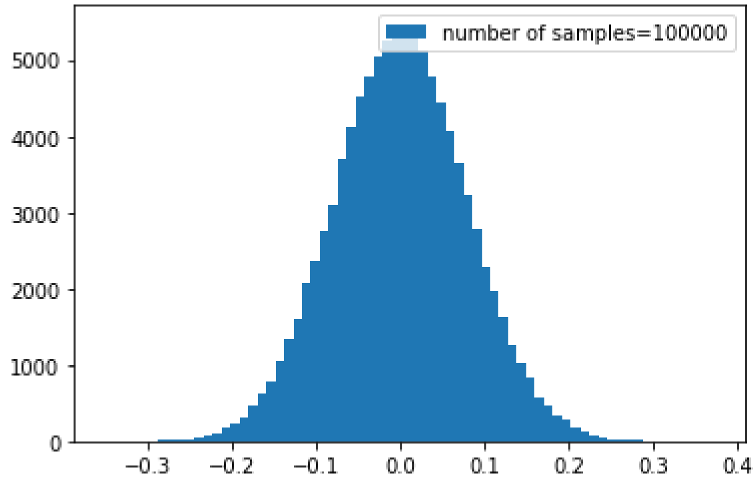

```
WHEN n=40
PLOT FOR SAMPLE MEANS IS
```




```
WHEN n=40
PLOT FOR SAMPLE MEANS IS
```




```
WHEN n=40
PLOT FOR SAMPLE MEANS IS
```

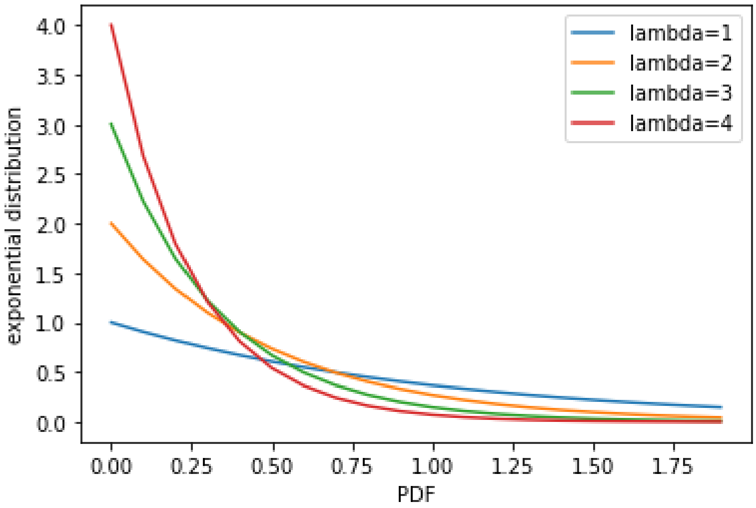


EXPONENTIAL DISTRIBUTION

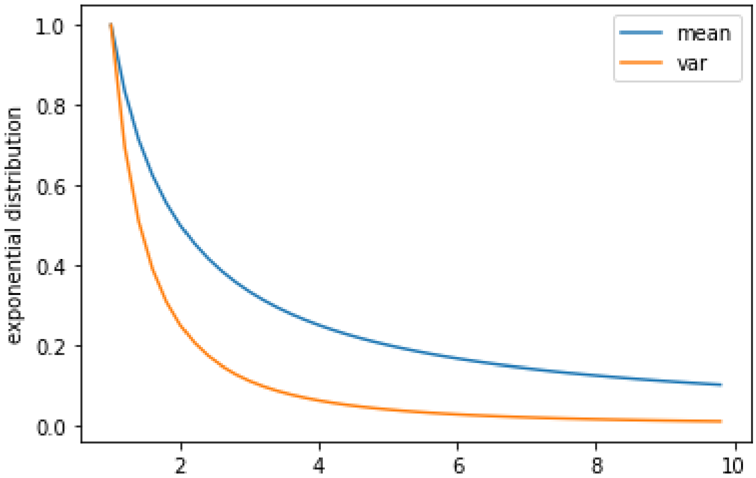
In [21]:

```
print("FIRST PART")
import numpy as np
import matplotlib.pyplot as plt
lam = 1
mean = 1/lam
std = 1/lam
var = np.square(std)
x = np.arange(0,2,.1)
print("PDF of exponential distribution when lambda is varying")
for i in range(4):
    f = lam*(np.exp(-(lam*x)))
    plt.plot(x,f,label='lambda={}'.format(lam))
    lam = lam+1
plt.ylabel('exponential distribution')
plt.xlabel('PDF')
plt.legend()
plt.show()
print("Mean and Variance of exponential distribution when lambda is varying")
x = np.arange(1,10,0.2)
mean = 1/x
std = 1/x
var = np.square(std)
plt.plot(x,mean,label='mean')
plt.plot(x,var,label='var')
plt.ylabel('exponential distribution')
plt.legend()
plt.show()
```

FIRST PART
PDF of exponential distribution when lambda is varying



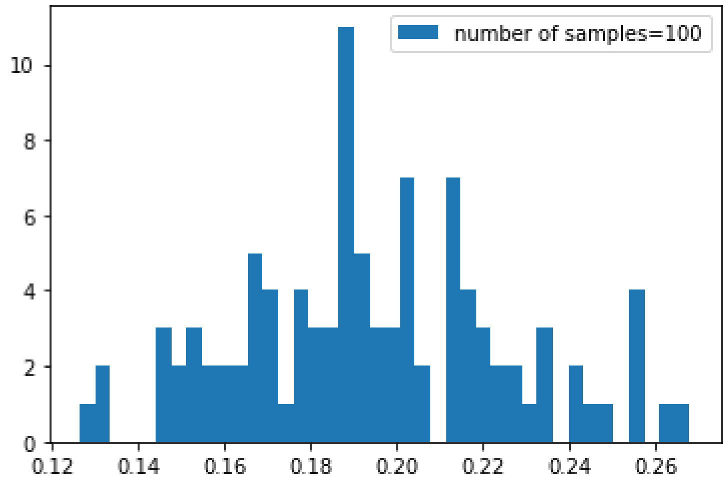
Mean and Variance of exponential distribution when lambda is varying

