Batch: A1 Roll No.: 16010123012

Experiment No. 3

Title: Modeling real world data using suitable probability distributions

Aim: Exploring the Binomial, Poisson, and Normal Distributions in R

Course Outcome:

CO2

Books/ Journals/ Websites referred:

1. The Comprehensive R Archive Network
2. Posit

Resources used:
https://www.rdocumentation.org/
https://www.w3schools.com/r/
https://www.geeksforgeeks.org/r-programming-language-introduction/

Theory:

Binomial Distribution

Definition

The binomial distribution is a discrete probability distribution that models the number of successes in a fixed number of independent trials, each with the same probability of success. It is commonly used in scenarios where outcomes are binary (success/failure, yes/no, etc.).

The probability mass function (PMF) for the binomial distribution is given by:

$$f(x) = \mathrm{P}[X = x] = \binom{n}{x} p^x (1-p)^{n-x}$$

Where:

- **n**: Number of trials

- **x**: Number of successes
- **p**: Probability of success in a single trial

dbinom

The function dbinom returns the value of the probability mass function (pmf) of the binomial distribution given a certain random variable x, number of trials (size) and probability of success on each trial (prob).

The syntax for using dbinom is as follows:

dbinom(x, size, prob)

Put simply, dbinom finds the probability of getting a certain number of successes (x) in a certain number of trials (size) where the probability of success on each trial is fixed (prob).

The following examples illustrate how to solve some probability questions using dbinom.

Example 1: Alice makes 60% of her free-throw attempts. If she shoots 12 free throws, what is the probability that she makes exactly 10?

```
> #find the probability of 10 successes during 12 trials where the probability of
> #success on each trial is 0.6
> dbinom(x=10, size=12, prob=.6)
[1] 0.06385228
```

The probability that he makes exactly 10 shots is 0.0639.

Example 2: Bob flips a fair coin 20 times. What is the probability that the coin lands on heads exactly 7 times?

```
> #find the probability of 7 successes during 20 trials where the probability of
> #success on each trial is 0.5
> dbinom(x=7, size=20, prob=.5)
[1] 0.07392883
```

The probability that the coin lands on heads exactly 7 times is 0.0739.

pbinom

The function pbinom returns the value of the cumulative density function (cdf) of the binomial distribution given a certain random variable q, number of trials (size) and probability of success on each trial (prob). The syntax for using pbinom is as follows:

pbinom(q, size, prob)

The following examples illustrate how to solve some probability questions using pbinom.

Example 1: Suppose Kishor scores a strike on 30% of his attempts when he bowls. If he bowls 10 times, what is the probability that he scores 4 or fewer strikes?

```
> #find the probability of 4 or fewer successes during 10 trials where the
> #probability of success on each trial is 0.3
> pbinom(4, size=10, prob=.3)
[1] 0.8497317
```

The probability that he scores 4 or fewer strikes is 0.8497.

Put simply, pbinom returns the area to the left of a given value q in the binomial distribution. If you're interested in the area to the right of a given value q, you can simply add the argument lower.tail = FALSE

pbinom(q, size, prob, lower.tail = FALSE)

Example 2: Ashok flips a fair coin 5 times. What is the probability that the coin lands on heads more than 2 times?

```
> #find the probability of more than 2 successes during 5 trials where the
> #probability of success on each trial is 0.5
> pbinom(2, size=5, prob=.5, lower.tail=FALSE)
[1] 0.5
```

The probability that the coin lands on heads more than 2 times is 0.5.

qbinom

The function qbinom returns the quantile (or the smallest integer) for which the cumulative density function (cdf) of the binomial distribution reaches or exceeds a given probability p. It effectively works as the inverse of pbinom, helping to find the threshold number of successes q for a specified cumulative probability p, given the number of trials (size) and the probability of success on each trial (prob).

The syntax for using qbinom is as follows:

qbinom(q, size, prob)

The following code illustrates a few examples of qbinom in action:

Example 1: Suppose Kishor scores a strike on 30% of his attempts when he bowls. If he bowls 10 times, what is the maximum number of strikes he can score given the cumulative probability is 0.8497?

```
> qbinom(0.8497, size = 10, prob = 0.3)
[1] 4
```

Example 2: Ashok flips a fair coin 5 times. What is the minimum number of heads he can observe to ensure the cumulative probability is greater than 0.5?

```
> qbinom(0.5, size = 5, prob = 0.5)
[1] 2
```

You can use qbinom to find out the percentile of the binomial distribution.

The following code illustrates a few examples:

#find the 10th percentile of a binomial distribution with 10 trials and prob

#of success on each trial = 0.4

```
> qbinom(.10, size=10, prob=.4)
[1] 2
```

#find the 40th percentile of a binomial distribution with 30 trials and prob

#of success on each trial = 0.25

```
> qbinom(.40, size=30, prob=.25)
[1] 7
```

rbinom

The function rbinom generates a vector of binomial distributed random variables given a vector length n, number of trials (size) and probability of success on each trial (prob).

The syntax for using rbinom is as follows:

rbinom(n, size, prob)

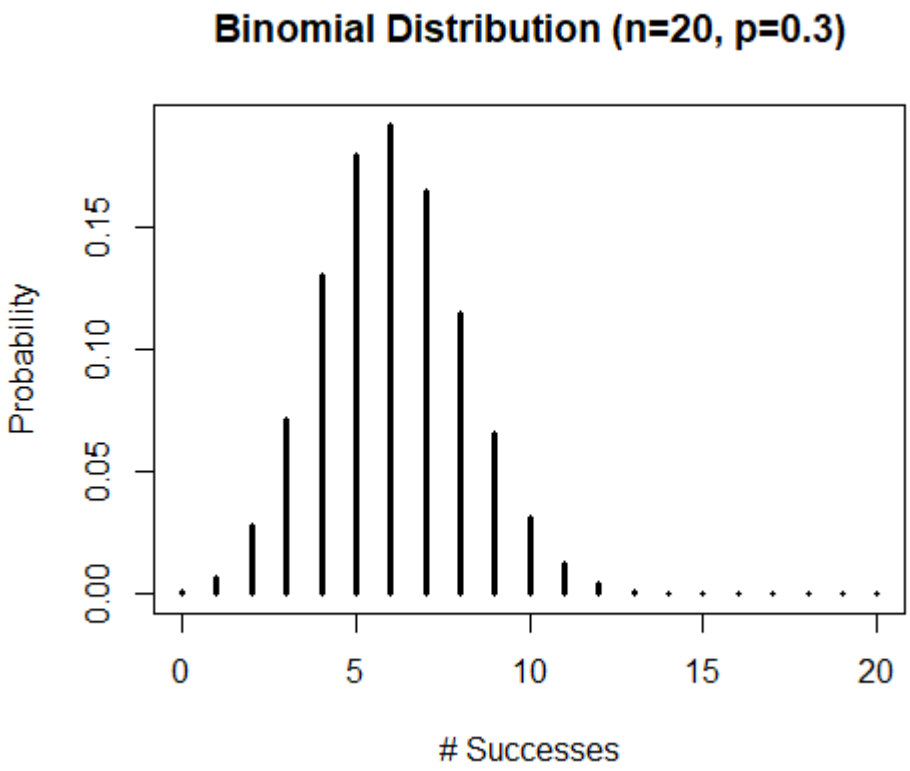
The following code illustrates a few examples of rbinom in action:

```
> #generate a vector that shows the number of successes of 10 binomial experiments with
> #100 trials where the probability of success on each trial is 0.3.
> results <- rbinom(10, size=100, prob=.3)
>
> print(results)
 [1] 37 35 33 34 44 26 28 27 27 20
```

Visualization

```
> success <- 0:20

  plot(success,dbinom(success,size=20,prob=.3),
       type='h',
       main='Binomial Distribution (n=20, p=0.3)',
       ylab='Probability',
       xlab ='# Successes',
       lwd=3)
```



Poisson Distribution

Definition

The Poisson distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time or space, provided these events occur with a known constant rate and independently of the time since the last event.

