Practical 8

Write a python program to compute and explore normal distribution, central limit theorem, point estimate, interval estimation and hypothesis testing.

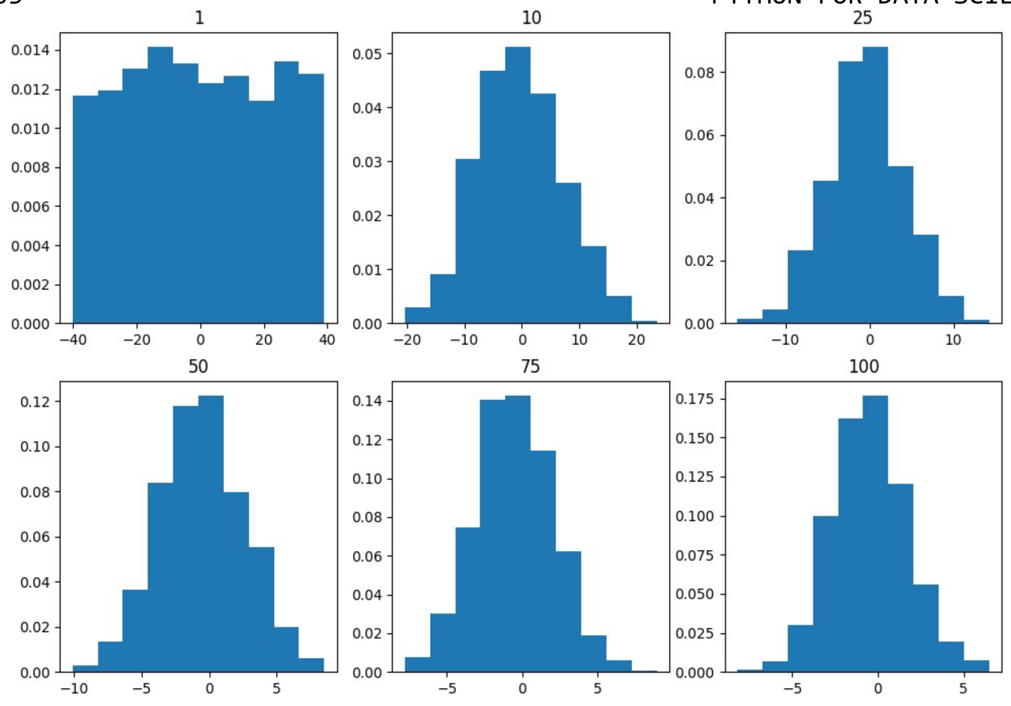
Importing Libraries

In [14]:
```
from scipy.stats import norm
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import ttest_ind
```

In [15]:
```
a,b=4.25,3.57
rv=norm(a,b)
quantile=np.arange(0.01,1,0.1)
r=norm.rvs(a,b)
print("Random variates:",r)
r=norm.pdf(a,b,quantile)
print("probability distribution:",r)
```

```
Random variates: 6.038100845292055
probability distribution: [0.00000000e+00 1.82499839e-08 1.00425806e-02 1.1606756
3e-01
 2.45929046e-01  3.21588382e-01  3.51347133e-01  3.55195341e-01
 3.46247027e-01  3.31601548e-01]
```

In [22]:
```
num=[1,10,25,50,75,100]
means=[]
for j in num:
    np.random.seed(1)
    x=[np.mean(np.random.randint(-40,40,j)) for i in range(1000)]
    means.append(x)
k=0
fig,ax=plt.subplots(2,3,figsize=(12,8))
for i in range(0,2):
    for j in range(0,3):
        ax[i,j].hist(means[k],10,density=True)
        ax[i,j].set_title(label=num[k])
        k=k+1
plt.show()
```



```
In [10]: from pylab import *
         import scipy.stats
         n=1000
         gamma=0.95
         mu=10
         sigma=2
         mu_hat=mean(x)
         sigma_hat=std(x,ddof=1)
         print("Sample Mean : ",mu_hat)
         print("Sample Standard Deviation : ",sigma_hat)
         l=scipy.stats.t.ppf((1-gamma)/2,n-1)
         u=scipy.stats.t.ppf(1-(1-gamma)/2,n-1)
         print("Confidence interval mu_hat:(%f,%f)"%(mu_hat+a*sigma_hat/sqrt(n),mu_hat+u*
         l=scipy.stats.chi2.ppf((1-gamma)/2,n-1)
         u=scipy.stats.chi2.ppf(1-(1-gamma)/2,n-1)
         print("Confidence  interval  sigma_hat:(%f,%f)"%(sqrt((n-1)/u)*sigma_hat,sqrt((n-1
```

```
Sample Mean :  -0.49524
Sample Standard Deviation :  2.1991484252310474
Confidence  interval  mu_hat:(-0.194813,-0.358773)
Confidence  interval  sigma_hat:(2.106811,0.072733)
```

```
In [21]: data1=[0.873,2.817,0.121,-0.945,-0.055,-1.436,0.36,-1.478,-1.637,1.869]
         data2=[0.125,-0.259,0.596,-1.569,-0.698,1.236,0.654,-1.121,0.856,0.879]

         stat,p=ttest_ind(data1,data2)

         print('Stat = %.3f, p = %.3f'%(stat,p))
         if p>0.05:
             print('Probably the Same distribution')
         else:
             print('Probably Different distribution')
```

```
Stat = -0.038, p = 0.970
Probably the Same distribution
```