QUESTION 1

A-) Binomial distribution with n=40, p=0.7 and Binomial distribution with n=50, p=0.5

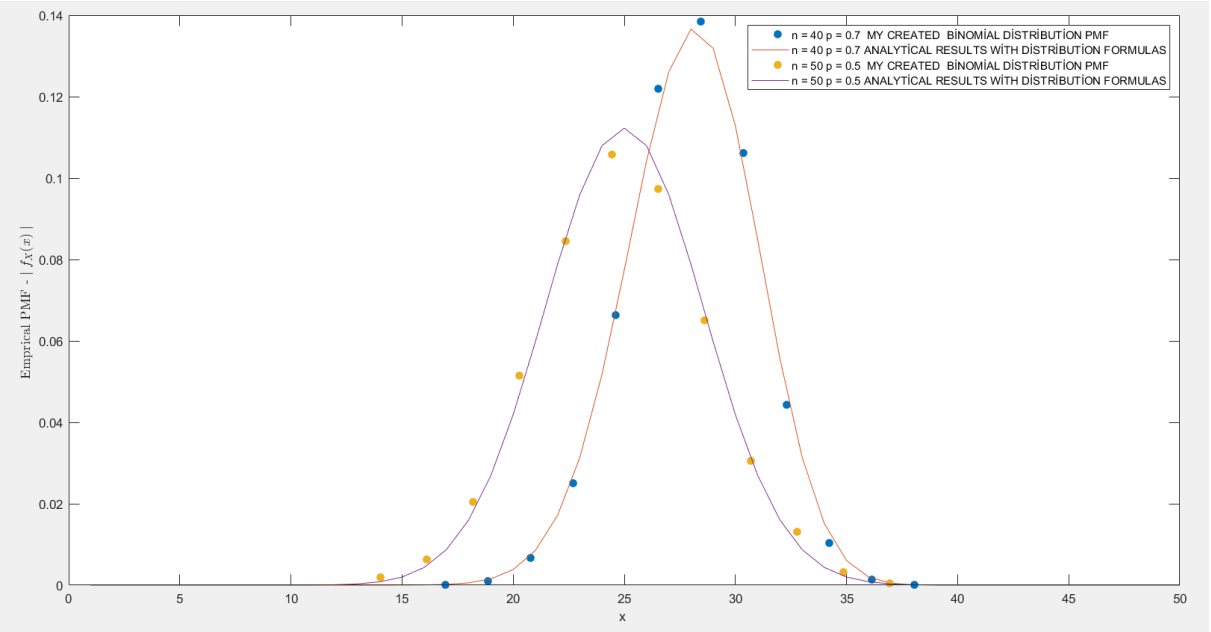
A Bernoulli random variable has two possible outcomes: 0 or 1. A binomial distribution is the sum of independent and identically distributed Bernoulli random variables. The number of successes in n Bernoulli trials, X, has a binomial distribution. So I used the Bernoulli random variable to generate the binomial distribution.

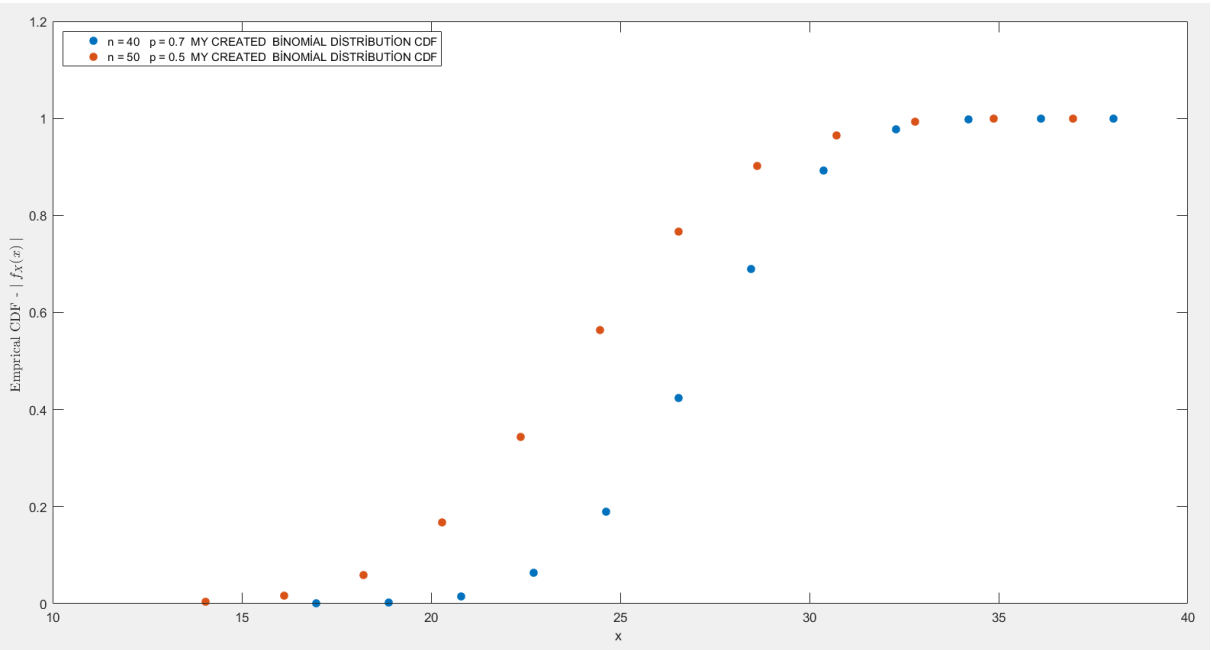
I got the sample numbers and n, p values from the user. I sent these numbers to the binomGenerate (noSamples,p,n) function, which returned us our binomial distribution samples. Called the bernouillirv (n, p) function as many as the number of samples we sent into this function. The bernouillirv (n, p) function also counted and returned how many of the n samples were successful. We recorded our success numbers in a series and this was our binomial distribution.

