BRSM Take Home Assignment (21-01-2025)

Submitted by - Ashish Chokhani(2021102016)

```python
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
```

Question 1

a. Assume that your population distribution is N(100,15)

```python
mean = 100
std_dev = 15
```

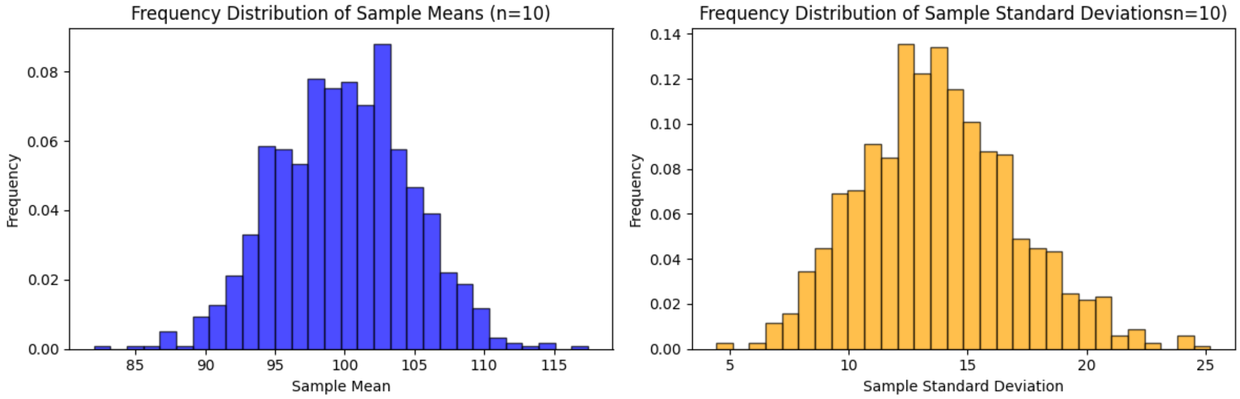
b. Sample 10 random numbers (i.e., your sample size) from N(100,15) and calculate the mean and standard deviation of those numbers

```python
sample_size = 10
sample = np.random.normal(mean, std_dev, sample_size)
mean_sample = np.mean(sample)
std_dev_sample = np.std(sample)
print("Sample Mean:",mean_sample)
print("Sample Standard Deviation:",std_dev_sample)
```

```
Sample Mean: 101.17520214856617
Sample Standard Deviation: 10.865099743121363
```

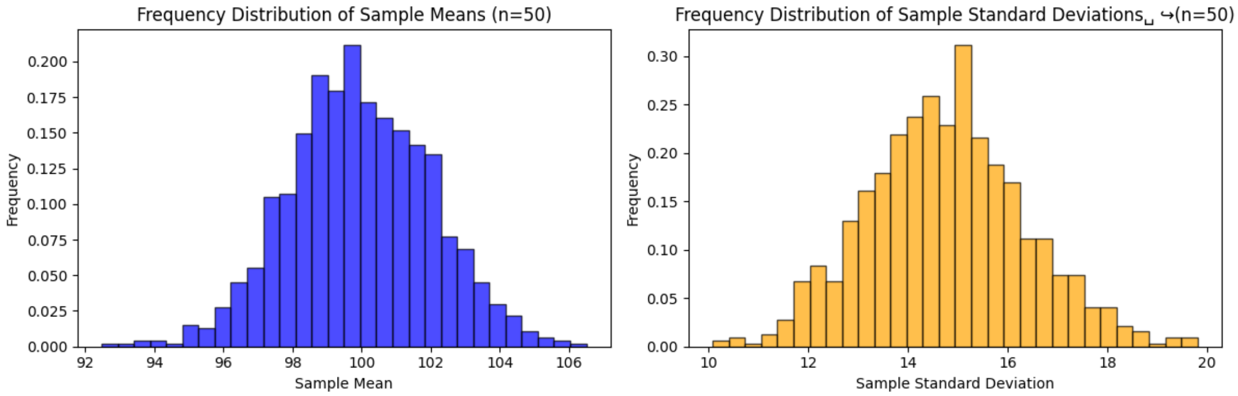
c. Repeat this for 1000 trials, and plot the frequency distribution of the obtained means and standard deviations separately.

```python
num_trials = 1000
means = np.zeros(num_trials)
std_devs = np.zeros(num_trials)
for i in range(num_trials):
    sample = np.random.normal(mean, std_dev, sample_size)
    means[i] = np.mean(sample)
    std_devs[i] = np.std(sample)
plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.hist(means, bins=30, density=True, color='blue', alpha=0.7,
edgecolor='black')
plt.title(f'Frequency Distribution of Sample Means (n={sample_size})')

plt.xlabel('Sample Mean')
plt.ylabel('Frequency')
plt.subplot(1, 2, 2)
plt.hist(std_devs, bins=30, density=True, color='orange', alpha=0.7,
edgecolor='black')
plt.title(f'Frequency Distribution of Sample Standard
Deviationsn={sample_size})')
plt.xlabel('Sample Standard Deviation')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```



d. Repeat steps b and c for 50, 100, 500, and 1500 numbers.

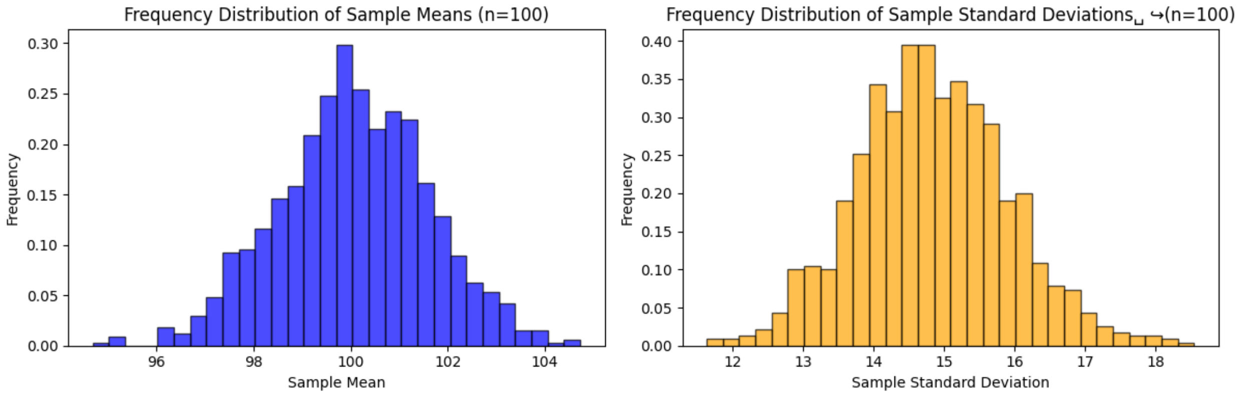
```
sample_sizes = [50,100,500,1500]
for sample_size in sample_sizes:
    print("Sample Size =",sample_size)
    sample = np.random.normal(mean, std_dev, sample_size)
    mean_sample = np.mean(sample)
    std_dev_sample = np.std(sample)
    print("Sample Mean:",mean_sample)
    print("Sample Standard Deviation:",std_dev_sample)