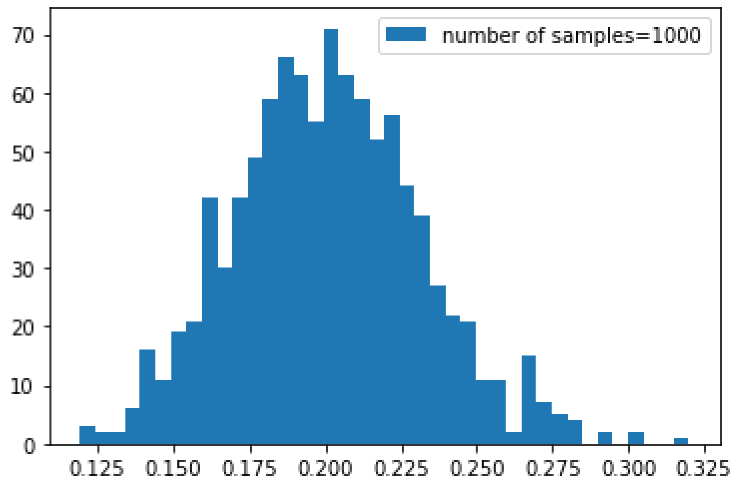


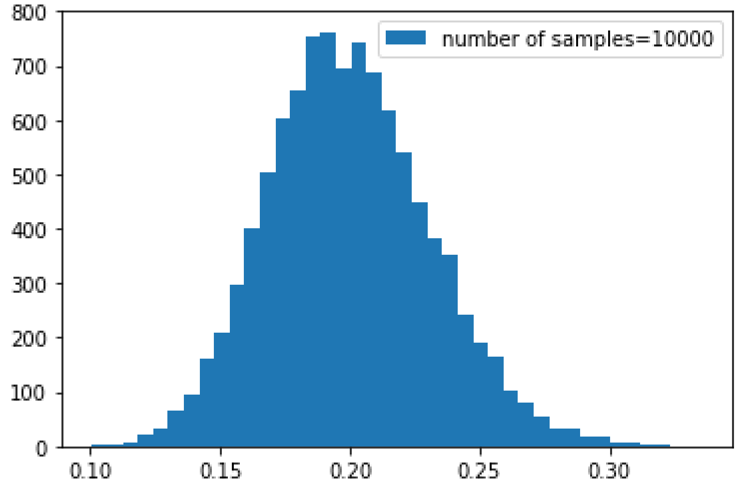
In [22]:

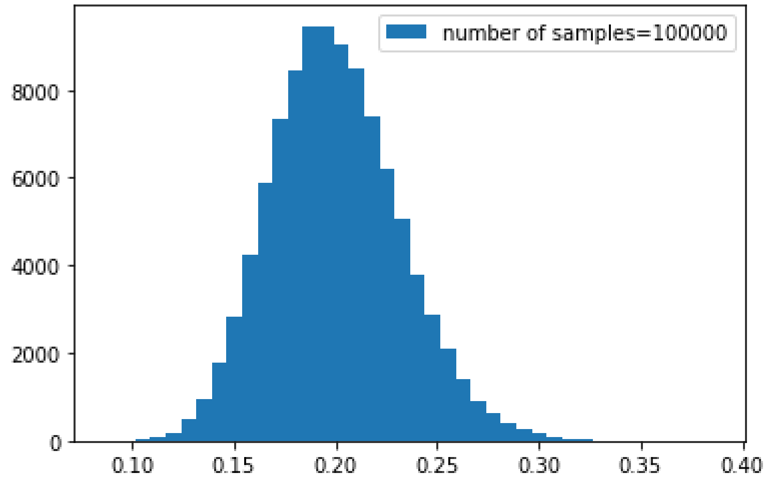
```
print("SECOND PART")
import numpy as np
import matplotlib.pyplot as plt
#calculate sample mean with different number of samples
lam=0.2
n=40
print("when lambda=0.2 and n=40")
ns=10
for i in range(4):
    ns=ns*10
    samplemean=[]
    for j in range(ns):
        sum1=0
        x = np.random.exponential(0.2,n)
        for k in x:
            sum1=sum1+k
        samplemean.append(sum1/n)
    plt.hist(samplemean,40,label='number of samples={}'.format(ns))
    plt.legend()
    plt.show()
```

```
SECOND PART
when lambda=0.2 and n=40
```