Then I created the histogram of this series using the array we created. Finally, we calculated Probability mass function and Cumulative distribution function from this histogram and plotted them.

I also drew the relevant analytical results using the binomial distribution formula. Finally, using myMeanFunction (incomingSamples) and myVarianceFunction (incomingSamples) functions average and variance of binomial distributions that we create I calculated the values and printed them on the screen.

The graphics I have drawn are shown below.





Command Window

```
Hesapladigimiz n=40 p=0,7 deki mean = 28.002600
Hesapladigimiz n=40 p=0,7 deki variance = 8.328193

Hesapladigimiz n=50 p=0,5 deki mean = 24.943300
Hesapladigimiz n=50 p=0,5 deki variance = 12.529285
fx >>
```

B-) Geometric distribution with p=0.5 and Geometric distribution with p=0.3

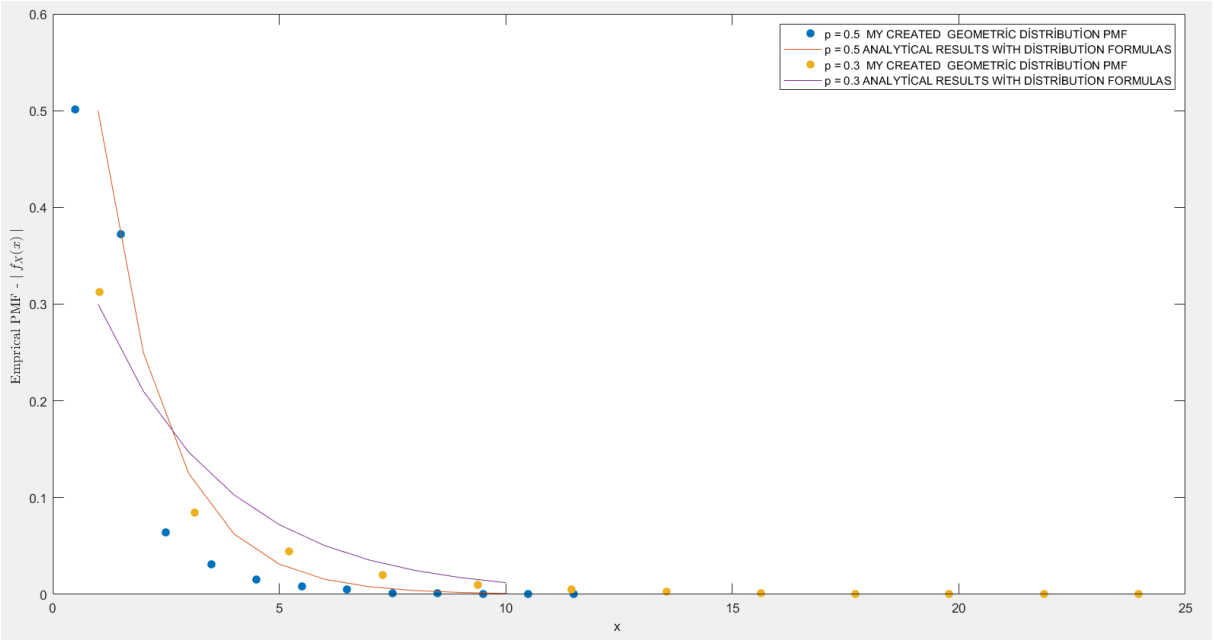
A Bernoulli random variable has two possible outcomes: 0 or 1. The geometric distribution represents the number of failures before you get a success in a series of Bernoulli trials. The number of Bernoulli trials, X, to achieve the 1st success. So I used the Bernoulli random variable to generate the geometric distribution.