    num_trials = 1000
    means = np.zeros(num_trials)
    std_devs = np.zeros(num_trials)
    for i in range(num_trials):
        sample = np.random.normal(mean, std_dev, sample_size)
        means[i] = np.mean(sample)
        std_devs[i] = np.std(sample)

    plt.figure(figsize=(12, 4))
    plt.subplot(1, 2, 1)
    plt.hist(means, bins=30, density=True, color='blue',
alpha=0.7,edgecolor='black')
    plt.title(f'Frequency Distribution of Sample Means
(n={sample_size})')
    plt.xlabel('Sample Mean')
    plt.ylabel('Frequency')
    plt.subplot(1, 2, 2)
    plt.hist(std_devs, bins=30, density=True, color='orange',
alpha=0.7,edgecolor='black')
    plt.title(f'Frequency Distribution of Sample Standard Deviations␣
↪(n={sample_size})')
    plt.xlabel('Sample Standard Deviation')
    plt.ylabel('Frequency')
    plt.tight_layout()
    plt.show()

Sample Size = 50
Sample Mean: 100.67927469664784
Sample Standard Deviation: 17.069327027129106
```



Frequency Distribution of Sample Means (n=50)

Frequency Distribution of Sample Standard Deviations␣ ↪(n=50)

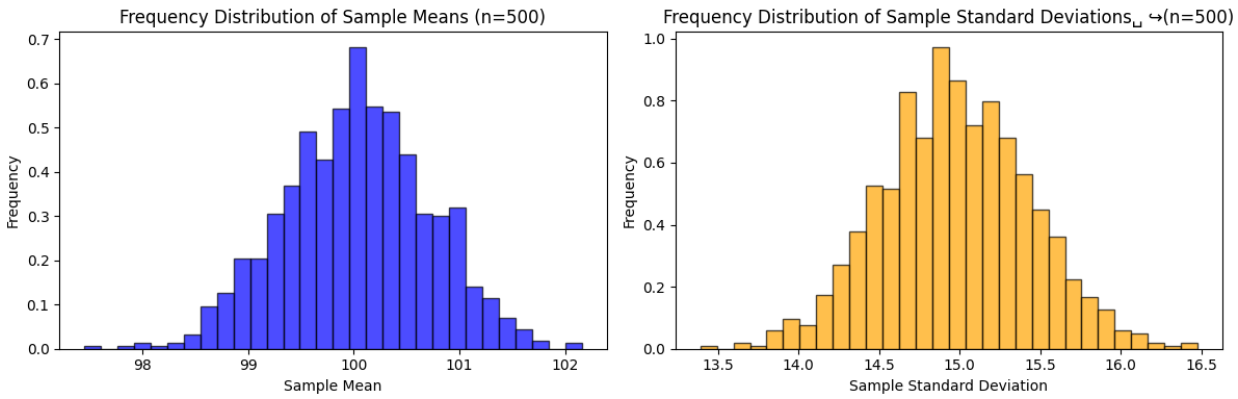
```
Sample Size = 100
Sample Mean: 100.94235976048219
Sample Standard Deviation: 14.343823101952092
```



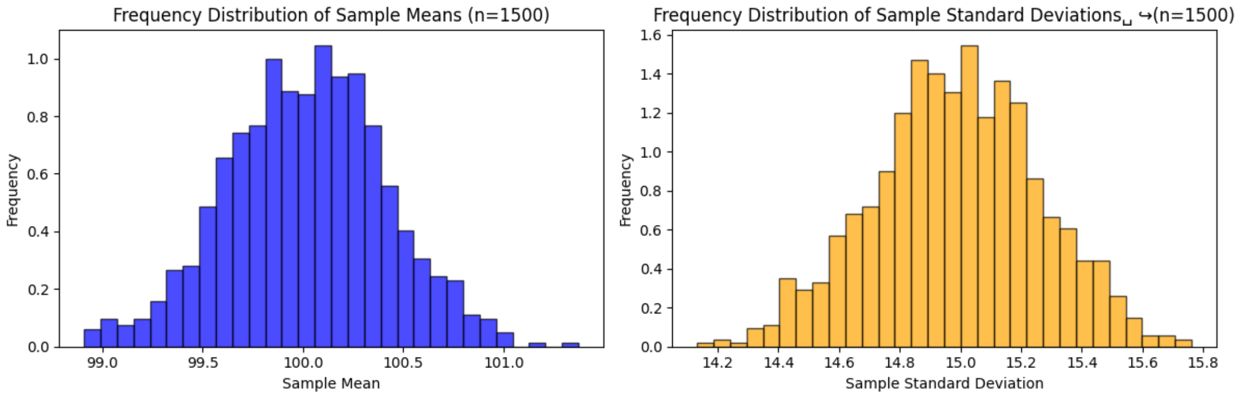
Frequency Distribution of Sample Means (n=100)

Frequency Distribution of Sample Standard Deviations␣ ↪(n=100)

```
Sample Size = 500
Sample Mean: 99.8472477782398
Sample Standard Deviation: 15.680709057453392
```



```
Sample Size = 1500
Sample Mean: 100.79486221261719
Sample Standard Deviation: 15.361824875812749
```



e. State and explain the observations and inferences you make regarding the above-produced histograms w.r.t to central limit theorem.

In the histograms, we observe that as the sample size increases, the distribution of sample means becomes more bell-shaped and symmetric, resembling a normal distribution. The standard deviation of the sample means decreases as the sample size increases, indicating that larger samples provide more precise estimates of the population mean. These observations align with the central limit theorem.

QUESTION 2

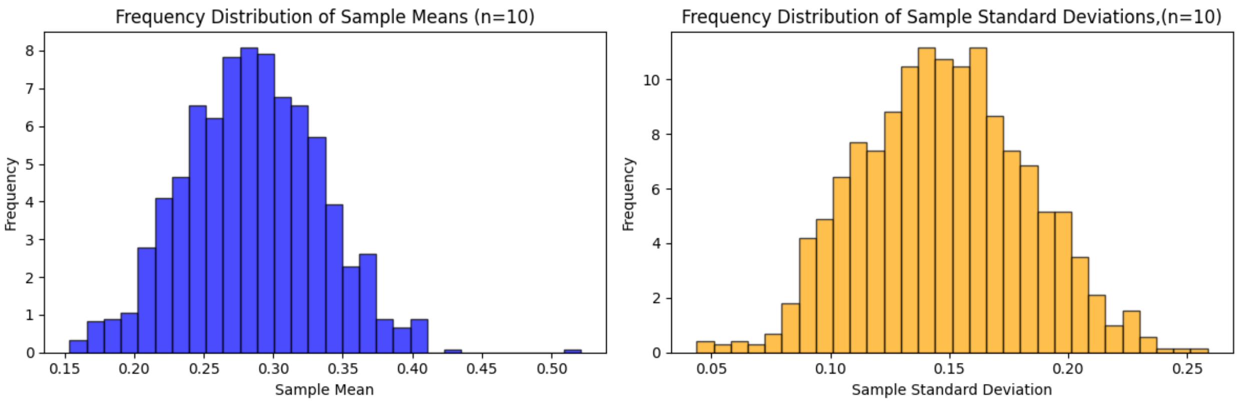
Repeat the above for a population distribution that is a Beta distribution with shape parameters 2 and 5.

```
alpha = 2
beta = 5

mean_beta = alpha / (alpha + beta)
std_dev_beta = np.sqrt((alpha * beta) / ((alpha + beta)**2 * (alpha +
beta + 1)))
```

```
print("Distribution Mean:",mean_beta)
print("Distribution Standard Deviation:",std_dev_beta)
sample_size = 10
sample_beta = np.random.beta(alpha, beta, sample_size)
mean_sample = np.mean(sample_beta)
std_dev_sample = np.std(sample_beta)
print("Sample Mean:",mean_sample)
print("Sample Standard Deviation:",std_dev_sample)
```

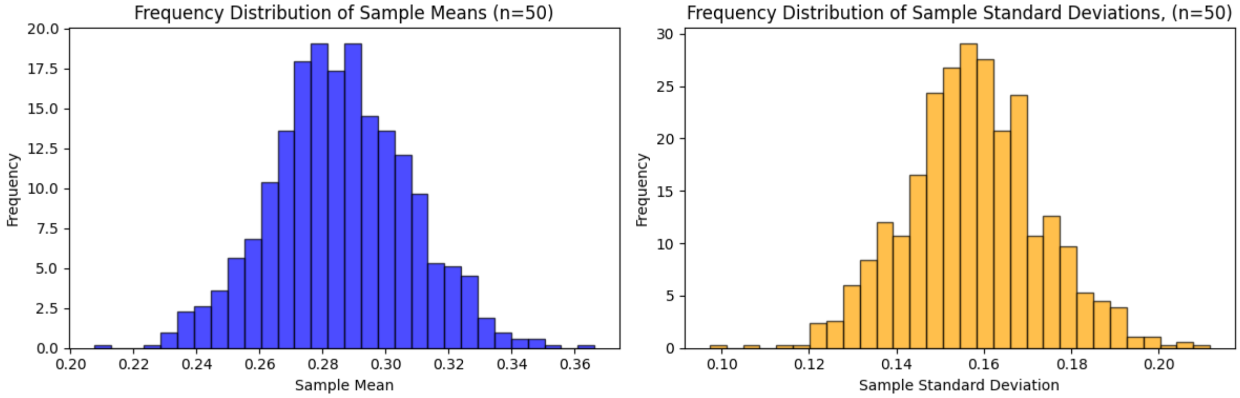
```
Distribution Mean: 0.2857142857142857
Distribution Standard Deviation: 0.15971914124998499
Sample Mean: 0.27298599530316275
Sample Standard Deviation: 0.16746596703231217
```

```
num_trials = 1000
means = np.zeros(num_trials)
std_devs = np.zeros(num_trials)
for i in range(num_trials):
    sample_beta = np.random.beta(alpha, beta, sample_size)
    means[i] = np.mean(sample_beta)
    std_devs[i] = np.std(sample_beta)
plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.hist(means, bins=30, density=True, color='blue',
alpha=0.7,edgecolor='black')
plt.title(f'Frequency Distribution of Sample Means (n={sample_size})')

plt.xlabel('Sample Mean')
plt.ylabel('Frequency')
plt.subplot(1, 2, 2)
plt.hist(std_devs, bins=30, density=True, color='orange',
alpha=0.7,edgecolor='black')
plt.title(f'Frequency Distribution of Sample Standard Deviations,
(n={sample_size})')
plt.xlabel('Sample Standard Deviation')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```



```
sample_sizes = [50,100,500,1500]
for sample_size in sample_sizes:
    print("Sample Size =",sample_size)
    sample_beta = np.random.beta(alpha, beta, sample_size)
    mean_sample = np.mean(sample_beta)
    std_dev_sample = np.std(sample_beta)
    print("Sample Mean:",mean_sample)
    print("Sample Standard Deviation:",std_dev_sample)
    num_trials = 1000
    means = np.zeros(num_trials)
    std_devs = np.zeros(num_trials)
    for i in range(num_trials):
        sample_beta = np.random.beta(alpha, beta, sample_size)
        means[i] = np.mean(sample_beta)
        std_devs[i] = np.std(sample_beta)
    plt.figure(figsize=(12, 4))
    plt.subplot(1, 2, 1)
    plt.hist(means, bins=30, density=True, color='blue',
alpha=0.7,edgecolor='black')
    plt.title(f'Frequency Distribution of Sample Means
(n={sample_size})')
    plt.xlabel('Sample Mean')
    plt.ylabel('Frequency')
    plt.subplot(1, 2, 2)
    plt.hist(std_devs, bins=30, density=True, color='orange',
alpha=0.7,edgecolor='black')
    plt.title(f'Frequency Distribution of Sample Standard Deviations,
(n={sample_size})')
    plt.xlabel('Sample Standard Deviation')
    plt.ylabel('Frequency')
    plt.tight_layout()
    plt.show()
```

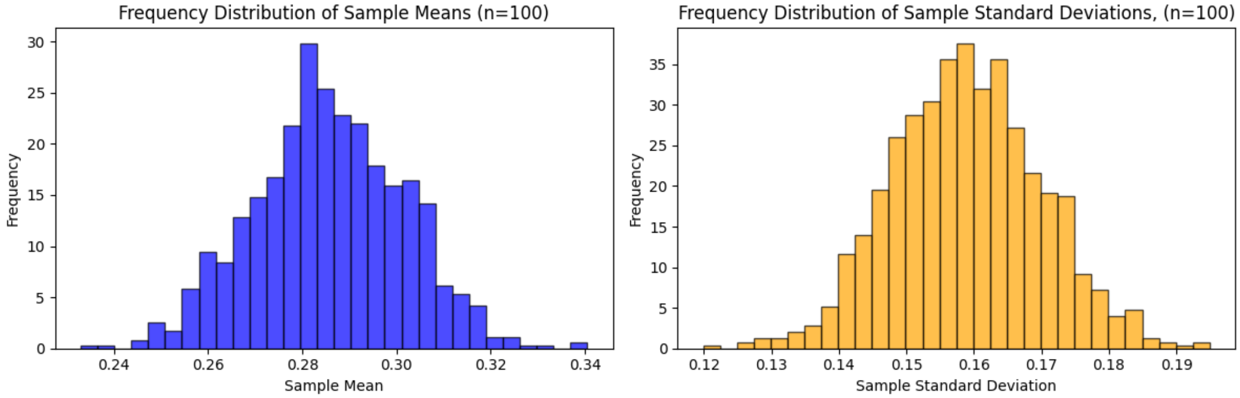
```
Sample Size = 50
Sample Mean: 0.2651051496271604
Sample Standard Deviation: 0.1546243968175475
```



Frequency Distribution of Sample Means (n=50)

Frequency Distribution of Sample Standard Deviations, (n=50)

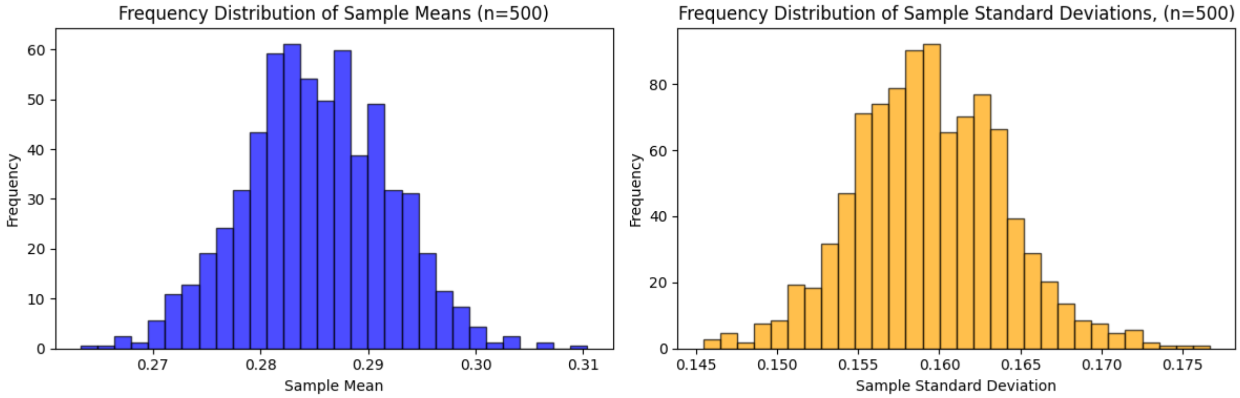
```
Sample Size = 100
Sample Mean: 0.29741064538154854
Sample Standard Deviation: 0.17651741382290384
```



Frequency Distribution of Sample Means (n=100)

Frequency Distribution of Sample Standard Deviations, (n=100)

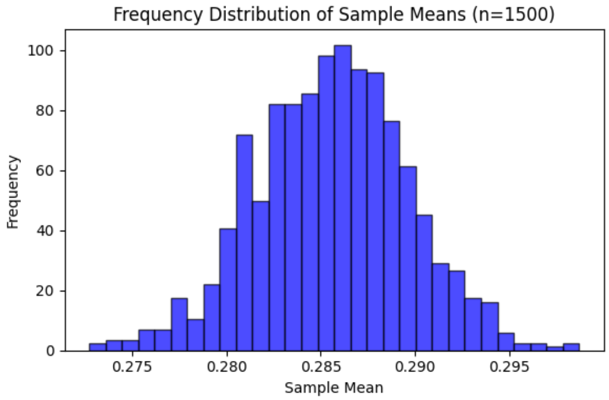
```
Sample Size = 500
Sample Mean: 0.27573161785947153
Sample Standard Deviation: 0.15067284539580048
```

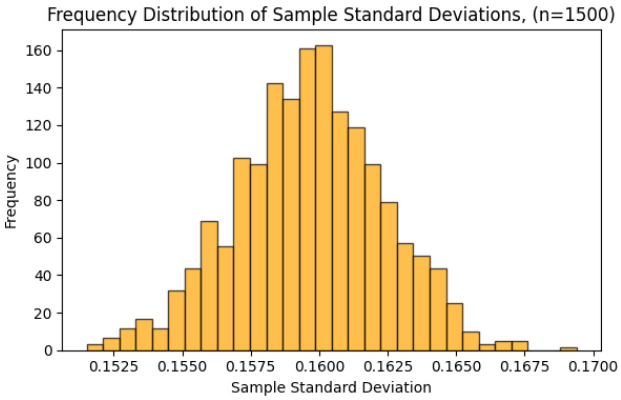


Frequency Distribution of Sample Means (n=500)

Frequency Distribution of Sample Standard Deviations, (n=500)

```
Sample Size = 1500
Sample Mean: 0.284882953222717
Sample Standard Deviation: 0.15987801928898857
```





Repeat the above for a population distribution that is a Beta distribution with shape parameters 2 and 5. Do you need a larger or smaller sample size now so that your sample estimate of the population mean is accurate?

- We observe that, as sample size increases the distribution tends to become more normal but it is still influenced by the original shape of beta distribution which is skewed and not symmetric.
- The distribution of sample means take longer to converge to a normal distribution compared to the normal distribution in Question 1.

Question 3

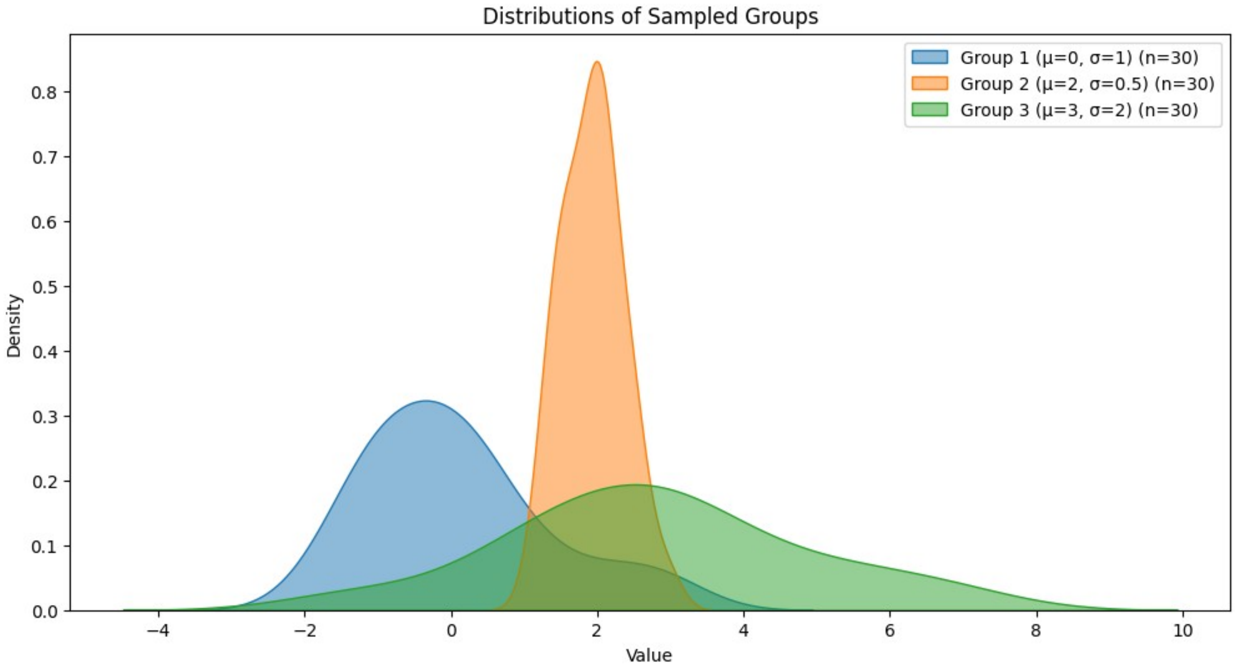
a. Sample 30 points from each of the normal distributions of mean 0 and SD 1, mean 2 and SD 0.5, and mean 3 and SD 2.

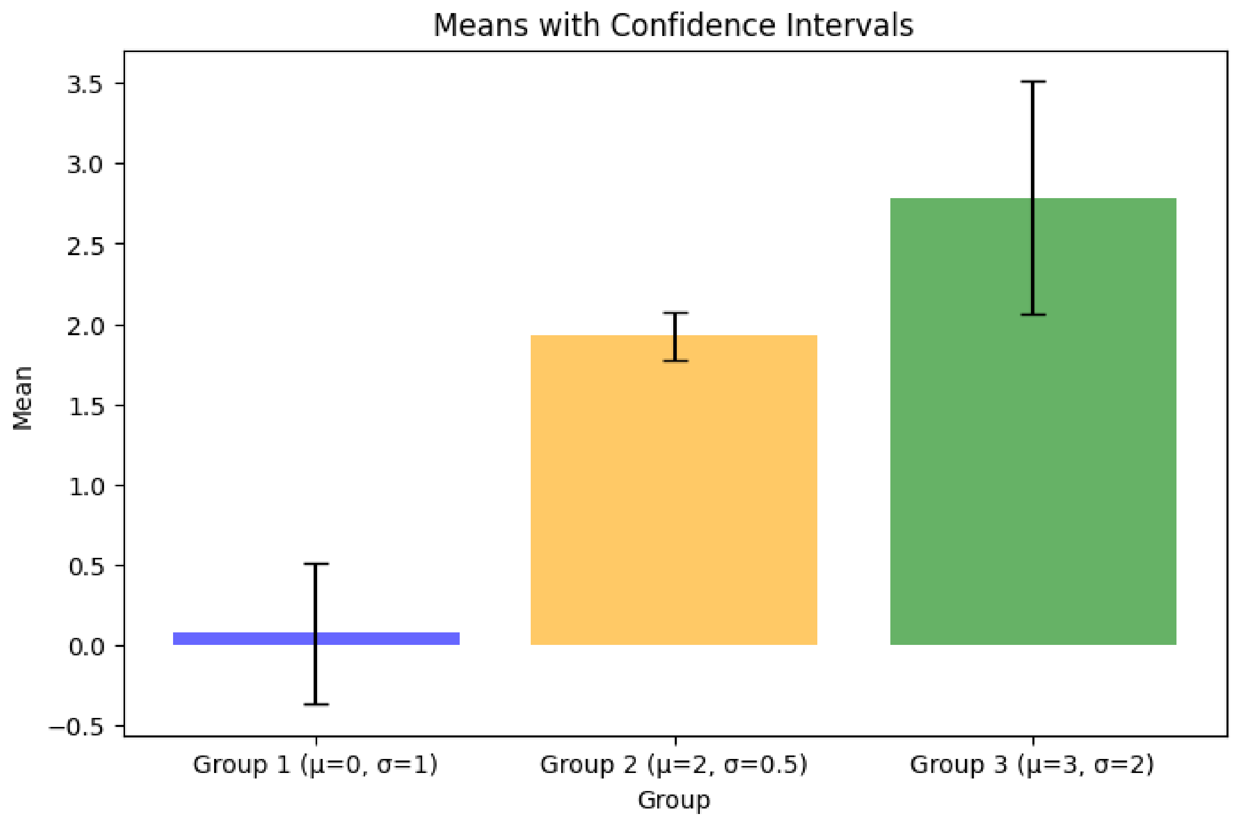
b. Plot the means and distributions of the three groups using the appropriate visualization tools. Include an error bar for each group indicating the confidence interval.

c. Repeat the above two steps by sampling 70 and 100 points

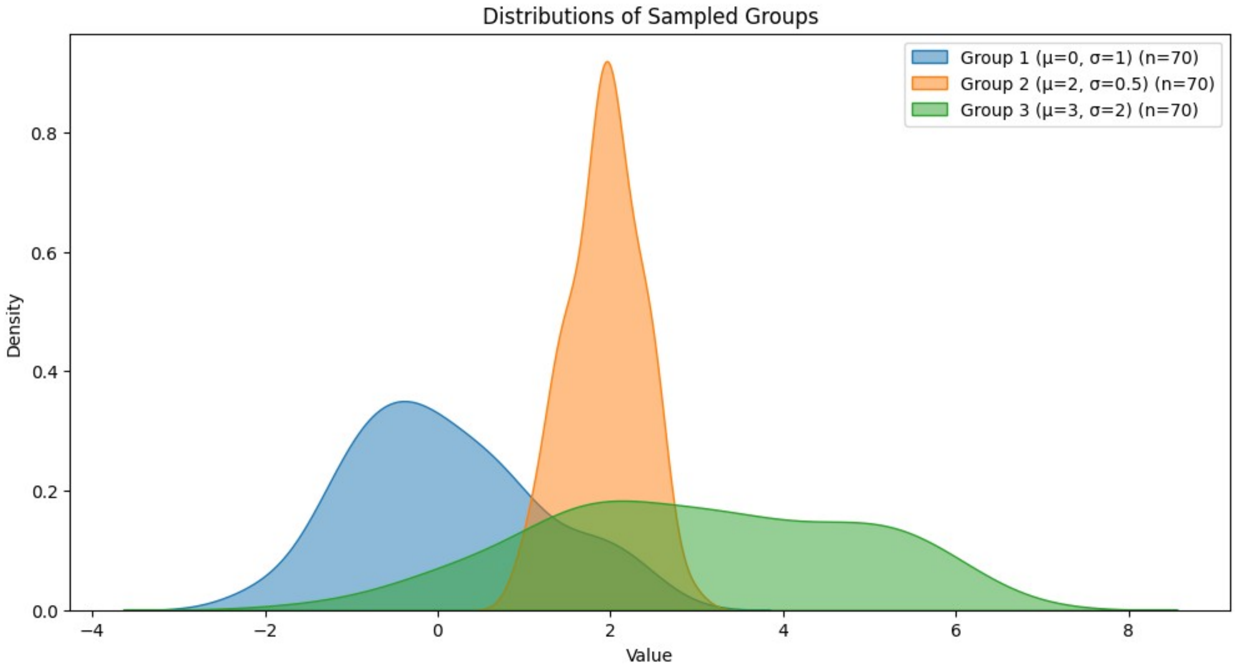
```
def calculate_confidence_interval(data, confidence=0.95):
    """
    Calculate the confidence interval for the mean of the data.
    """
    mean = np.mean(data)
    std_error = np.std(data, ddof=1) / np.sqrt(len(data))
    margin_of_error = norm.ppf((1 + confidence) / 2) * std_error
    return mean, mean - margin_of_error, mean + margin_of_error

def plot_groups_with_confidence_intervals(group_data, group_labels,
sample_sizes):
    """
    Plot the means, distributions, and confidence intervals for each
group.
    """
    num_groups = len(group_data)
    plt.figure(figsize=(12, 6))

    # Plot the distributions
    for i, (data, label) in enumerate(zip(group_data, group_labels)):
        sns.kdeplot(data, fill=True, alpha=0.5, label=f"{label}
(n={sample_sizes[i]})")

    plt.title("Distributions of Sampled Groups")
    plt.xlabel("Value")
    plt.ylabel("Density")
    plt.legend()
    plt.show()

    # Plot the means with error bars
    means = []
    lower_bounds = []
    upper_bounds = []

    for data in group_data:
        mean, lower, upper = calculate_confidence_interval(data)
        means.append(mean)
        lower_bounds.append(lower)
        upper_bounds.append(upper)

    plt.figure(figsize=(8, 5))
    x = np.arange(num_groups)
    plt.bar(x, means, yerr=[np.array(means) - np.array(lower_bounds),
                            np.array(upper_bounds) - np.array(means)],

            capsize=5, alpha=0.6, color=['blue', 'orange', 'green'],
tick_label=group_labels)
    plt.title("Means with Confidence Intervals")
    plt.ylabel("Mean")
    plt.xlabel("Group")
    plt.show()

# Parameters for the distributions
params = [
    (0, 1),    # mean=0, std_dev=1
    (2, 0.5),  # mean=2, std_dev=0.5
    (3, 2)     # mean=3, std_dev=2
]

sample_sizes = [30, 70, 100]
for sample_size in sample_sizes:
    print(f"Sample Size: {sample_size}")

    # Sampling data from the distributions
    group_data = [np.random.normal(mean, std_dev, sample_size) for
mean, std_dev in params]
    group_labels = [f"Group {i+1} (μ={mean}, σ={std_dev})" for i,
(mean, std_dev) in enumerate(params)]

    # Plot distributions and confidence intervals
    plot_groups_with_confidence_intervals(group_data, group_labels,
[sample_size] * len(params))
```

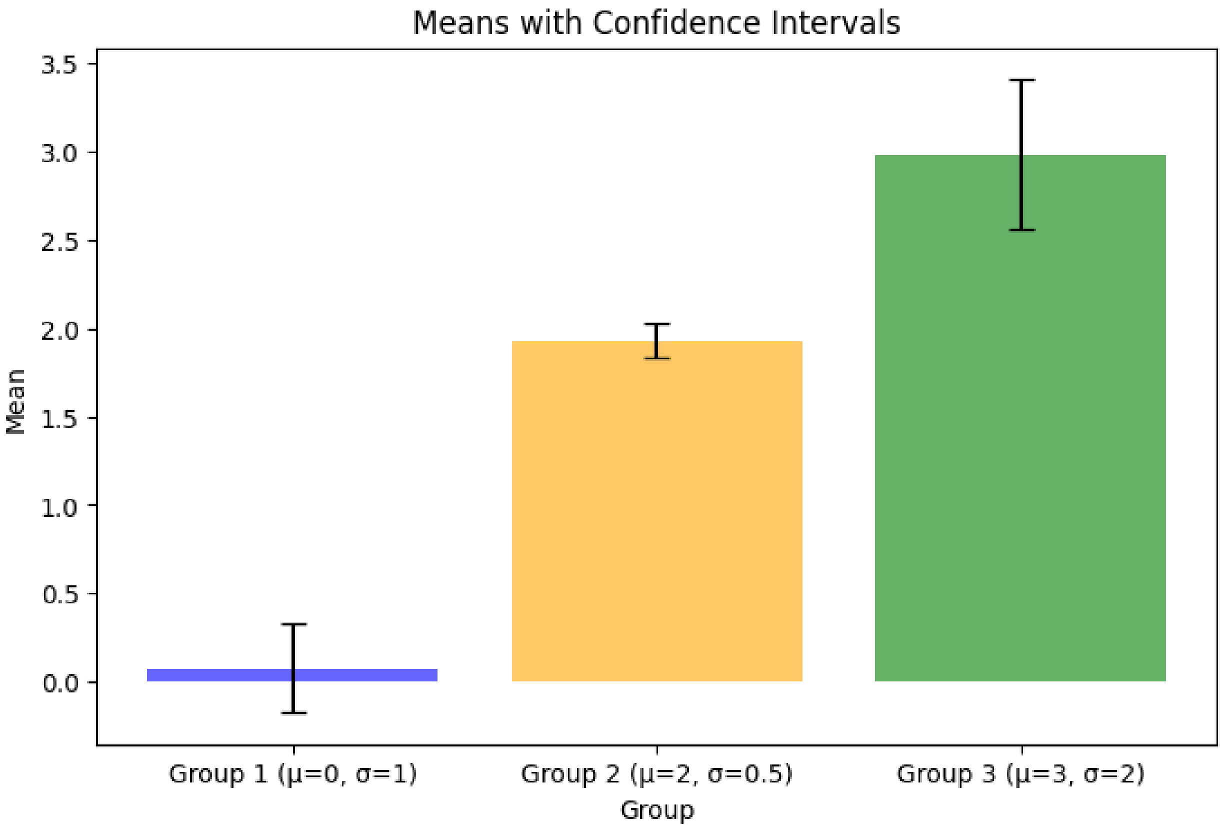
```
Sample Size: 30
```



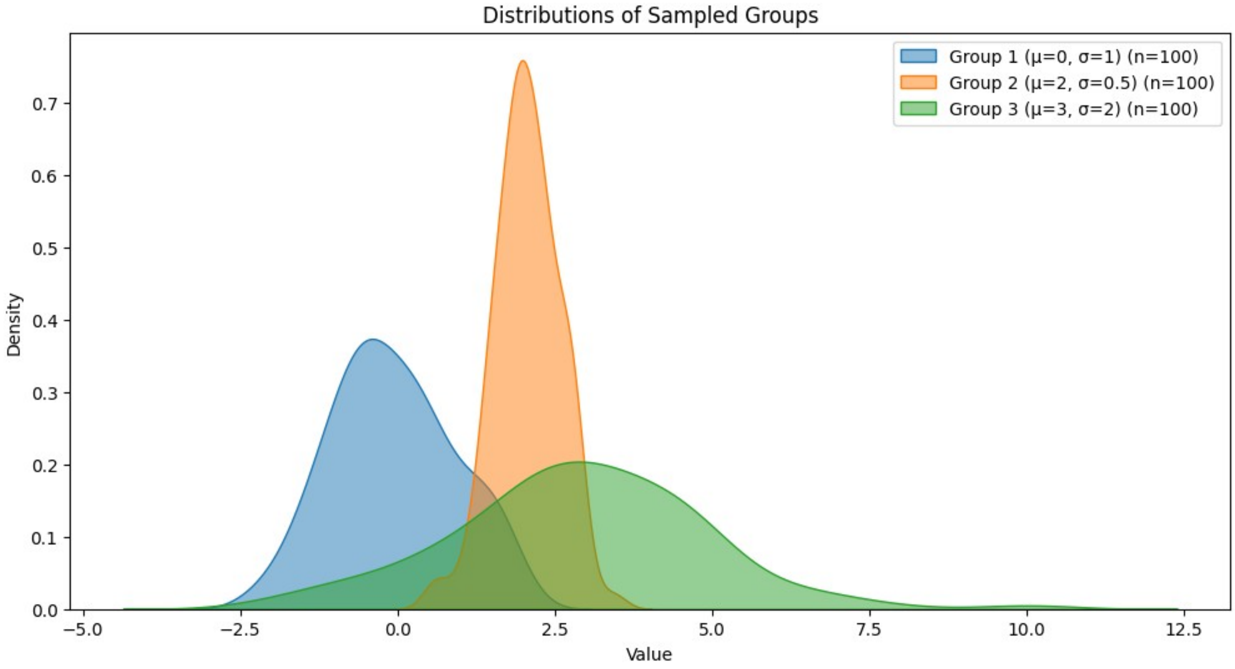


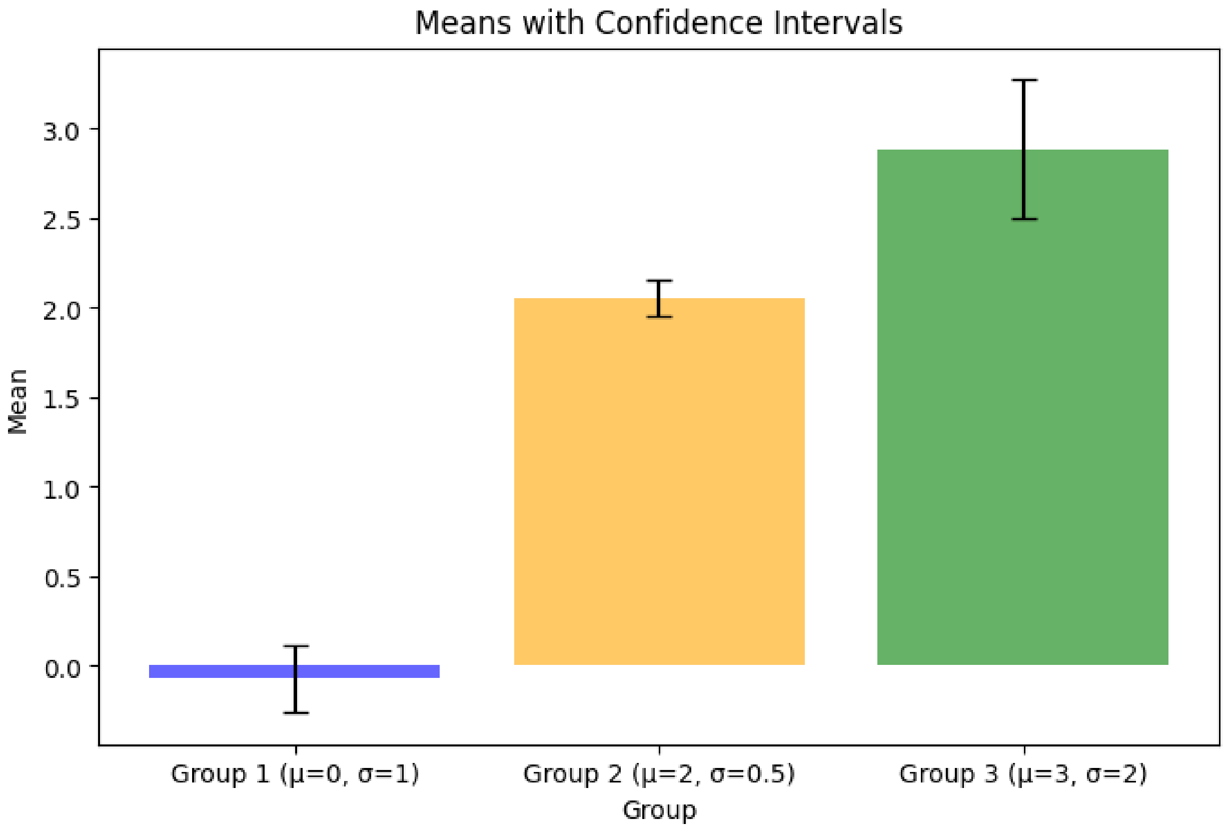
```
Sample Size: 70
```





```
Sample Size: 100
```





d. How does the condence interval change along with the increase in the number of sampled points?

Part d: Effect of sample size on confidence intervals As the sample size increases:

- The confidence interval becomes narrower.
- The margin of error decreases because the standard error (sigma/sqrt(n)) is inversely proportional to the square root of the sample size.

e. Write down your interpretation of the obtained condence intervals.

- Confidence intervals provide a range within which the population mean is likely to fall with a certain confidence level (e.g., 95%).
- Smaller sample sizes result in wider confidence intervals, reflecting greater uncertainty in estimating the population mean.
- Larger sample sizes reduce uncertainty, leading to narrower confidence intervals, providing more precise estimates of the population mean.