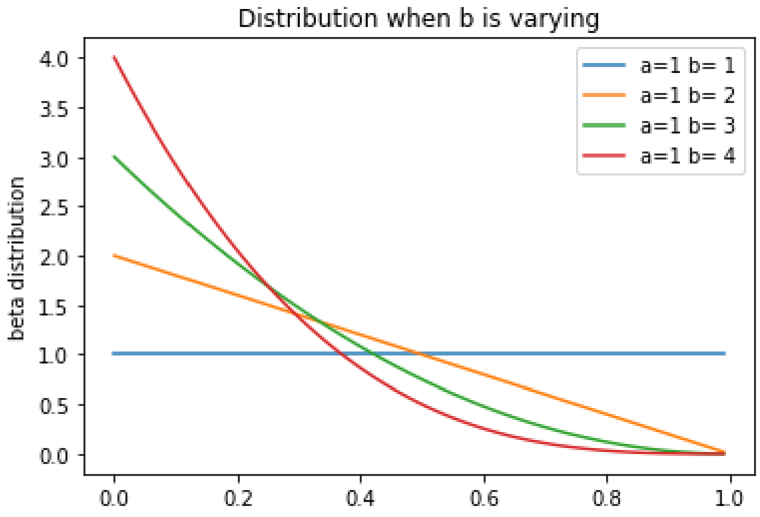


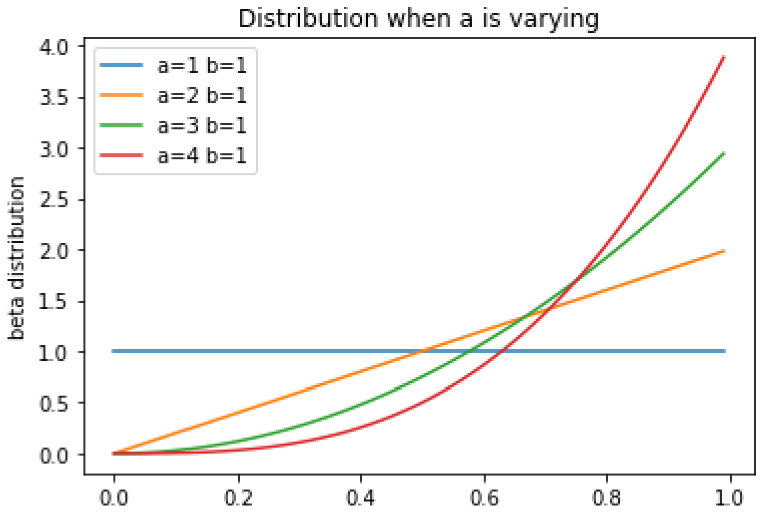
BETA DISTRIBUTION

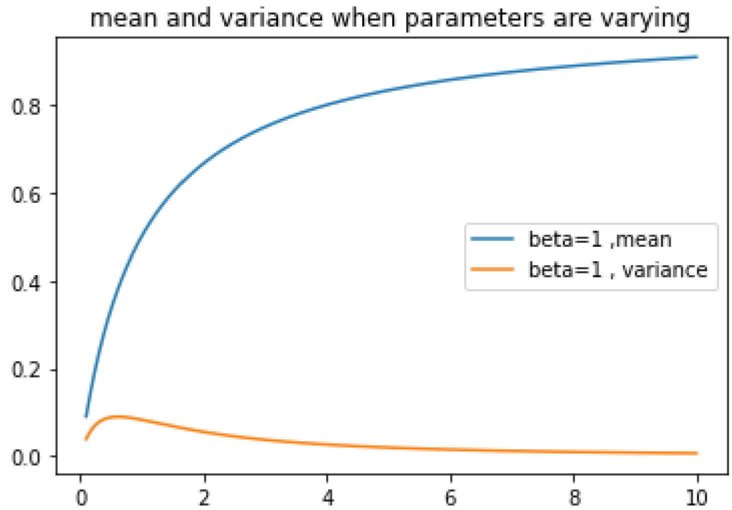
In [23]:

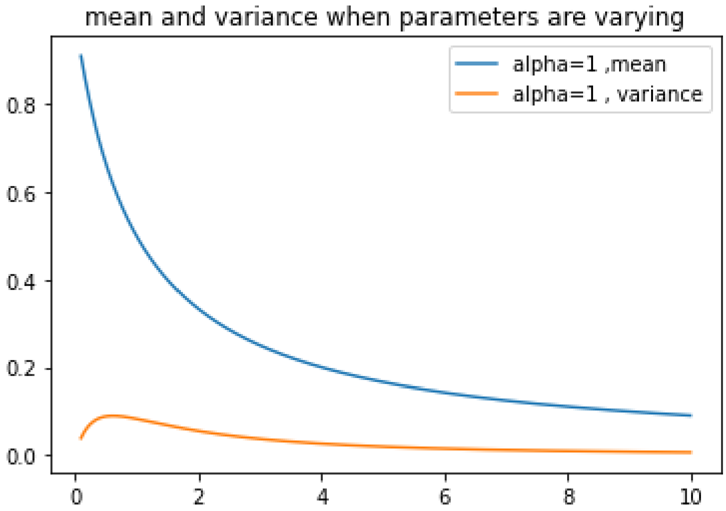
```
print("FIRST PART")
import numpy as np
import matplotlib.pyplot as plt
def fact(n):
    if n==0:
        return 1
    return n*fact(n-1)

x=np.arange(0,1, 0.01)
a=1
b=1
for i in range(4):

    prob = fact(a+b-1)/(fact(a-1)*fact(b-1)) * ( x**(a-1) * (1-x)**(b-1) )
    plt.plot(x,prob,label='a=1 b= {}'.format(b))
    b=b+1
plt.title('Distribution when b is varying')
plt.ylabel('beta distribution')
plt.legend()
plt.show()
b=1
a=1
for i in range(4):

    prob = fact(a+b-1)/(fact(a-1)*fact(b-1)) * ( x**(a-1) * (1-x)**(b-1) )
    plt.plot(x,prob,label='a={} b=1'.format(a))
    a=a+1
plt.title('Distribution when a is varying')
plt.ylabel('beta distribution')
plt.legend()
plt.show()
a=np.arange(0.1,10, 0.01)
b=1
mean=a/(a+b);
var=(a*b)/((a+b)*(a+b)*(a+b+1));
plt.plot(a,mean,label='beta=1 ,mean');
plt.plot(a,var,label='beta=1 , variance');
plt.title('mean and variance when parameters are varying')
plt.legend()
plt.show()

b=np.arange(0.1,10, 0.01)
a=1
mean=a/(a+b);
var=(a*b)/((a+b)*(a+b)*(a+b+1));
plt.plot(b,mean,label='alpha=1 ,mean');
plt.plot(b,var,label='alpha=1 , variance');
plt.title('mean and variance when parameters are varying')
plt.legend()
plt.show()
```