The probability mass function (PMF) for the Poisson distribution is given by:

$$f(x, \lambda) = P(X = x) = \frac{\lambda^x \cdot e^{-\lambda}}{x!}$$

Where:

- **x**: Number of events
- **λ (lambda)**: Average number of events per interval (rate parameter)
- **e**: Euler's number ≈2.718

dpois

The dpois function finds the probability that a certain number of successes occur based on an average rate of success, using the following syntax:

dpois(x, lambda)

where:

x: number of successes

lambda: average rate of success

Example : It is known that a certain website makes 10 sales per hour. In a given hour, what is the probability that the site makes exactly 8 sales?

```
> dpois(x=8, lambda=10)
[1] 0.112599
```

The probability that the site makes exactly 8 sales is 0.112599

ppois

The ppois function finds the probability that a certain number of successes or less occur based on an average rate of success, using the following syntax:

ppois(q, lambda)

where:

q: number of successes

lambda: average rate of success

Example : It is known that a certain website makes 10 sales per hour. In a given hour, what is the probability that the site makes 8 sales or less?

```
> ppois(q=8, lambda=10)
[1] 0.3328197
```

qpois

The qpois function finds the number of successes that corresponds to a certain percentile based on an average rate of success, using the following syntax:

qpois(p, lambda)

where:

p: percentile

lambda: average rate of success

Example : It is known that a certain website makes 10 sales per hour. How many sales would the site need to make to be at the 90th percentile for sales in an hour?

```
> qpois(p=.90, lambda=10)
[1] 14
```

rpois

The rpois function generates a list of random variables that follow a Poisson distribution with a certain average rate of success, using the following syntax:

rpois(n, lambda)

where:

n: number of random variables to generate

lambda: average rate of success

Here's an example of when you might use this function in practice:

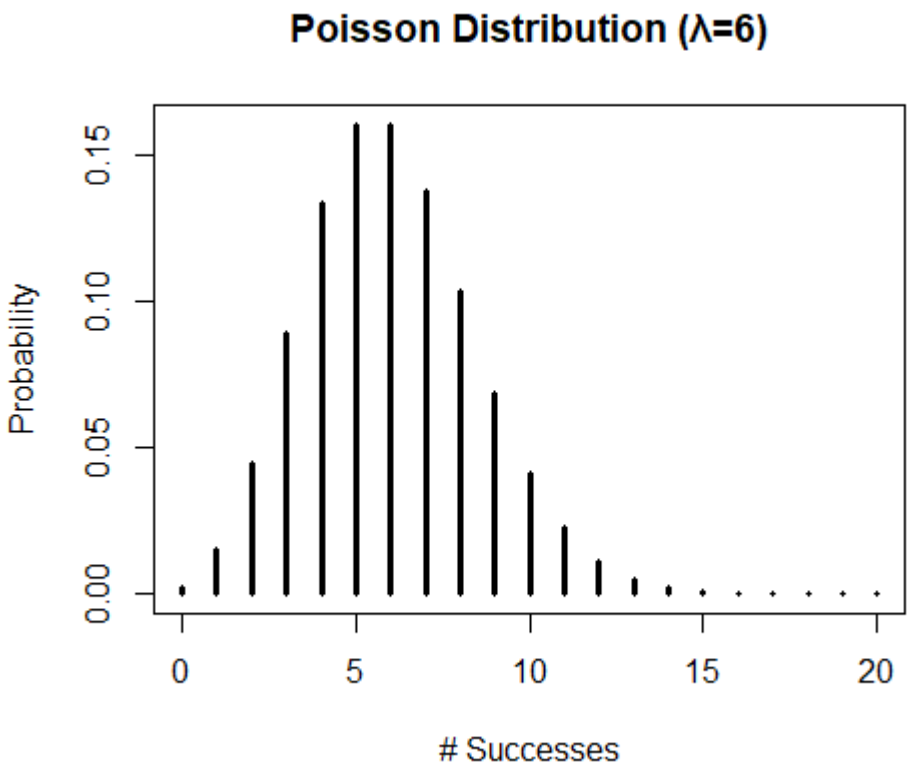
Generate a list of 15 sales per hour that follow a Poisson distribution with an average rate of sales being equal to 10.

```
> rpois(n=15, lambda=10)
 [1]  8  7 10  6  7 10 12 14  8 10  8 14  6 10 13
```

Visualization

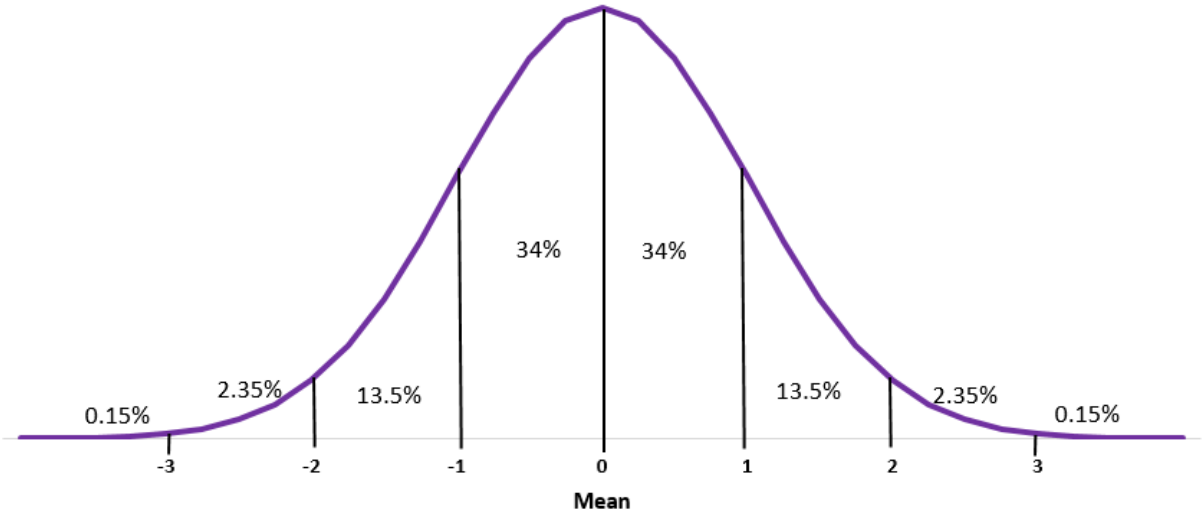
```
> # Set parameters
  lambda <- 6  # Mean of the Poisson distribution
  success <- 0:20

  # Plot the Poisson distribution
  plot(success, dpois(success, lambda),
       type='h',
       main='Poisson Distribution (λ=6)',
       ylab='Probability',
       xlab='# Successes',
       lwd=3)
```



Normal distribution

The normal distribution is the most common probability distribution in statistics.



Normal distributions have the following features:

- Bell shape
- Symmetrical
- Mean and median are equal; both are located at the center of the distribution
- About 68% of data falls within one standard deviation of the mean
- About 95% of data falls within two standard deviations of the mean
- About 99.7% of data falls within three standard deviations of the mean

dnorm

The function dnorm returns the value of the probability density function (pdf) of the normal distribution given a certain random variable x, a population mean μ and population standard deviation σ. The syntax for using dnorm is as follows:

dnorm(x, mean, sd)

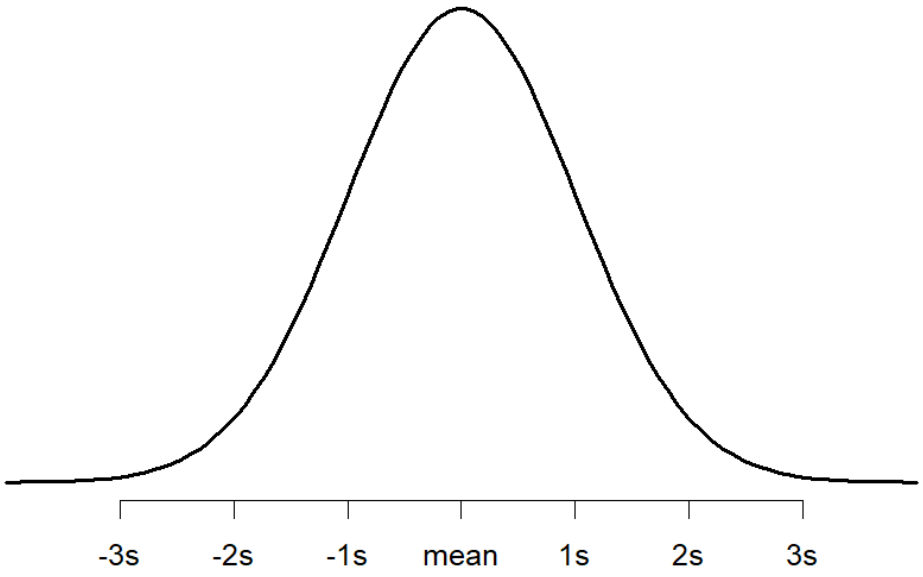
```
> #find the value of the standard normal distribution pdf at x=0
> dnorm(x=0, mean=0, sd=1)
[1] 0.3989423
> #by default, R uses mean=0 and sd=1
> dnorm(x=0)
[1] 0.3989423
> #find the value of the normal distribution pdf at x=10 with mean=20 and sd=5
> dnorm(x=10, mean=20, sd=5)
[1] 0.01079819
```

