In [ ]:

```
import numpy as np
```

In [ ]:

```
np.random.seed(1234)
np.random.uniform(0, 100), np.random.normal(size=(2,3))
# Mersenne Twister - pseudo random number generator
```

Out[ ]:

```
(19.151945037889227, array([[-1.0336313 ,  2.02683258,  0.66250904],
        [ 0.67524634, -0.94029827, -0.95658428]]))
```

In [ ]:

```
rng = np.random.default_rng(1234)
rng.uniform(0, 100), rng.normal(size=(2,3))
```

Out[ ]:

```
(97.66997666981422, array([[ 0.06409991,  0.7408913 ,  0.15261919],
        [ 0.86374389,  2.91309922, -1.47882336]]))
```

In [ ]:

```
# quasi random number generators
from scipy.stats import qmc
# sobol sequence
gen = qmc.Sobol(d=3, seed=1234)
gen.random_base2(m=2) # generate 2^m numbers
```

Out[ ]:

```
array([[0.99361137, 0.28360828, 0.74058864],
       [0.03475311, 0.95930153, 0.21378241],
       [0.33277869, 0.02253785, 0.82184589],
       [0.6387977 , 0.7218523 , 0.34877372]])
```

In [ ]:

```
# simulated option price
S0, K, T, r, q, vol = 180, 160, 0.5, 0.02, 0.015, 0.20
```

In [ ]:

```
n = 100000
rng = np.random.default_rng(1234)
z = rng.standard_normal(size=n)
S = S0*np.exp((r-q-0.5*vol**2)*T +vol*np.sqrt(T)*z)
disc_payoffs = np.exp(-r*T)*np.maximum(S - K, 0.0)
sim_price = np.mean(disc_payoffs)
sim_price
```

Out[ ]:

```
22.881914778296434
```

In [ ]:

```
def simulated_price(S0, K, T, r, q, vol, n):
  z = rng.standard_normal(size=n)
  S = S0*np.exp((r-q-0.5*vol**2)*T +vol*np.sqrt(T)*z)
  disc_payoffs = np.exp(-r*T)*np.maximum(S - K, 0.0)
  sim_price = np.mean(disc_payoffs)
  std_error = np.std(disc_payoffs, ddof=1)/np.sqrt(n)
  return sim_price, std_error
```

In [ ]:

```
simulated_price(S0, K, T, r, q, vol, n=1000000)
```

Out[ ]:

```
(22.86255531661002, 0.021936586885224922)
```

In [ ]:

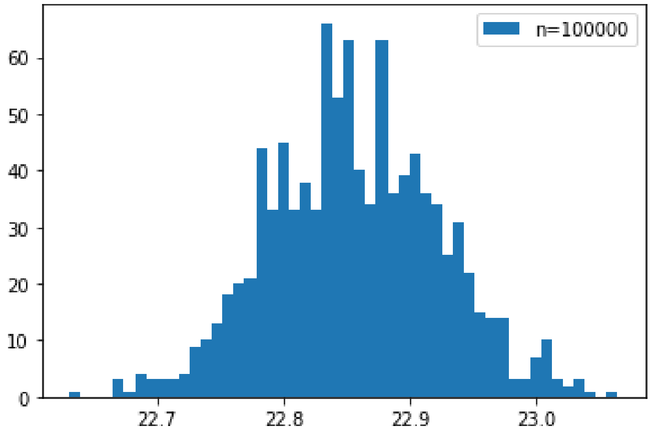
```
# empirical distribution of simulated price
sim_prices = np.zeros(shape=1000)
for i in range(1000):
  sim_prices[i], _ = simulated_price(S0, K, T, r, q, vol, n)
```

In [ ]:

```
import matplotlib.pyplot as plt
```

```
In [ ]: plt.hist(sim_prices, bins=50, label='n=100000')
        plt.legend()
```

```
Out[ ]: <matplotlib.legend.Legend at 0x7f795b6e8760>
```


n=100000
60
50
40
30
20
10
0
22.7
22.8
22.9
23.0