FIRST PART


Distribution when b is varying
beta distribution
a=1 b= 1
a=1 b= 2
a=1 b= 3
a=1 b= 4
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Distribution when a is varying
beta distribution
a=1 b=1
a=2 b=1
a=3 b=1
a=4 b=1
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0

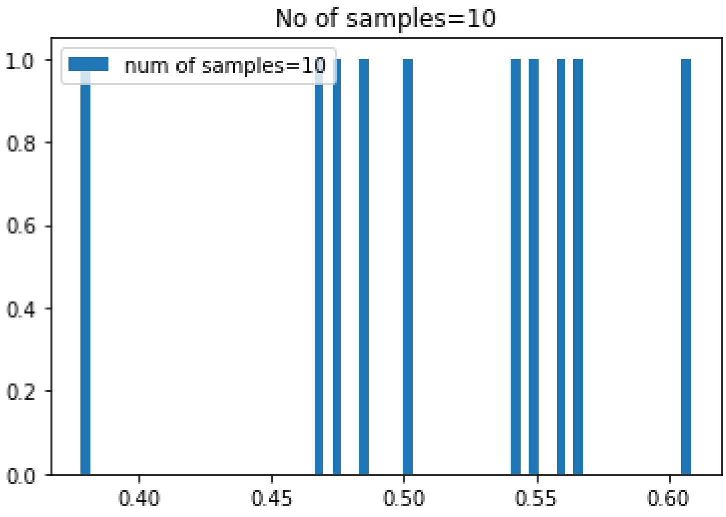

mean and variance when parameters are varying
beta=1 ,mean
beta=1 , variance
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
10

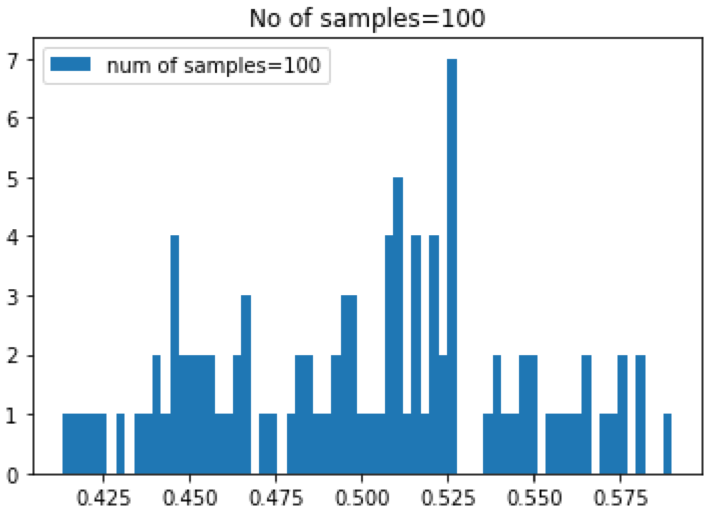

mean and variance when parameters are varying
alpha=1 ,mean
alpha=1 , variance
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8
10

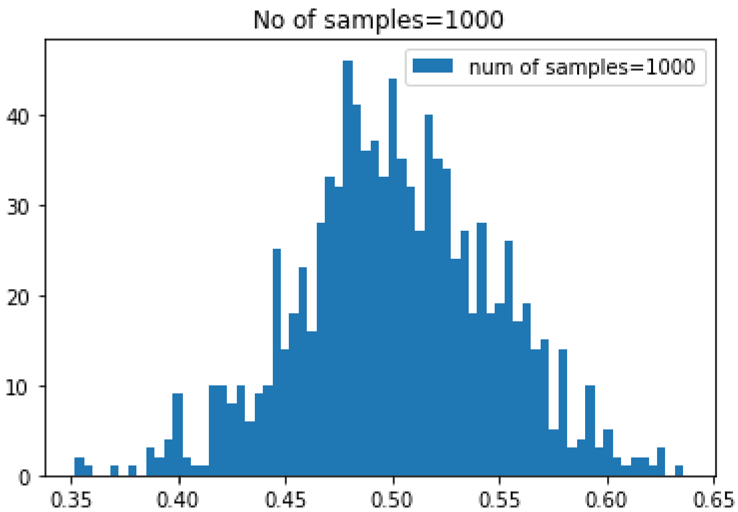
In [24]:

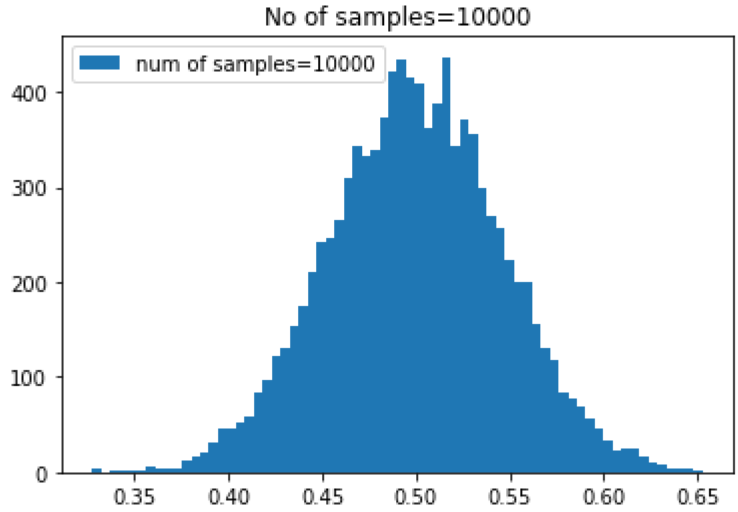
```
print("SECOND PART")
import numpy as np
import matplotlib.pyplot as plt
nt=40#number of trails

ns=[10,100,1000,10000]
xPoints=[]
sampleMean=[]
yPoints=[]
#Here n is varying
for i in range(4):
    for k in range(ns[i]):
        temp=np.random.beta(1,1,nt)
        sum1=0
        for m in temp:
            sum1=sum1+m
        sampleMean.append(sum1/40)
    plt.hist(sampleMean,68,label="num of samples={}".format(k+1))

    plt.title('No of samples={}'.format(ns[i]))

    sampleMean=[]
    plt.legend()
    plt.show()
```