Visualization

```
> #Create a sequence of 100 equally spaced numbers between -4 and 4
  x <- seq(-4, 4, length=100)

  #create a vector of values that shows the height of the probability distribution
  #for each value in x
  y <- dnorm(x)

  #plot x and y as a scatterplot with connected lines (type = "l") and add
  #an x-axis with custom labels
  plot(x,y, type = "l", lwd = 2, axes = FALSE, xlab = "", ylab = "")
  axis(1, at = -3:3, labels = c("-3s", "-2s", "-1s", "mean", "1s", "2s", "3s"))
```



pnorm

The function pnorm returns the value of the cumulative density function (cdf) of the normal distribution given a certain random variable q, a population mean μ and population standard deviation σ. The syntax for using pnorm is as follows:

pnorm(q, mean, sd)

Put simply, pnorm returns the area to the left of a given value x in the normal distribution. If you're interested in the area to the right of a given value q, you can simply add the argument lower.tail = FALSE

pnorm(q, mean, sd, lower.tail = FALSE)

Example 1: Suppose the height of males at a certain school is normally distributed with a mean of μ=70 inches and a standard deviation of σ = 2 inches. Approximately what percentage of males at this school are taller than 74 inches?

```
> #find percentage of males that are taller than 74 inches in a population with
> #mean = 70 and sd = 2
> pnorm(74, mean=70, sd=2, lower.tail=FALSE)
[1] 0.02275013
```

Example 2: Suppose the weight of a certain species of otters is normally distributed with a mean of μ=30 lbs and a standard deviation of σ = 5 lbs. Approximately what percentage of this species of otters weigh less than 22 lbs?

```
> #find percentage of otters that weight less than 22 lbs in a population with
> #mean = 30 and sd = 5
> pnorm(22, mean=30, sd=5)
[1] 0.05479929
```

Example 3: Suppose the height of plants in a certain region is normally distributed with a mean of μ=13 inches and a standard deviation of σ = 2 inches. Approximately what percentage of plants in this region are between 10 and 14 inches tall?

```
> #find percentage of plants that are less than 14 inches tall, then subtract the
> #percentage of plants that are less than 10 inches tall, based on a population
> #with mean = 13 and sd = 2
> pnorm(14, mean=13, sd=2) - pnorm(10, mean=13, sd=2)
[1] 0.6246553
```

qnorm

The function qnorm returns the value of the inverse cumulative density function (cdf) of the normal distribution given a certain random variable p, a population mean μ and population standard deviation σ. The syntax for using qnorm is as follows:

qnorm(p, mean, sd)

Put simply, you can use qnorm to find out what the Z-score is of the p-th quantile of the normal distribution.

```
> #find the z-score of the 99th quantile of the standard normal distribution
> qnorm(.99, mean=0, sd=1)
[1] 2.326348
```

```
> #by default, R uses mean=0 and sd=1
> qnorm(.99)
[1] 2.326348
> #find the Z-score of the 95th quantile of the standard normal distribution
> qnorm(.95)
[1] 1.644854
> #find the Z-score of the 10th quantile of the standard normal distribution
> qnorm(.10)
[1] -1.281552
```

rnorm

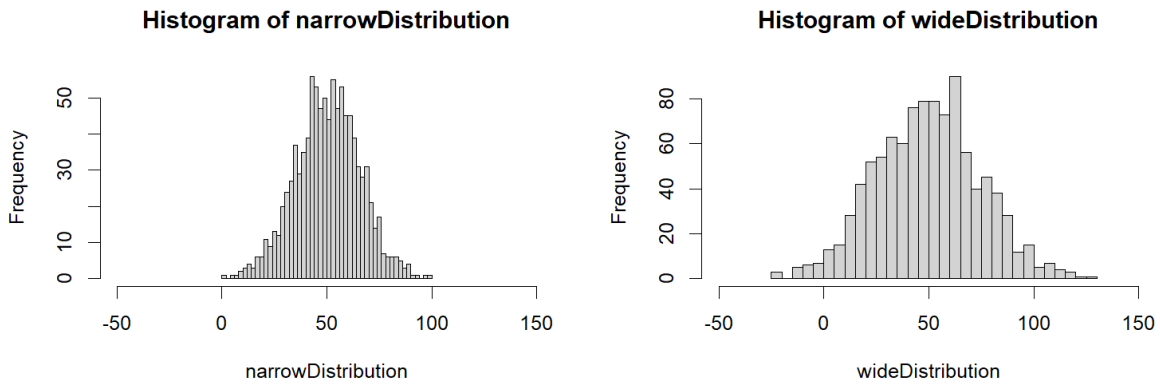
The function **rnorm** generates a vector of normally distributed random variables given a vector length *n*, a population mean μ and population standard deviation σ. The syntax for using rnorm is as follows:

rnorm(n, mean, sd)

```
> #generate a vector of 5 normally distributed random variables with mean=10 and sd=2
> five <- rnorm(5, mean = 10, sd = 2)
> five
[1]  9.105876  6.522804 10.357730 13.794931  5.456149
```

Visualization

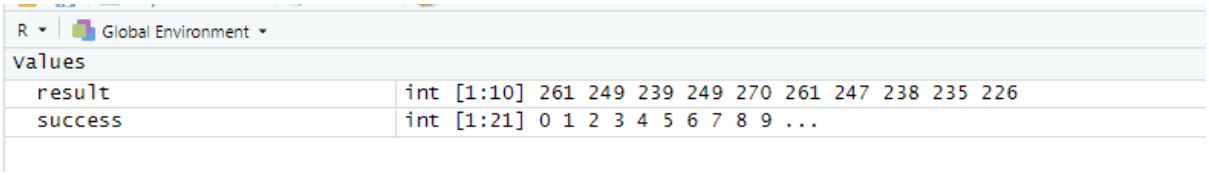
```
> #generate a vector of 1000 normally distributed random variables with mean=50 and sd=5
> narrowDistribution <- rnorm(1000, mean = 50, sd = 15)
> #generate a vector of 1000 normally distributed random variables with mean=50 and sd=25
> wideDistribution <- rnorm(1000, mean = 50, sd = 25)
>
> #generate two histograms to view these two distributions side by side, specify
> #50 bars in histogram and x-axis limits of -50 to 150
> par(mfrow=c(1, 2)) #one row, two columns
> hist(narrowDistribution, breaks=50, xlim=c(-50, 150))
> hist(wideDistribution, breaks=50, xlim=c(-50, 150))
```