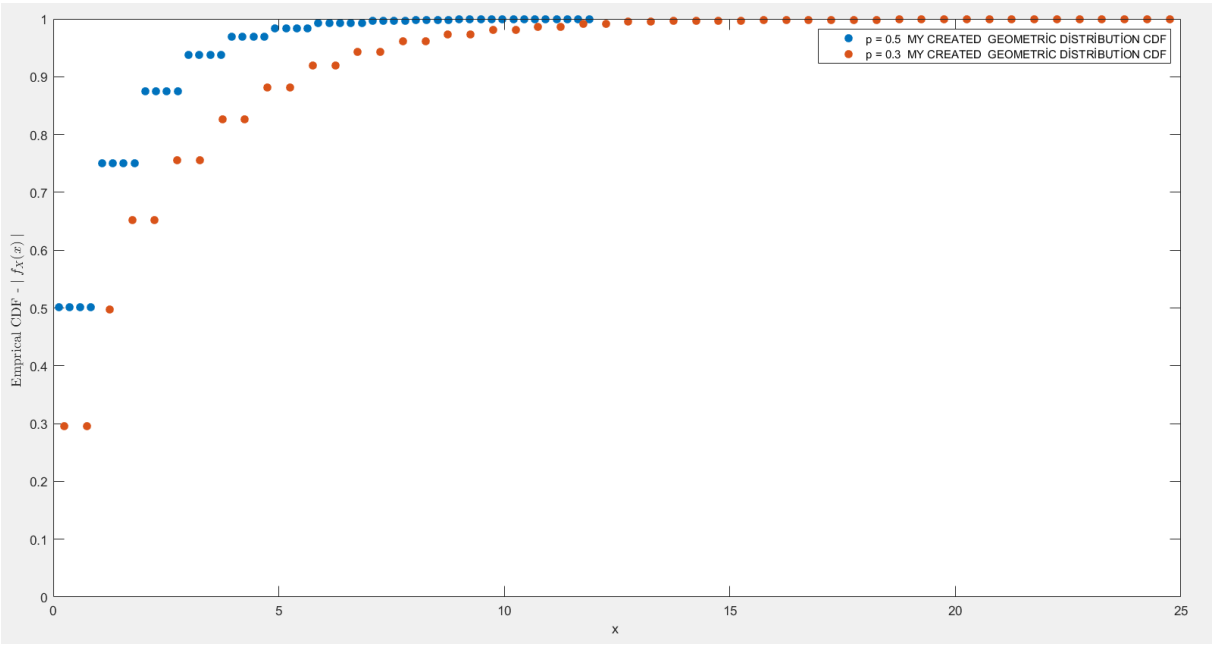
I got the sample numbers and p values from the user. I sent these numbers to the geometricGenerate(p,samplesize) function, which returned us our geometric distribution samples. Until this function was successful, Bernouilli produced a random variable and counted the number of its failures until the first success. We recorded these numbers of failures in an array, and this became our geometric distribution.

Then I created the histogram of this series using the array we created. Finally, we calculated Probability mass function and Cumulative distribution function from this histogram and plotted them.

I also drew the relevant analytical results using the geometric distribution formula. Finally, using myMeanFunction (incomingSamples) and myVarianceFunction (incomingSamples) functions average and variance of geometric distributions that we create I calculated the values and printed them on the screen.

The graphics I have drawn are shown below.





Command Window

```
Hesapladigimiz p=0,5 deki mean = 0.996300
Hesapladigimiz p=0,5 deki variance = 1.966086

Hesapladigimiz p=0,5 deki mean = 2.357400
Hesapladigimiz p=0,3 deki variance = 7.667865
fx >>
```

C-) Poisson distribution with λ=6 and Poisson distribution with λ=4.5

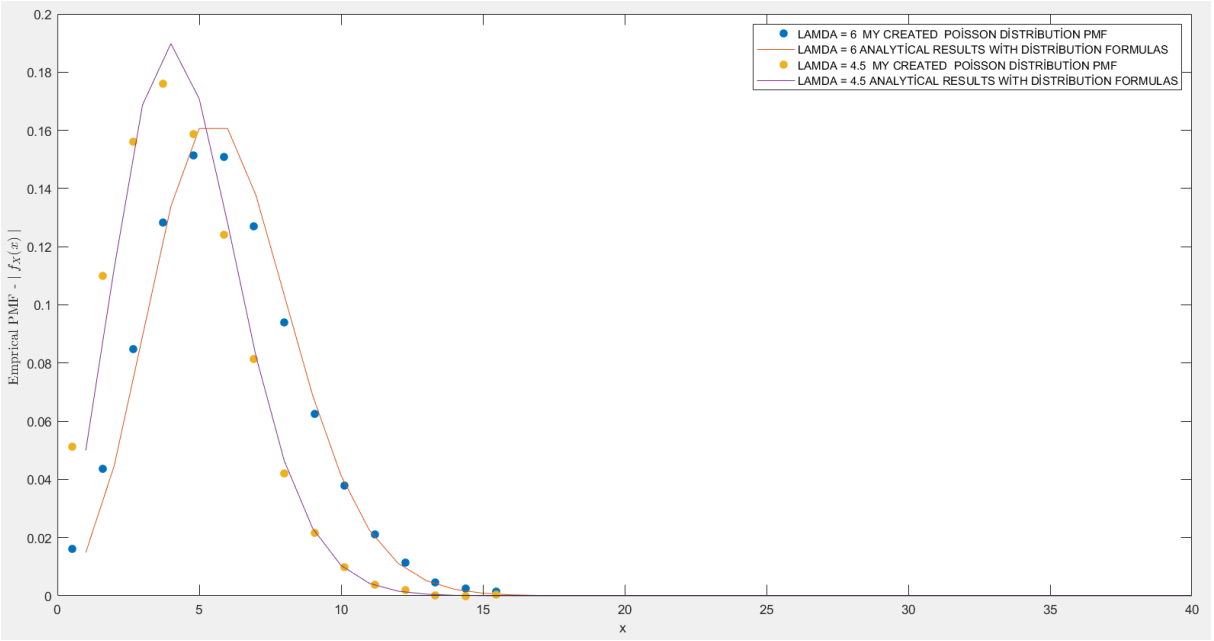
The Poisson distribution is the discrete probability distribution of the number of events occurring in a given time period, given the average number of times the event occurs over that time period.