SECOND PART


No of samples=10
num of samples=10
1.0
0.8
0.6
0.4
0.2
0.0
0.40
0.45
0.50
0.55
0.60


No of samples=100
num of samples=100
7
6
5
4
3
2
1
0
0.425
0.450
0.475
0.500
0.525
0.550
0.575


No of samples=1000
num of samples=1000
40
30
20
10
0
0.35
0.40
0.45
0.50
0.55
0.60
0.65

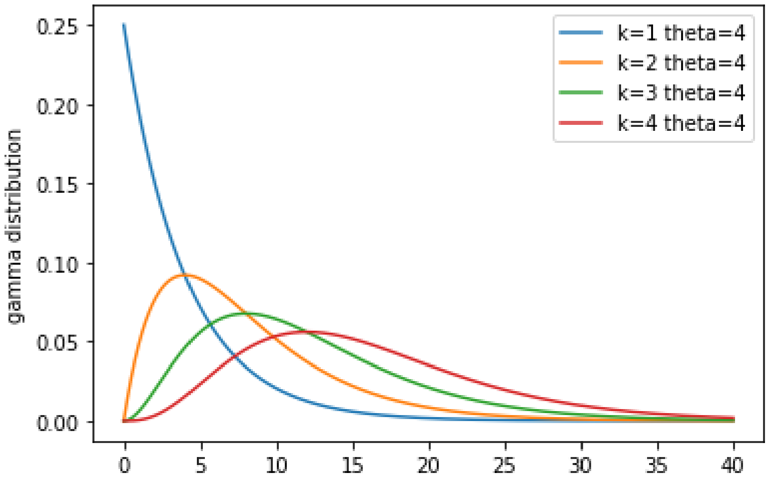


GAMMA DISTRIBUTION

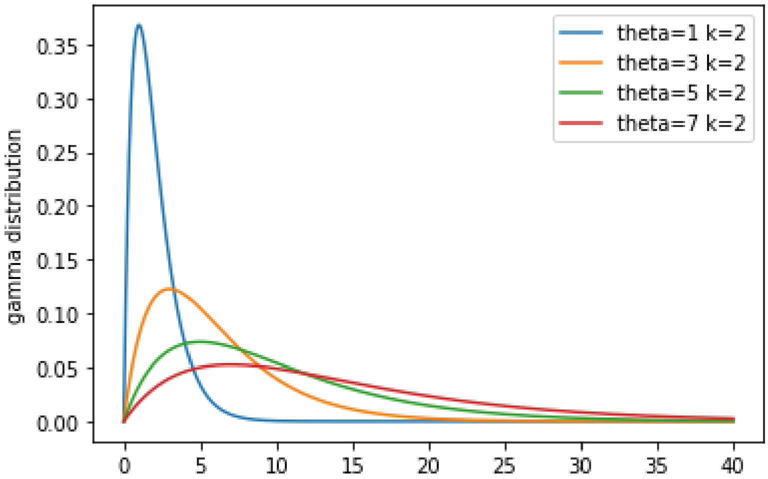
In [26]:

```
print("FIRST PART")
import numpy as np
import matplotlib.pyplot as plt

#factorial function
def fact(n):
    return 1 if (n==1 or n==0) else n * fact(n - 1);

print('Gamma distribution plot where theta is same and k is varying')
x=np.arange(0,40, 0.001)
k=1
theta=4

for i in range(4):
    y=((np.power(x,k-1))*np.exp(-x/theta))/(fact(k-1)*(np.power(theta,k)))
    plt.plot(x,y,label='k={} theta=4'.format(k))
    k=k+1
plt.legend()
plt.ylabel('gamma distribution')
plt.show()

print('Gamma distribution plot where k is same and theta is varying')
x=np.arange(0,40, 0.001)
k=2
theta=1
for i in range(4):
    y=((np.power(x,k-1))*np.exp(-x/theta))/(fact(k-1)*(np.power(theta,k)))
    plt.plot(x,y,label='theta={} k=2'.format(theta))
    theta=theta+2

plt.ylabel('gamma distribution')
plt.legend()
plt.show()

print('Mean and Variance graphs when theta is same and k is varying:')
theta=2
k=np.arange(0.1,10, 0.001)
mean=k*theta
var=k*theta*theta
plt.plot(k,mean,label='mean theta=2')
plt.plot(k,var,label='variance theta=2')
plt.legend()
plt.show()

print('Mean and Variance graphs when k is same and theta is varying:')
k=1
theta=np.arange(0.1,10, 0.001)
mean=k*theta
var=k*theta*theta
plt.plot(theta,mean,label='mean k=1')
plt.plot(theta,var,label='variance k=1')
plt.legend()
plt.show()
```

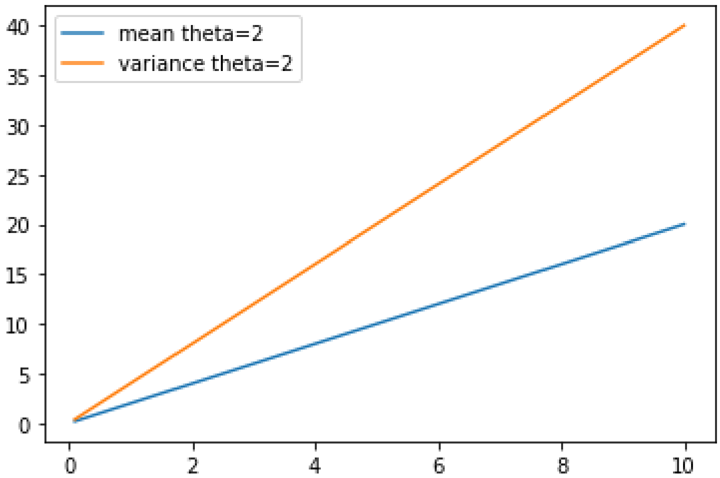
FIRST PART
Gamma distribution plot where theta is same and k is varying