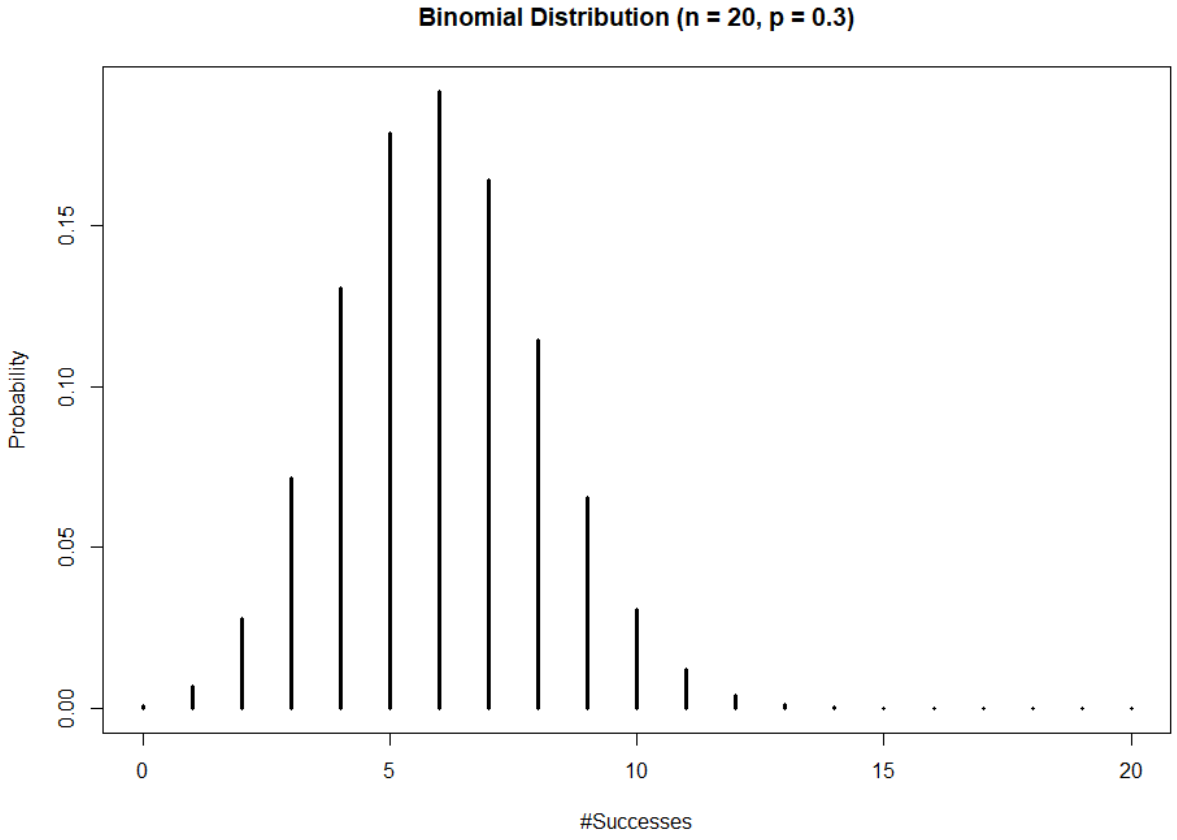


Notice how the wide distribution is much more spread out compared to the narrow distribution. This is because we specified the standard deviation in the wide distribution to be 25 compared to just 15 in the narrow distribution.

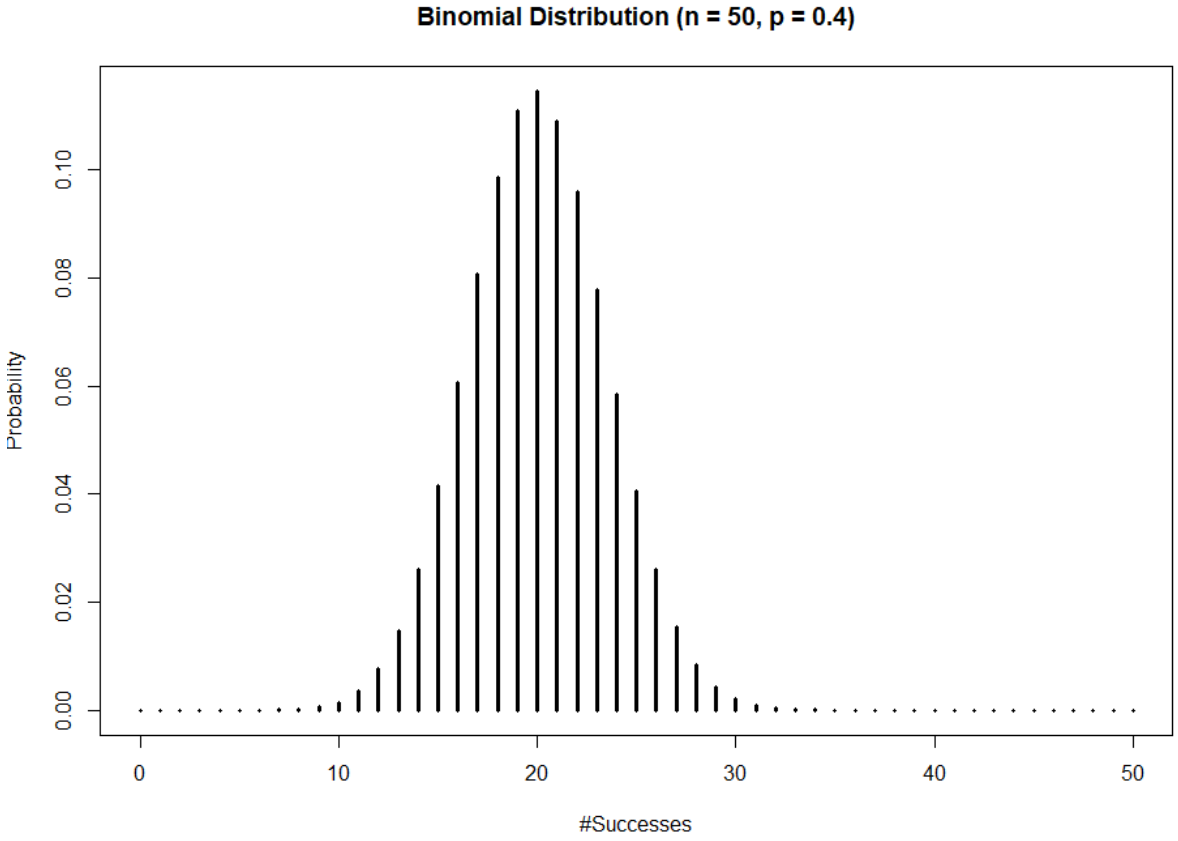
Students have to experiment with different values of parameters - p, n, lambda, mu, sigma - to understand how the binomial, poisson and normal distributions behave.

Lab Work:



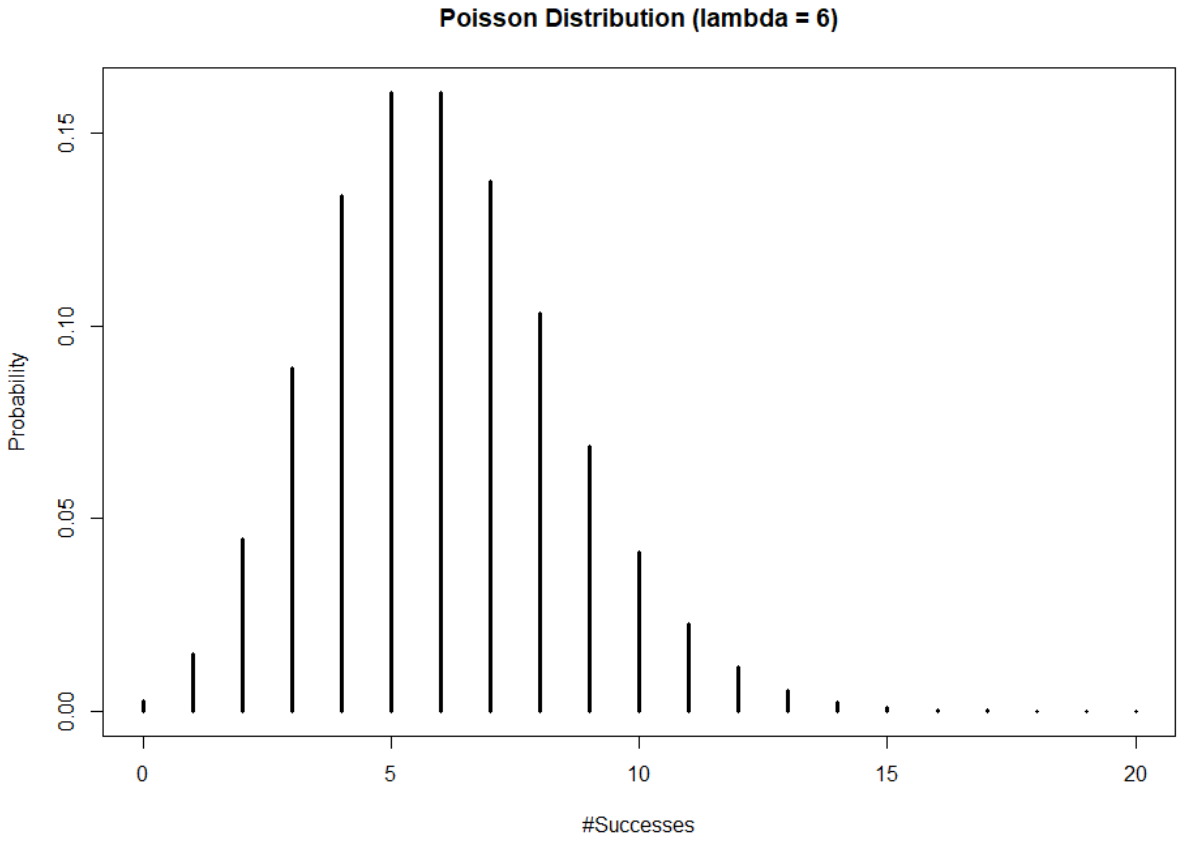


```
> dbinom(x = 10, size = 12, prob = 0.6)
[1] 0.06385228
> dbinom(x = 7, size = 20, prob = 0.5)
[1] 0.07392883
> dbinom(x = 7, size = 100, prob = 0.5)
[1] 1.262774e-20
>
> pbinom(4, size = 10, prob = 0.3)
[1] 0.8497317
> pbinom(4, size = 50, prob = 0.3)
[1] 0.0001719274
> pbinom(4, size = 5, prob = 0.5, lower.tail = FALSE)
[1] 0.03125
> pbinom(2, size = 5, prob = 0.5, lower.tail = FALSE)
[1] 0.5
>
> qbinom(0.8479, size = 10, prob = 0.3)
[1] 4
> qbinom(0.5, size = 100, prob = 0.5)
[1] 50
> qbinom(0.5, size = 5, prob = 0.5)
[1] 2
> qbinom(0.1, size = 20, prob = 0.4)
[1] 5
> qbinom(0.4, size = 30, prob = 0.25)
[1] 7
>
> result <- rbinom(10, size = 100, prob = 0.3)
> print(result)
 [1] 38 34 26 31 32 32 31 34 29 43
> result <- rbinom(10, size = 500, prob = 0.5)
> print(result)
 [1] 261 249 239 249 270 261 247 238 235 226
>
> success <- 0:20
> plot(success, dbinom(success, size = 20, prob = 0.3),
+      type = 'h',
+      main = 'Binomial Distribution (n = 20, p = 0.3)',
+      ylab = 'Probability',
+      xlab = '#Successes',
+      lwd = 3)
> success <- 0:50
> plot(success, dbinom(success, size = 50, prob = 0.4),
+      type = 'h',
+      main = 'Binomial Distribution (n = 50, p = 0.4)',
+      ylab = 'Probability',
+      xlab = '#Successes',
+      lwd = 3)
>
```




```
> dnorm(x = 0, mean = 0, sd = 1)
[1] 0.3989423
> dnorm(x = 0)
[1] 0.3989423
> dnorm(x = 10, mean = 20, sd = 5)
[1] 0.01079819
> dnorm(x = 100, mean = 400, sd = 15)
[1] 3.680632e-89
```