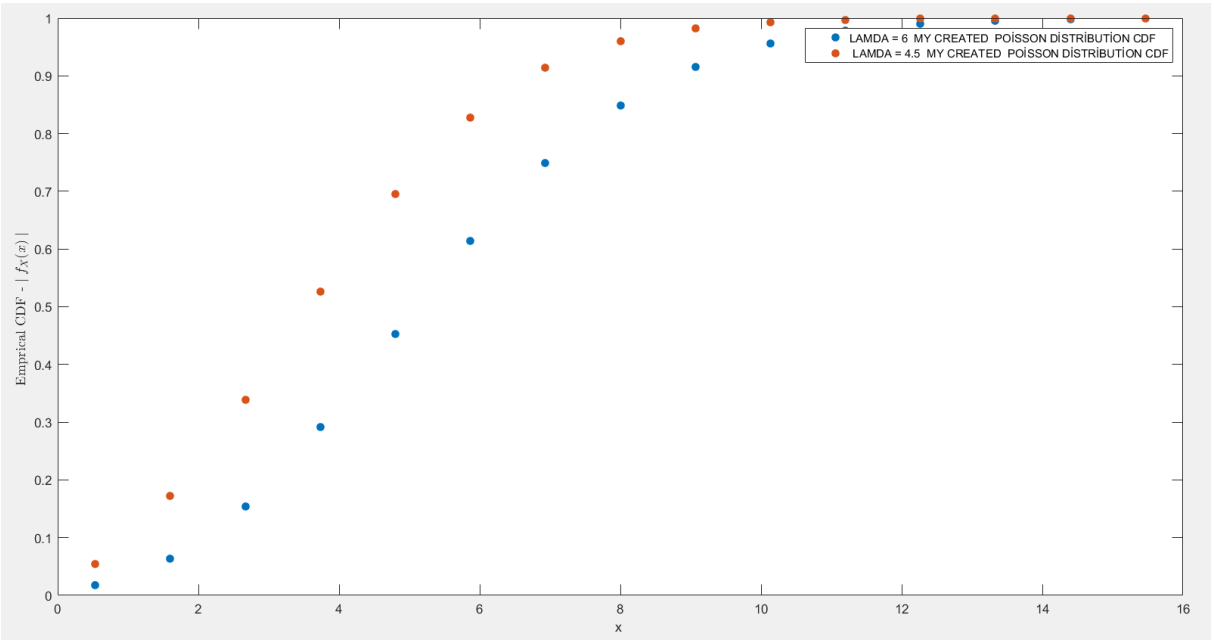
I got the sample numbers and lamda values from the user. I sent these numbers to the poissonGenerate(lambda,samplesize) function, which returned us our geometric distribution samples. First we generated a random number between 0 and 1. Then we took the negative logarithm of this number and added it to the total variable. We increased our counter one by one until the total variable was larger than lambda. We put this total number in an array. This series was our poisson distribution.

Then I created the histogram of this series using the array we created. Finally, we calculated Probability mass function and Cumulative distribution function from this histogram and plotted them.

I also drew the relevant analytical results using the poisson distribution formula. Finally, using myMeanFunction (incomingSamples) and myVarianceFunction (incomingSamples) functions average and variance of poisson distributions that we create I calculated the values and printed them on the screen.

The graphics I have drawn are shown below.





Command Window

```
Hesapladigimiz p=0,5 deki mean = 0.996300
Hesapladigimiz p=0,5 deki variance = 1.966086

Hesapladigimiz p=0,5 deki mean = 2.357400
Hesapladigimiz p=0,3 deki variance = 7.667865
fx >>
```

D-) Binomial distribution with n=40, p=6/40 and Binomial distribution with n=50, p=4.5/50

