Problem 3 -> Part 1] Generating Data

Generate N observations from a normal distribution: data = randn(N, 1); This will generate N 1-D samples with mean equal to 0 and variance equal to 1.

```
import pandas as pd
```

```
import numpy as np
```

Following funtion generateObservation(), generates n 1-D observation with mean = 0 and variance = 1.

```
def generateObservation(n):
  observation = np.random.normal(0, 1, n)
  return observation
```

```
generateObservation(20) # will produce 20 random observations, with
```

Estimate the mean and variance of the data for N = 10, 100, 1000 etc.

```
def generateObservation(n):
  observation = np.random.normal(0, 1, n)
  return observation
```

Estimating mean and variance

```
n = input("Enter the number of observation : ")
m = np.mean(generateObservation(int(n)))
v = np.var(generateObservation(int(n)))
```

```
f"mean = {m} and variance = {v}"
```

Estimating mean and variance for N = 10

```
m1 = np.mean(generateObservation(10))
f"mean = {m1}"
```

```
v1 = np.var(generateObservation(10))
f"variance = {v1}"
```

Estimating mean and variance for N = 100

```
m2 = np.mean(generateObservation(100))
f"mean = {m2}"
```

```
v2 = np.var(generateObservation(100))
f"variance = {v2}"
```

Estimating mean and variance for N = 1000

```
m3 = np.mean(generateObservation(1000))
f"mean = {m3}"
```

```
v3 = np.var(generateObservation(1000))
f"variance = {v3}"
```

Modify the code so that the generated data have mean and variance equal to user-specified parameters mean and var.

```
import math
def generateData(m, variance):
  sigma = math.sqrt(variance)
  return np.random.normal(m, sigma, 1000)
```

Please pass the mean and variance of your choice as the arguments for generateData() function. (This function gives 1000 observation)

```
generateData(500, 5)
```

Following function will generate 1000 observation with the user-specified parameters mean and variance

```
import math
def generateYourData():
  m = input("Enter mean : ")
  sigma = input("Enter variance : ")
  def generateData(m, variance):
    sigma = math.sqrt(variance)
    return np.random.normal(m, sigma, 1000)
  return generateData(int(m), int(sigma))
```

call the function generateYourData()

```
generateYourData()
```

Problem 3. Part 2] Verification

```
import math
def generateData(m, variance, n):
  sigma = math.sqrt(variance)
  return np.random.normal(m, sigma, n)
```

```
np.mean(generateData(1, 4, 2000))
```

```
np.var(generateData(1, 4, 2000))
```

```
np.mean(generateData(4, 9, 1000))
```

```
np.var(generateData(4, 9, 1000))
```

For the combined data the mean and variance is very close to theorically calculated combined mean and combined variance

```
np.mean(generateData(2, 7.67, 3000))
```

```
np.var(generateData(2, 7.67, 3000))
```

We can see that the mean and the variance of the combined data is very close to the value we have calculated.

Which is: mean = 2 variance = 7.6667