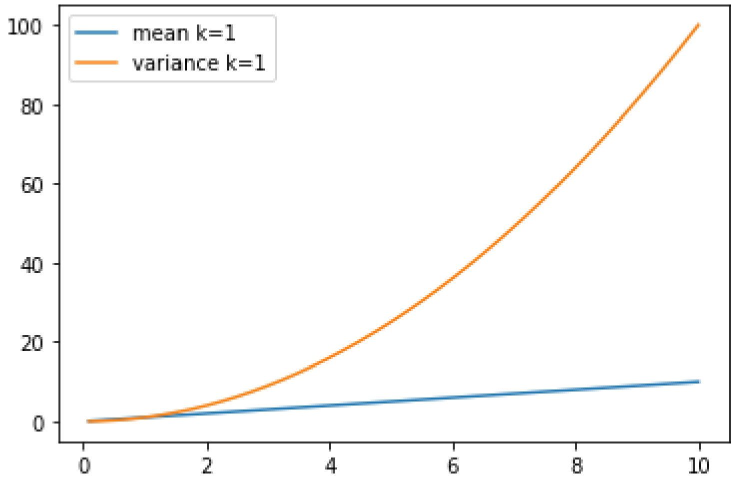
Gamma distribution plot where k is same and theta is varying



Mean and Variance graphs when theta is same and k is varying:



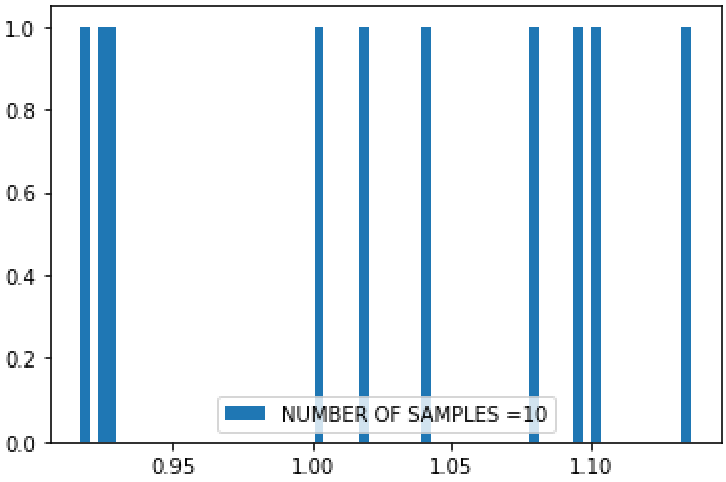
Mean and Variance graphs when k is same and theta is varying:

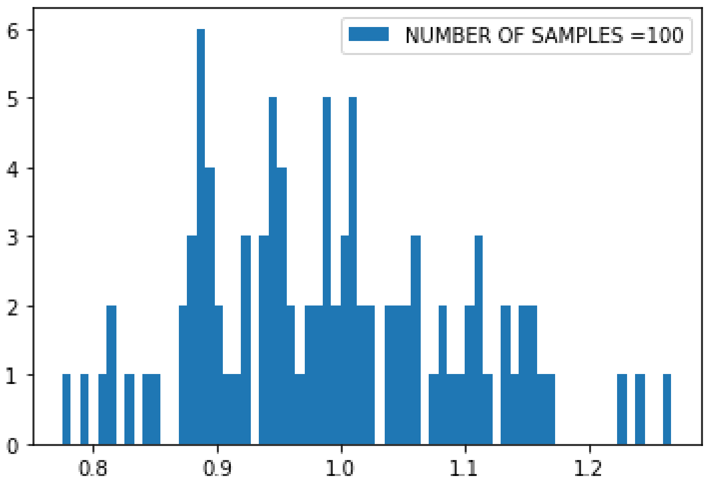


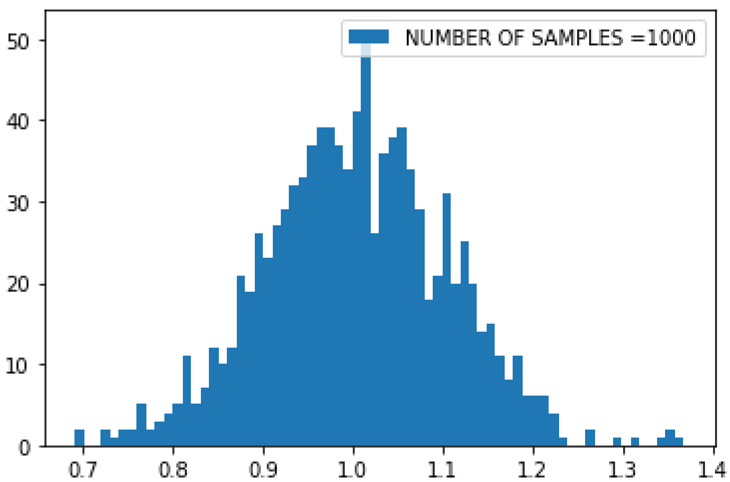
In [34]:

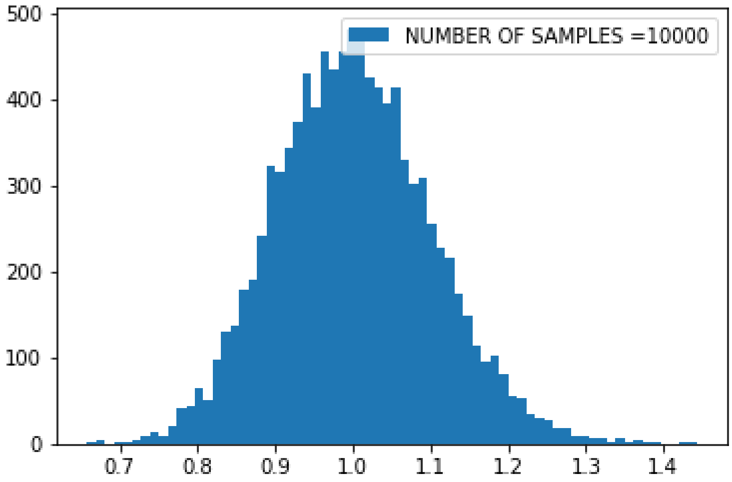
```
print("SECOND PART")
import numpy as np
import matplotlib.pyplot as plt
#plotting for different number of samples in gamma distribution
k=1
theta=1
n=100
ns=10
for i in range(4):

    samplemean=[]
    for j in range(ns):
        sum=0
        x = np.random.gamma(1,1,n)
        for k in x:
            sum=sum+k
        samplemean.append(sum/n)
    plt.hist(samplemean,68,label="NUMBER OF SAMPLES ={}".format(ns))
    plt.legend()
    plt.show()
    ns=ns*10
```

SECOND PART


NUMBER OF SAMPLES =10
0.0
0.2
0.4
0.6
0.8
1.0
0.95
1.00
1.05
1.10


NUMBER OF SAMPLES =100
0
1
2
3
4
5
6
0.8
0.9
1.0
1.1
1.2


NUMBER OF SAMPLES =1000
0
10
20
30
40
50
0.7
0.8
0.9
1.0
1.1
1.2
1.3
1.4

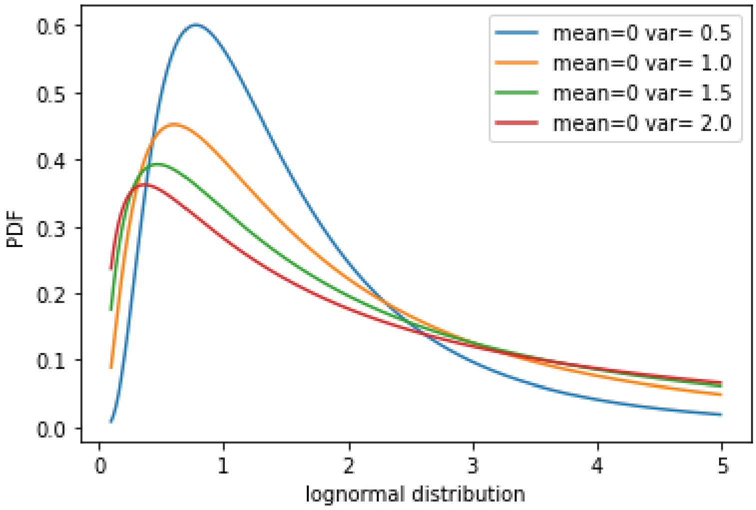

NUMBER OF SAMPLES =10000
500
400
300
200
100
0
0.7
0.8
0.9
1.0
1.1
1.2
1.3
1.4

LOG NORMAL

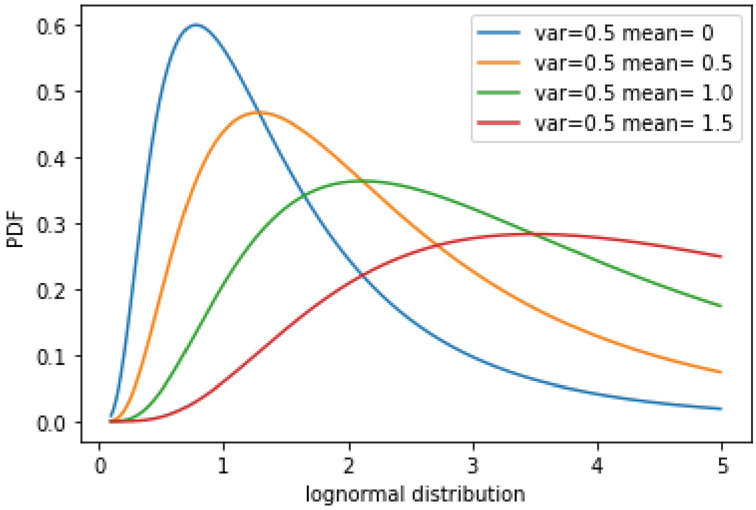
In [29]:

```
print("FIRST PART")
import numpy as np
import matplotlib.pyplot as plt

mui = 0
var=0.5
x = np.arange(0.1,5,.01)
mean=0
pi=3.14

print('Lognormal distribution plot where mean is same and variance is varying:')
for i in range(4):
    f = 1/np.sqrt(2*pi*var*x)*np.exp((-1/(2*var))*np.power((np.log(x)-mui),2))
    plt.plot(x,f,label='mean=0 var= {}'.format(var))
    var=var+0.5
plt.ylabel('PDF')
plt.xlabel('lognormal distribution')
plt.legend()
plt.show()

mui = 0
var=0.5
x = np.arange(0.1,5,.01)
mean=0
pi=3.14

print('Lognormal distribution plot where variance is same and mean is varying:')
for i in range(4):
    f = 1/np.sqrt(2*pi*var*x) * np.exp((-1/(2*var))*np.power((np.log(x)-mui),2))
    plt.plot(x,f,label='var=0.5 mean= {}'.format(mui))
    mui=mui+0.5

plt.ylabel('PDF')
plt.xlabel('lognormal distribution')
plt.legend()
plt.show()
```