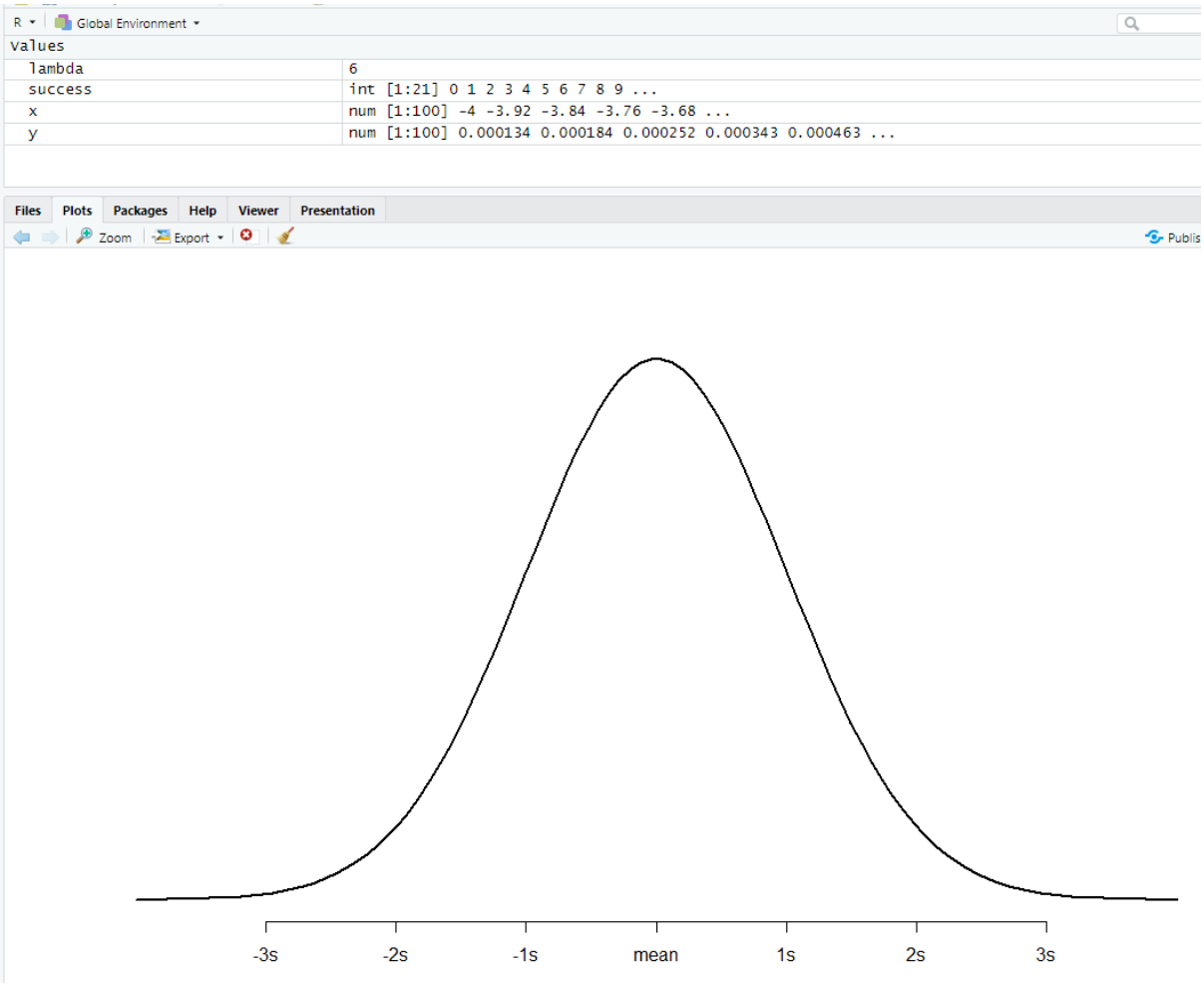
```
> dpois(x = 8, lambda = 10)
[1] 0.112599
> dpois(x = 5, lambda = 100)
[1] 3.100063e-36
> ppois(q = 8, lambda = 10)
[1] 0.3328197
> ppois(q = 10, lambda = 100)
[1] 1.137688e-30
> qpois(p = 0.9, lambda = 10)
[1] 14
>
> qpois(p = 0.5, lambda = 19)
[1] 19
>
> rpois(n = 15, lambda = 10)
 [1] 11 14 10  8 14 10 14 17 10 10 11 10 11  9  8
> rpois(n = 30, lambda = 1000)
 [1]  993 1048  959  982  935  983  942  977  989
[10] 1022  982  968 1050 1033  990  974  945 1032
[19]  997  981 1091  972  945 1036  975 1010 1024
[28]  999 1012 1052
>
> lambda <- 6
> success <- 0:20
> plot(success, dpois(success, lambda),
+      type = 'h',
+      main = 'Poisson Distribution (lambda = 6)',
+      ylab = 'Probability',
+      xlab = '#Successes',
+      lwd = 3)
> y <- dnorm(x)
> plot(x, y, type = "l", lwd = 2, axes = FALSE, xlab = "", ylab = "")
> axis(1, at = -3:3, label = c("-3s", "-2s", "-1s", "mean", "1s", "2s", "3s"))
> x <- seq(-4, 4, length = 100)
> y <- dnorm(x)
> plot(x, y, type = "l", lwd = 2, axes = FALSE, xlab = "", ylab = "")
> axis(1, at = -3:3, label = c("-3s", "-2s", "-1s", "mean", "1s", "2s", "3s"))
```



