Maximum Likelihood Estimation

In [ ]:

```
import numpy as np
import math
from scipy.optimize import minimize
```

In [ ]:

```
mean = 10
std = 20
```

In [ ]:

```
s = np.random.normal(mean, std, 3000)
```

In [ ]:

```
def likelihood(mean, std, x):
    return (1 / math.sqrt(2 * math.pi * std**2)) * np.exp(-(x - mean)**2 / (2 * st

def log_likelihood(mean, std, data):
    return sum(np.log(likelihood(mean, std, x)) for x in data)
```

In [ ]:

```
neg_log_likelihood = lambda mean: -log_likelihood(mean, std, s)
```

In [ ]:

```
result = minimize(neg_log_likelihood, x0=0.0)
mean_mle = result.x[0]
print(mean_mle)
```

```
9.437499921115245
```

In [ ]:

```
print("Difference between original mean and new mean", mean-mean_mle)
```

```
Difference between original mean and new mean 0.5625000788847547
```

In [ ]: