Approach

I created the program in Python and used Google Colab as my IDE. The code can be run by following the link https://colab.research.google.com/drive/1CsSwt3bBIScntpiKpFIKfMtVI0L2YUth?usp=sharing or by running the .ipynb file that was attached to this Canvas Submission. I have also attached a pdf of what the expected results should be to this Canvas submission.

I used the following Python libraries: NumPy, SciPy, and Matplotlib. I used NumPy for generating samples of random variables and storing them in an array, SciPy for generating pdf and cdf of Gaussian random variables, and Matplotlib for creating the plots.

I will outline the general approach I followed for each question:

Question 1:

- In total, I created 4 plots
- I chose to plot the sum of n=100, 200, 500, and 1000 Bernoulli(p=0.4) RV with sample sizes of 10000, 20000, 50000, 100000 respectively
- I increased the number of samples as n increases because the variance of the PMF of the sum increases
- For each plot, I used a stem plot to represent the PMF of the sum and I overlaid the corresponding Gaussian pdf
- The Gaussian RV had a mean of 0.4*n and standard deviation of sqrt(0.24*n)

Question 2:

- I followed a similar procedure as Question 1 where I plotted the sum of n=100, 200, 500, and 1000 Poisson(lambda=5) RV with sample sizes of 10000, 20000, 50000, 100000 respectively
- This time, I used a histogram instead of a stem plot since a stem plot would be too cluttered due to the much larger variance of sums
- The Gaussian RV had a mean of 5*n and a standard deviation of sqrt(5*n)

Question 3:

- I generated a Uniform random variable in [0,1]
- I then transformed the uniform samples to those of a gaussian random variable with mean 3 and standard deviation srt(2) using the inverse CDF
- The samples are sorted so that the empirical CDF can be generated
- The empirical and theoretical CDF are overlaid on the sample plot for comparison

Question 4:

- I generated 10,000 samples of a Binomial(n=5, p=0.4) Random Variable using a for loop
- For each sample, I kept track of the running mean and stored it in an array
- I then plotted the sample means and also the true mean
- I did the same approach for 10,000 samples of the Bernoulli(p=0.3) random variable

Question 5:

- I created a zero-mean, unit variance Gaussian random variable
- I took 10,000 samples from this random variable and stored in a numpy array
- I calculated X^2 and I obtained the estimated mean for X^2

Results

Question 1:

- As n increases, we observe that the PMF becomes closer to a Gaussian distribution as n increases
- In the beginning, we observe some outlier points in the PMF that deviate from the Gaussian curve, but the frequency of these decrease as n
- As per the Central Limit Theorem, the sum of n Bernoulli(p=0.3) random variables will converge to a normal distribution of mean n*0.3 and standard deviation of sqrt(n*0.3*(1-0.3))

Question 2:

- Similar observations to Question 1 where the PMF becomes closer to a Guassian distribution as n increases
- We notice that the PMF is definitely more “blocky” compared to the PMF in Question 1 and this can be explained by the fact that we use a histogram instead of a stem plot and also the larger variance
- Since histogram clumps values into bins, bin width has an effect on the smoothness of the PMF
- Nonetheless, as n increases, the PMF becomes less “blocky” and more smooth and will resemble a normal distribution of mean n*5 and standard deviation sqrt(n*5) which follows the Central Limit Theorem

Question 3:

- The empirical and theoretical CDF are very close to each other as expected

Question 4:

- For both plots, I observed that the sample means approach the true mean as n increases
- This is what is expected as per the law of large numbers

Question 5:

- The estimated mean for X^2 is 0.9939478554328285 which is close to 1
- This aligns with what should be expected because Var(X) = E[X^2] - (E[X])^2 = 1
- Since E[X] = 0, this means that E[X^2] = 1