R | Global Environment

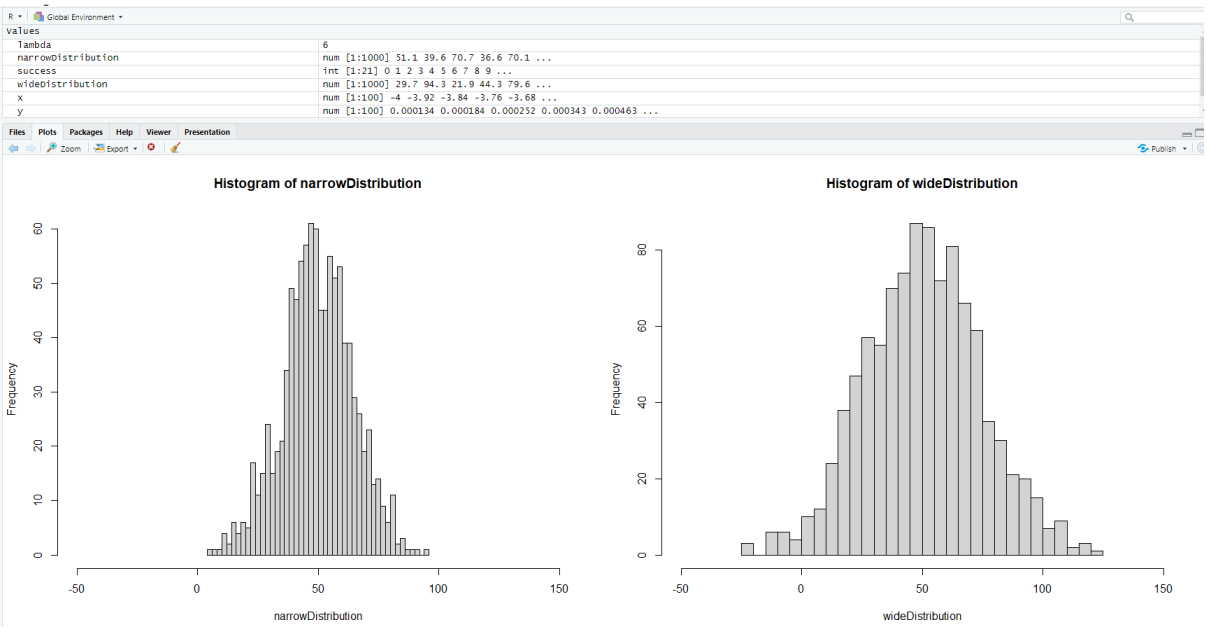
values

lambda	6
success	int [1:21] 0 1 2 3 4 5 6 7 8 9 ...




```
> qnorm(0.99, mean = 1, sd = 1)
[1] 3.326348
> qnorm(0.99)
[1] 2.326348
> qnorm(0.95)
[1] 1.644854
> qnorm(0.01)
[1] -2.326348
>
> z <- rnorm(5, mean = 10, sd = 2)
> z
[1] 11.370158  9.836304 10.192661  9.175306 12.691954
```

```
> narrowDistribution <- rnorm(1000, mean = 50, sd = 15)
> wideDistribution <- rnorm(1000, mean = 50, sd = 25)
>
> par(mfrow = c(1, 2))
> hist(narrowDistribution, breaks = 50, xlim = c(-50, 150))
> hist(wideDistribution, breaks = 50, xlim = c(-50, 150))
```



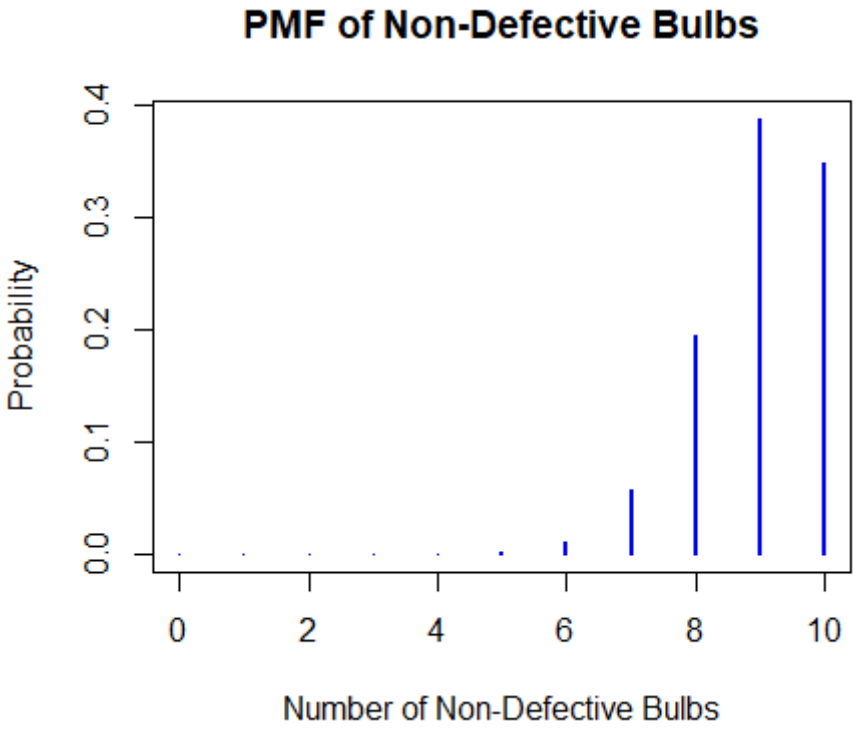
Conclusion: I successfully completed this experiment and gained a detailed understanding of probability distributions, including binomial, Poisson, and normal distributions, along with their implementation in R.

Post Lab questions

Solve the following using R:

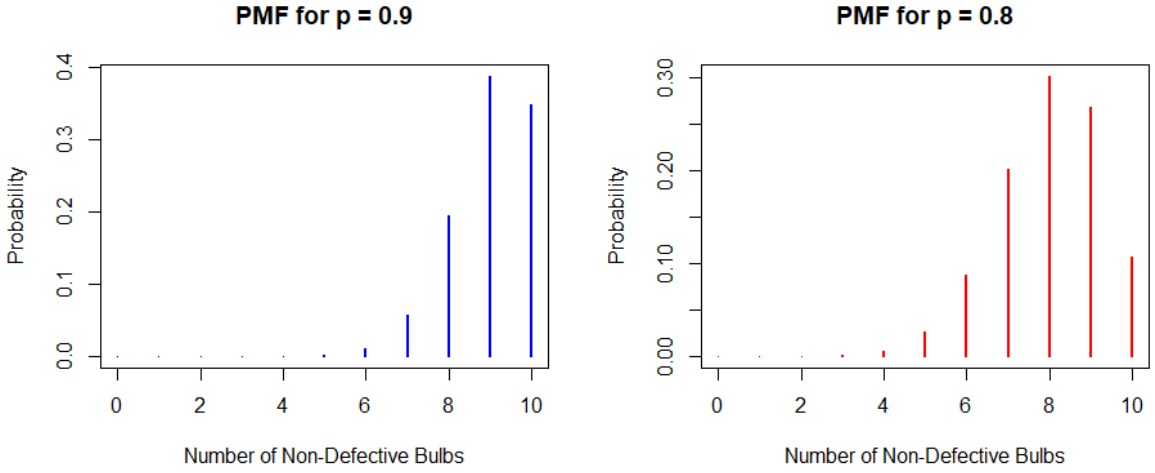
1A. Imagine a factory producing light bulbs, where each bulb has a 90% probability of being non-defective. If 10 bulbs are selected at random, use the binomial distribution to calculate the probability of finding exactly 8 non-defective bulbs. Visualize the distribution of defective bulbs in repeated samples

```
> k_values <- 0:10
> pmf_values <- dbinom(k_values, size = 10, prob = 0.9)
> plot(k_values, pmf_values, type = "h", lwd = 2, col = "blue", main = "PMF of Non-Defective Bulbs", xlab = "Number
of Non-Defective Bulbs",
+ ylab = "Probability")
```

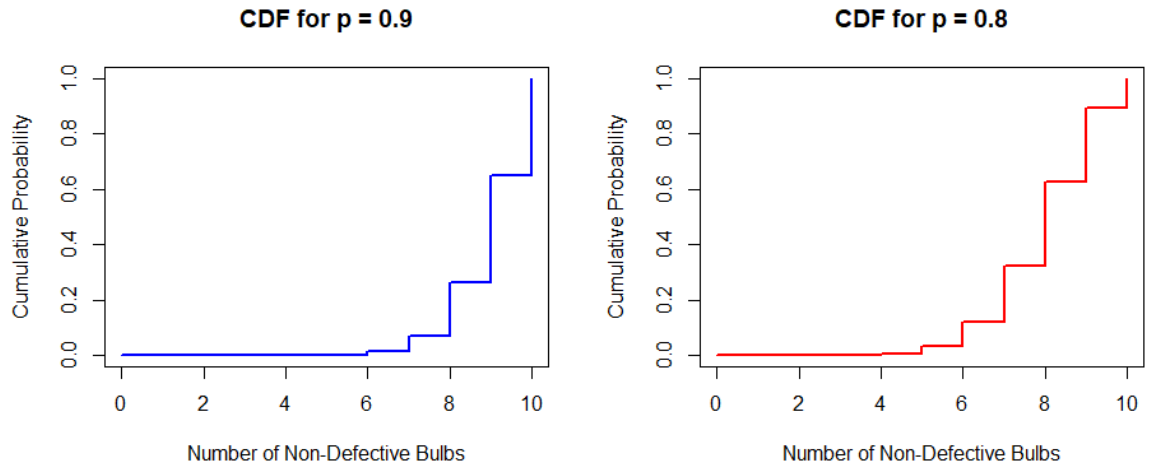


1B. If the probability of a bulb being non-defective decreases to 80%, how does this affect the shape of the PMF and CDF? Visualize it in R and interpret the changes.

```
> p_new <- 0.8
> pmfvalues90 <- dbinom(k_values, size = 10, prob = 0.9)
> pmfvalues80 <- dbinom(k_values, size = 10, prob = p_new)
> par(mfrow = c(1, 2))
> plot(k_values, pmfvalues90, type = "h", lwd = 2, col = "blue", main = "PMF for p = 0.9", xlab = "Number of Non-Def
ective Bulbs", ylab = "Probability")
> plot(k_values, pmfvalues80, type = "h", lwd = 2, col = "red", main = "PMF for p = 0.8", xlab = "Number of Non-Defe
ctive Bulbs", ylab = "Probability")
```

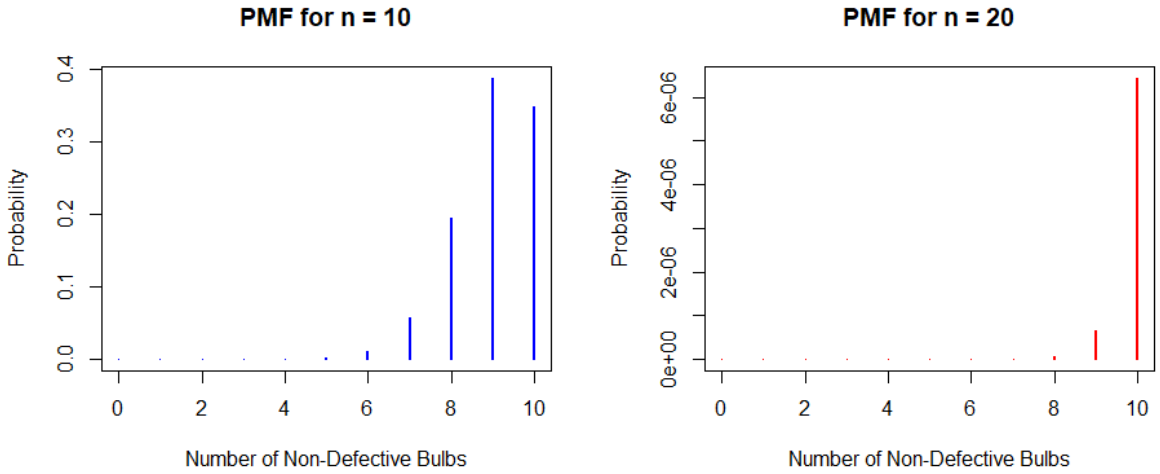


```
> cdfvalues90 <- pbinom(k_values, size = 10, prob = 0.9)
>
> cdfvalues90 <- pbinom(k_values, size = 10, prob = 0.9)
> cdfvalues80 <- pbinom(k_values, size = 10, prob = p_new)
> plot(k_values, cdfvalues90, type = "s", lwd = 2, col = "blue", main = "CDF for p = 0.9", xlab = "Number of Non-Def
ective Bulbs",ylab = "Cumulative Probability")
> plot(k_values, cdfvalues80, type = "s", lwd = 2, col = "red",main = "CDF for p = 0.8", xlab = "Number of Non-Defec
tive Bulbs", ylab = "Cumulative Probability")
```



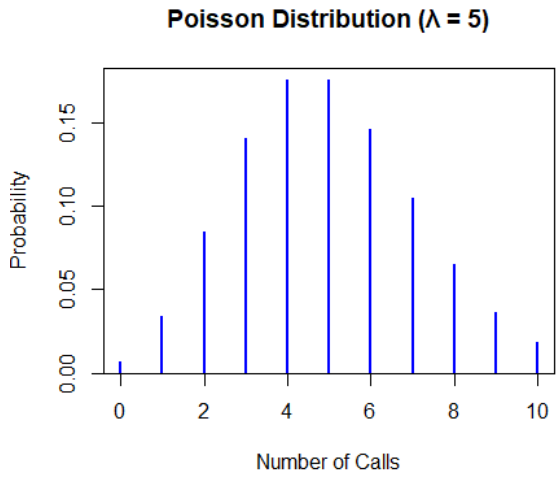
1C. In the context of quality control, how would increasing the sample size n to 20 impact the expected value and the spread of the distribution?

```
> n_new <- 20
> pmfvalues10 <- dbinom(k_values, size = 10, prob = 0.9)
> pmfvalues20 <- dbinom(k_values, size = n_new, prob = 0.9)
> par(mfrow = c(1, 2))
> plot(k_values, pmfvalues10, type = "h", lwd = 2, col = "blue", main = "PMF for n = 10", xlab = "Number of Non-Defe
ctive Bulbs", ylab = "Probability")
> plot(k_values, pmfvalues20, type = "h", lwd = 2, col = "red", main = "PMF for n = 20", xlab = "Number of Non-Defec
tive Bulbs", ylab = "Probability")
```



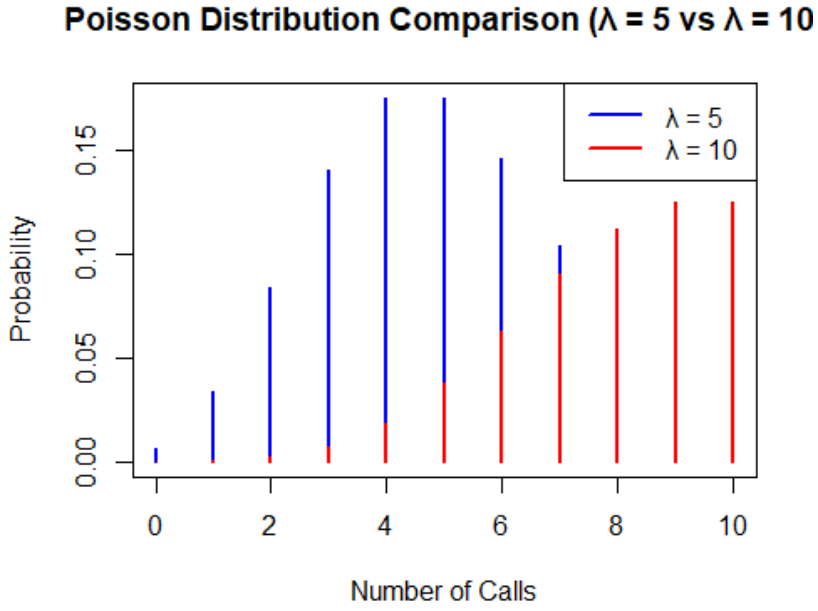
2A. Consider a call center that receives an average of 5 calls per hour. Use the Poisson distribution to determine the probability of receiving exactly 3 calls in an hour. Visualize how the number of calls varies over multiple hours.

```
> lambda=5
> k<-3
> dpois(k,lambda)
[1] 0.1403739
> k_values<- 0:10
> pmf_values <-dpois(k_values,lambda)
> plot(k_values, pmf_values, type = "h", lwd = 2, col = "blue",xlab = "Number of Calls", ylab = "Probability",main =
"Poisson Distribution (λ = 5)")
```



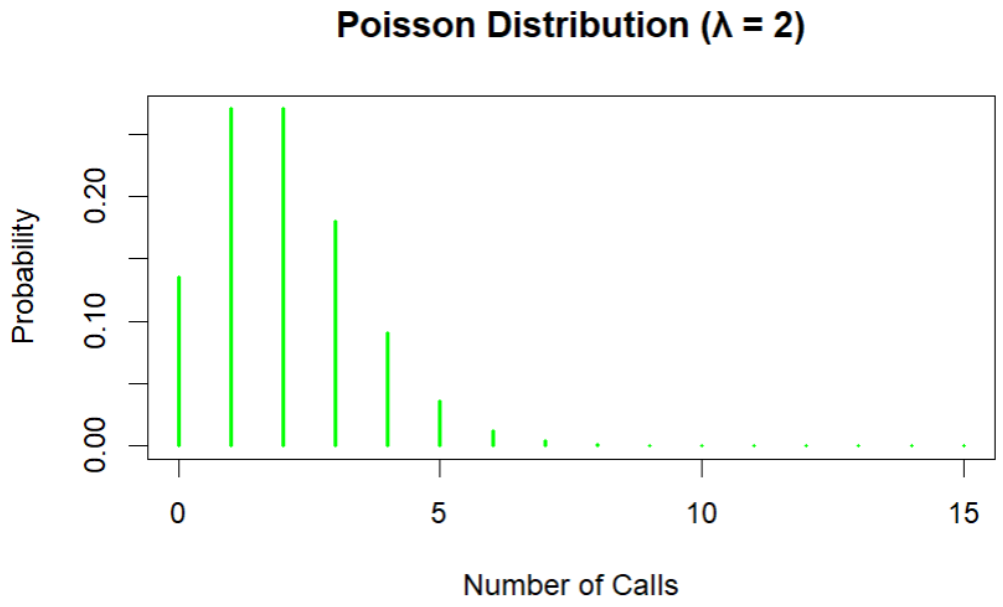
2B If the call rate increases to 10 calls per hour, how does the PMF and CDF change? Visualize it in R and explain.

```
> lambda_new <- 10
> dpois(k,lambda_new)
[1] 0.007566655
> pmf_values_new <- dpois(k_values, lambda_new)
> plot(k_values, pmf_values, type = "h", lwd = 2, col = "blue", xlab = "Number of Calls", ylab = "Probabilit
y", ylim = c(0, max(c(pmf_values, pmf_values_new))), main = "Poisson Distribution Comparison (λ = 5 vs λ = 1
0)")
> lines(k_values, pmf_values_new, type = "h", lwd = 2, col = "red")
> legend("topright", legend = c("λ = 5", "λ = 10"), col = c("blue", "red"), lwd = 2)
```



2C: What happens to the spread of the distribution as λ becomes smaller (e.g. λ=2)? Interpret this in the context of low call volumes.

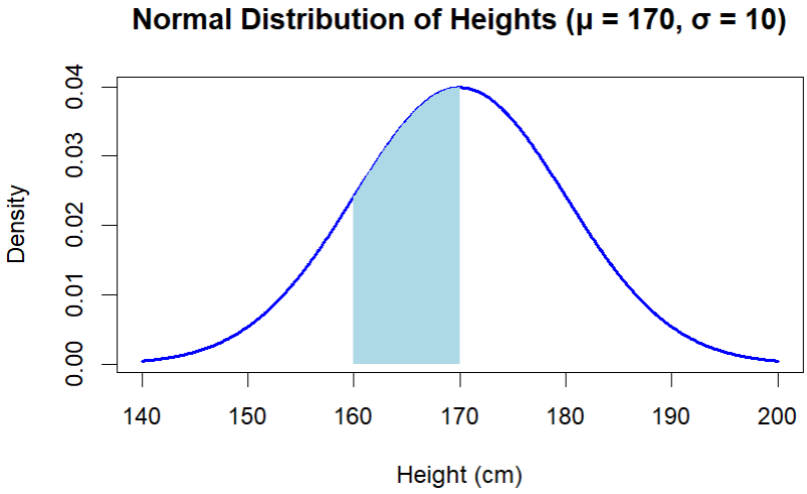
```
> lambda <- 2
> probabilities_2 <- dpois(x, lambda)
> plot(x, probabilities_2, type = "h", lwd = 2, col = "green", main = "Poisson Di
stribution (λ = 2)", xlab = "Number of Calls", ylab = "Probability")
```



3A: The heights of individuals often follow a normal distribution. If the average height of adults in a region is 170 cm with a standard deviation of 10 cm, use the normal distribution to calculate the probability of height ranging between 160 to 170 cm. Visualize the distribution.

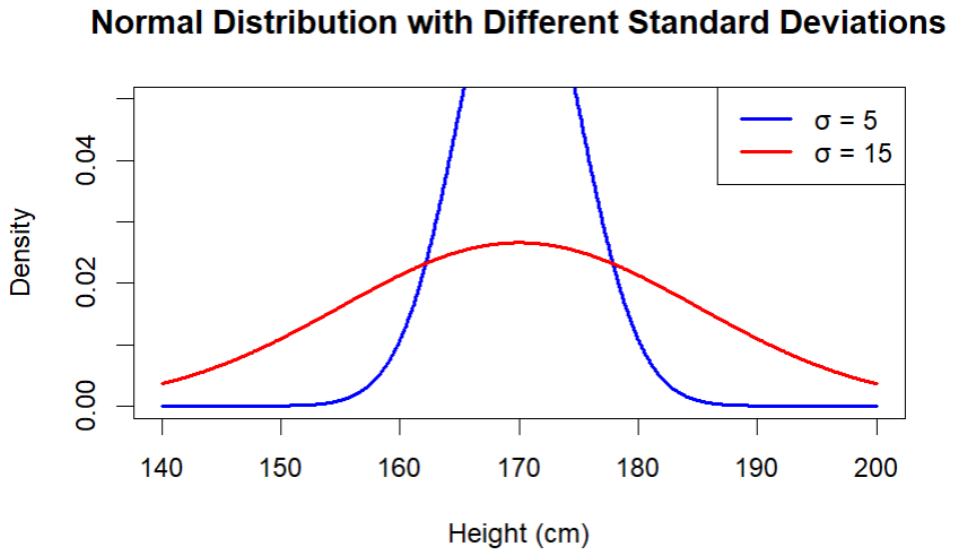
```
> mu <- 170
> sigma <- 10
> lower <- 160
> upper <- 170
> probability <- pnorm(upper, mean = mu, sd = sigma) - pnorm(lower, mean = mu, sd
= sigma)
> probability <- pnorm(upper, mean = mu, sd = sigma) - pnorm(lower, mean = mu, sd
= sigma)
> probability
[1] 0.3413447
> x <- seq(140, 200, length.out = 1000)
> y <- dnorm(x, mean = mu, sd = sigma)
> plot(x, y, type = "l", lwd = 2, col = "blue", main = "Normal Distribution of He
ights (μ = 170, σ = 10)", xlab = "Height (cm)", ylab = "Density")
> x_fill <- seq(160, 170, length.out = 100)
> y_fill <- dnorm(x_fill, mean = mu, sd = sigma)

> polygon(c(x_fill, rev(x_fill)), c(rep(0, length(x_fill)), rev(y_fill)), col =
"lightblue", border = NA)
```



3B: How does changing the standard deviation affect the shape of the bell curve? Visualize this in R for σ=5 and σ=15.

```
> sigma_5 <- 5
> sigma_15 <- 15
> x <- seq(140, 200, length.out = 1000)
> y_5 <- dnorm(x, mean = mu, sd = sigma_5)
> y_15 <- dnorm(x, mean = mu, sd = sigma_15)
> plot(x, y_5, type = "l", lwd = 2, col = "blue", main = "Normal Distribution wit
h Different Standard Deviations", xlab = "Height (cm)", ylab = "Density", ylim =
c(0, 0.05))
> lines(x, y_15, lwd = 2, col = "red")
> legend("topright", legend = c("σ = 5", "σ = 15"), col = c("blue", "red"), lwd =
2)
```



3C: What happens to the PDF if the mean shifts to 180 cm? How does this relate to real-world population differences?

```
> mu_new <- 180
> y_new <- dnorm(x, mean = mu_new, sd = sigma)
> plot(x, y_new, type = "l", lwd = 2, col = "green", main = "Normal Distribution
with Mean Shifted to 180 cm", xlab = "Height (cm)", ylab = "Density")
```