Binomial Distribution Probability mass function

$$B(p,n) = P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

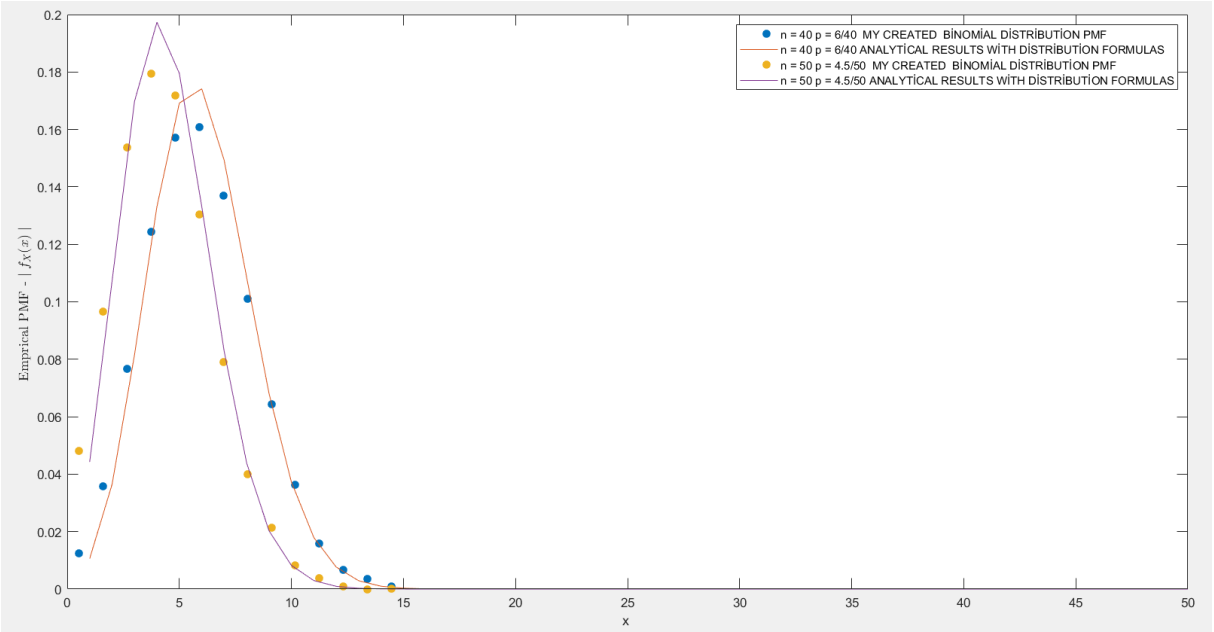
As mentioned above, let's define lambda as follows: $\lambda = np \Rightarrow p = \frac{\lambda}{n}$

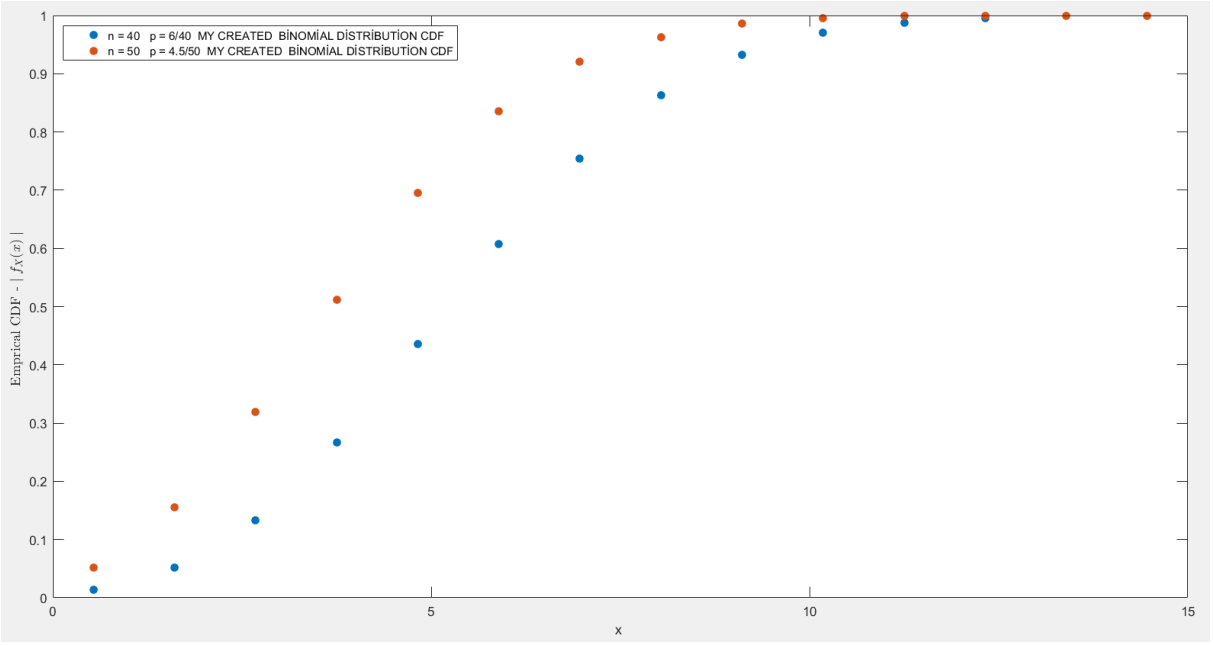
If we replace it in the formula: $\lim_{n\to\infty} P(X = k) = \lim_{n\to\infty} \frac{n!}{k!(n-k)!} \left(\frac{\lambda}{n}\right)^k \left(1 - \frac{\lambda}{n}\right)^{n-k}$

If we solve this limit: $P(\lambda, k) = \left(\frac{\lambda^k e^{-\lambda}}{k!}\right)$

This formula is the probability mass function formula of poisson distribution. Therefore, the results we find in option C and option D are similar.