FIRST PART
Lognormal distribution plot where mean is same and variance is varying:



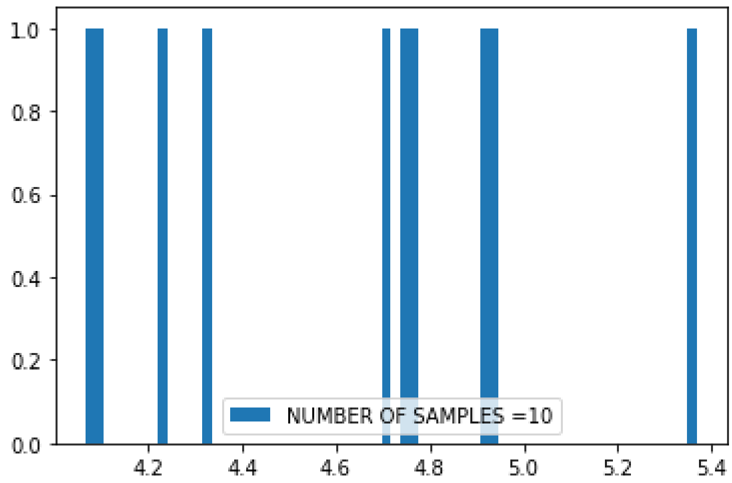
Lognormal distribution plot where variance is same and mean is varying:

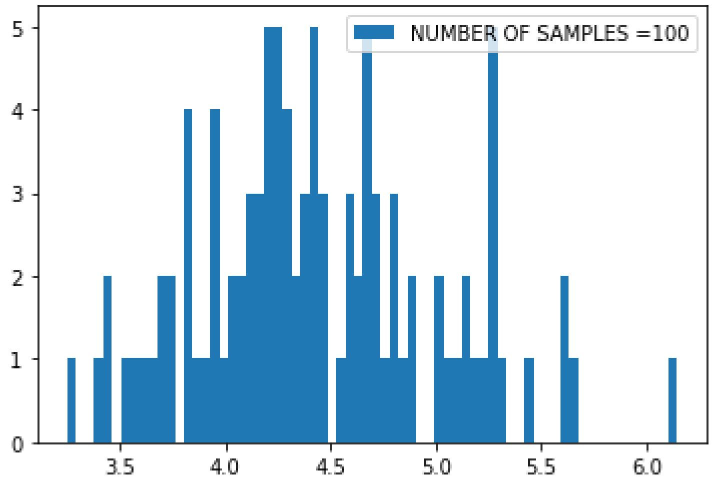


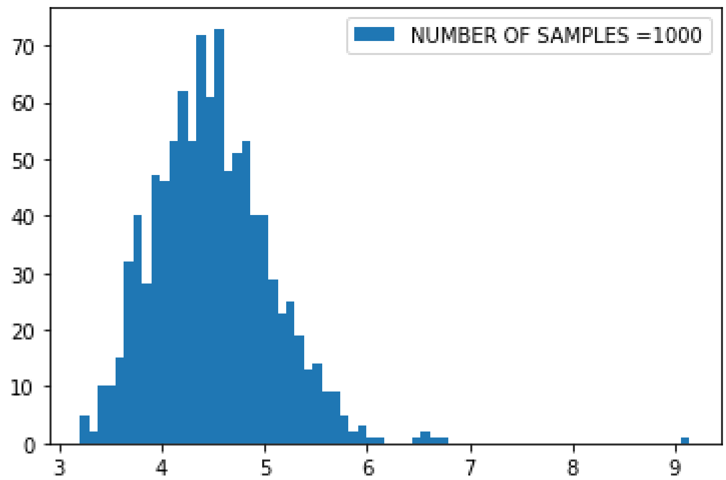
In [32]:

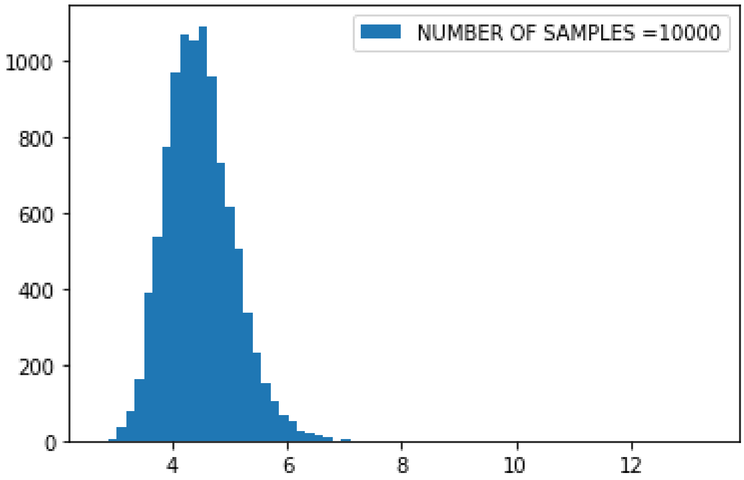
```
print("SECOND PART")
import numpy as np
import matplotlib.pyplot as plt
#plotting for different number of samples in lognormal distribution
mui=1
varinace=1
n=100
ns=10
for i in range(4):

    samplemean=[]
    for j in range(ns):
        sum1=0
        x = np.random.lognormal(1,1,n)
        for k in x:
            sum1=sum1+k
        samplemean.append(sum1/n)
    plt.hist(samplemean,68,label="NUMBER OF SAMPLES ={}".format(ns))
    plt.legend()
    plt.show()
    ns=ns*10
```

```
SECOND PART
```


NUMBER OF SAMPLES =10


NUMBER OF SAMPLES =100


NUMBER OF SAMPLES =1000



In [ ]:

```

```