The graphics I have drawn are shown below.





Command Window

```
Hesapladigimiz n=40 p=0,7 deki mean = 5.987300
Hesapladigimiz n=40 p=0,7 deki variance = 5.185539

Hesapladigimiz n=50 p=0,5 deki mean = 4.558800
Hesapladigimiz n=50 p=0,5 deki variance = 4.144543
fx >>
```

QUESTION 2

A-) Exponential distribution with λ=1.2 and Exponential distribution λ=2.1

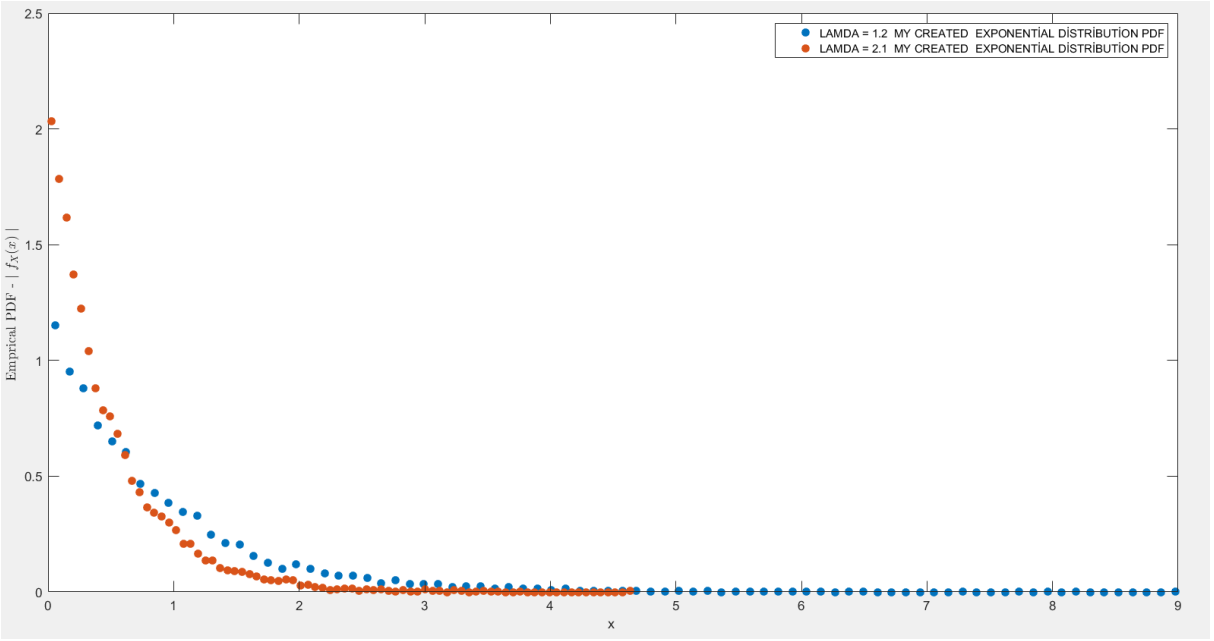
The exponential distribution is one of the widely used continuous distributions. It is often used to model the time elapsed between events.

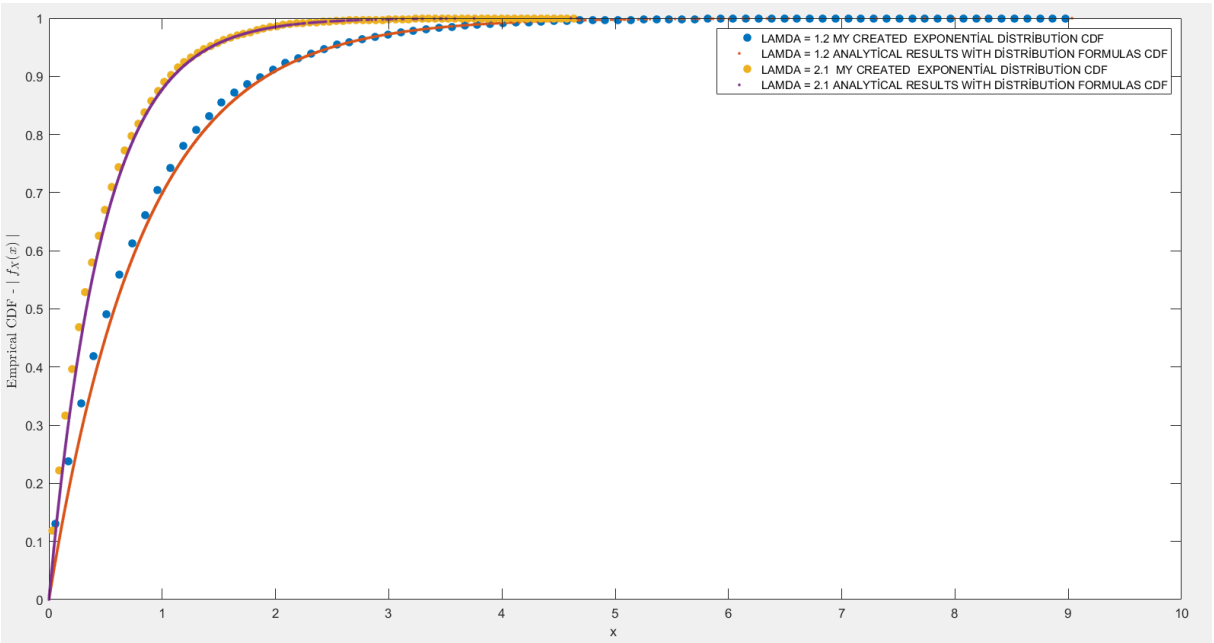
I got the sample numbers and lamda values from the user. I sent these numbers to the exponentialGenerate (lambda,samplesize) function, which returned us our exponential distribution samples. First we generated a random number between 0 and 1. Then we took the negative logarithm of this number and divided it into lambda. We put this number in an array. This series was our exponential distribution. To create this distribution, we used the reverse CDF technique.

Then I created the histogram of this series using the array we created. Finally, we calculated Probability density function and Cumulative distribution function from this histogram and plotted them.

I also drew the relevant analytical results using the exponential distribution formula. Finally, using myMeanFunction (incomingSamples) and myVarianceFunction (incomingSamples) functions average and variance of exponential distributions that we create I calculated the values and printed them on the screen.

The graphics I have drawn are shown below.





Command Window

```
Hesapladigimiz LAMDA = 1.2 daki mean = 0.835684
Hesapladigimiz LAMDA = 1.2 daki variance = 0.726447

Hesapladigimiz LAMDA = 2.1 daki mean = 0.473302
Hesapladigimiz LAMDA = 2.1 daki variance = 0.230472
fx >>
```

B-) Normal distribution with μ=3.1, σ^2=5 and Normal distribution with μ=1, σ^2=2

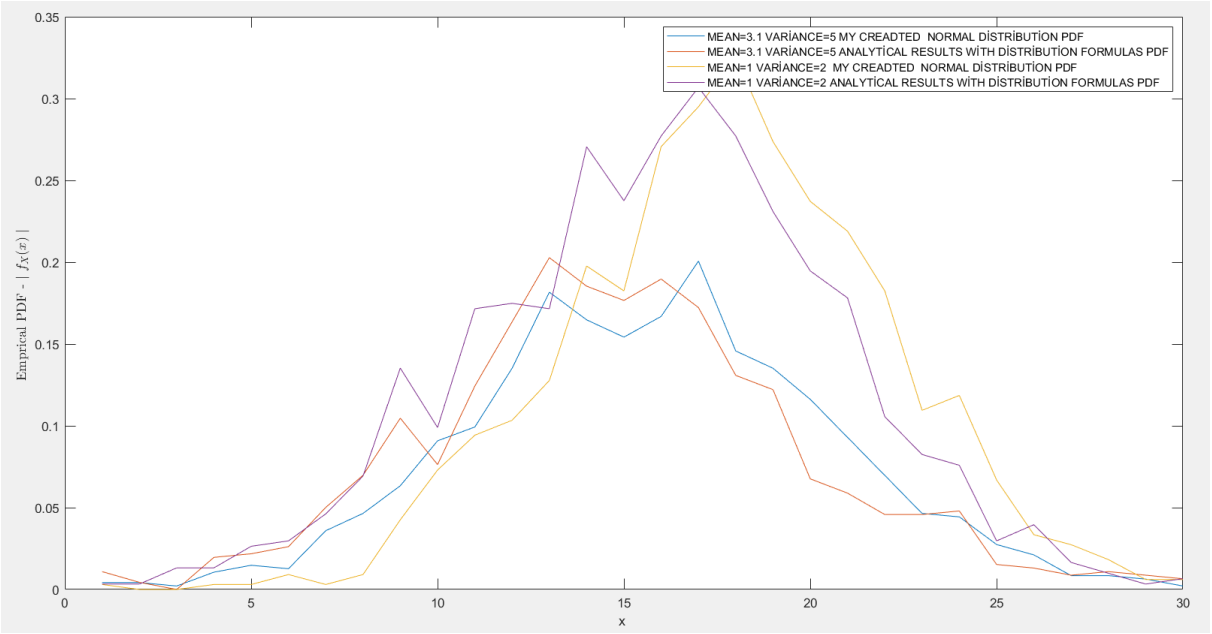
The normal distribution is the most important probability distribution in statistics because it fits many natural phenomena.

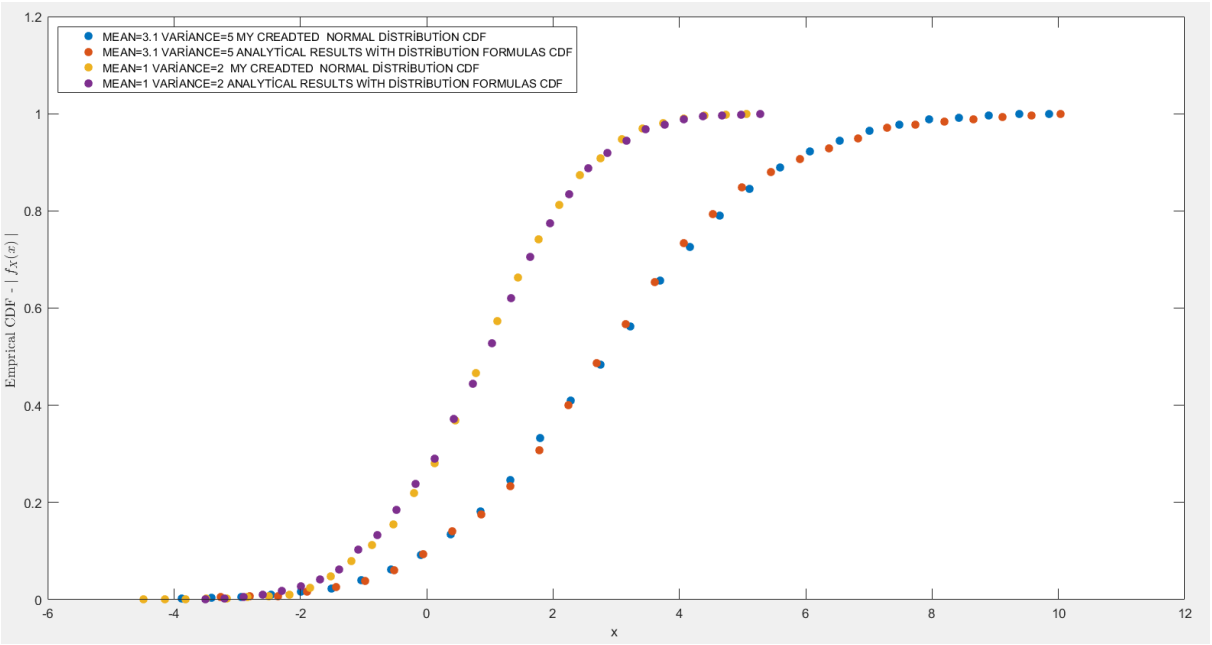
I got the mean and variance values from the user. I sent these numbers to the normalGenerate(mean,variance) function, which returned us our normal distribution samples.

Then I created the histogram of this series using the array we created. Finally, we calculated Probability density function and Cumulative distribution function from this histogram and plotted them.

I also drew the relevant analytical results using the normal distribution formula. Finally, using myMeanFunction (incomingSamples) and myVarianceFunction (incomingSamples) functions average and variance of normal distributions that we create I calculated the values and printed them on the screen.

The graphics I have drawn are shown below.





Command Window

```
Hesapladigimiz LAMDA = 1.2 daki mean = 3.081989
Hesapladigimiz LAMDA = 1.2 daki variance = 5.083653

Hesapladigimiz LAMDA = 2.1 daki mean = 1.029547
Hesapladigimiz LAMDA = 2.1 daki variance = 1.933779
fx >>
```

QUESTION 3

A-)

Entities: Two service lines and Three callers

Attributes of a Lines: busy or not busy

Attributes of a Callers: impatient

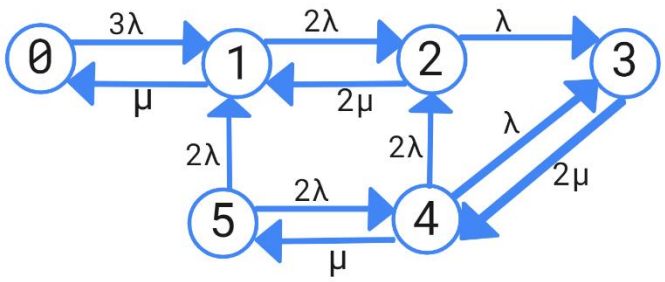
Attributes of a Activities: Answering calls and blocked calls

Attributes of a Events: Each caller makes calls that are exponentially distributed in length

Attributes of a State Variables: Number of callers successful,Number of callers impatient

B-)

0 no calls in progress,3 callers idle
1 1 call in progress,2 callers idle
2 2 calls in progress,1 caller idle
3 2 calls in progress,1 caller impatient
4 1 call in progress,1 caller impatient
5 0 calls in progress,1 caller impatient



C-)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This function generates samples of exponential distribution with rate
% lambda, whose PDF is given in the question
function [samples] = genexprv(lambda, noSamples)
    %
    % Write your own code here
    %
     [samples]=-log(rand)/lambda;
end
```

D-)

I have defined the number of customers looking for impatience as the new state variable.

The graphics I have drawn are shown below new state variable.