Assignment 3

2023-04-26

R has built-in functions for calculating probabilities from all known probability distributions and the capability for writing user defined functions for custom distributions. We will explore the use of these for the discrete and continuous distributions discussed in class. This means that we can create probability tables for all of these distributions for any values of their parameters. That means these functions completely replace the paper tables we have learned to use in class. However, we will need to learn to draw pictures for probability problems because we still need to use algebraic combinations of areas under the probability distributions to solve these problems; just as we did in class.

Each distribution function comes with a prefix.These prefixes are (where *** represent the R name, or R stem) for the distribution, which we will discuss below):

d***(x,) returns the density or value on the vertical (y-) axis of a probability distribution for a discrete value of x and other arguments.

p***(x,) by default this returns the cumulative density function or the area under the graph of the distribution's curve to the LEFT (lower tail) of an x value on a probability distribution curve and other arguments. This finds P(X <= x). You can set this to return the upper tail (see if you can figure out how by reading the documentation), but for simplicity we will use the default.

q***(x,) returns the quantile value (standardized z-value) for x and other arguments. The quantile function is the inverse of the cumulative probability function.
The p-quantile is the value of a random variable with the property that there ia a probability p of getting a value less than or equal to it. For example, the median is defined to be the 50% quantile.

r***(n,) returns a random deviate (simulation value) of size n from the given distribution and other arguments.

The R functions the we will consider in this assignment are:

binom() the binomial distribution with parameters n (of trials), p (probability of success on each trial)

pois() the Poisson distribution with parameter lambda

norm() the normal distribution with parameters the mean and standard deviations.

You should read the help documentation for each of these probability distributions and their prefixes.For example run "help(pbinom)".

```
help(pbinom)
```

For completeness, R stem for the multinomial distribution is multinom() with parameters of the number of possible outcomes for each trial, their probabilities of being drawn, and the number of outcomes of each kind, and for the hypergeometric distribution, the R stem is hyper() with parameters of the number of objects drawn, the number of objects of one kind (a success),the number of objects of a second kind (a failure)

Later in the course we will use:

```
*t() Student's t distribution with parameters
     the degrees of freedom

*chisq() the chi-squared distribution with
      parameters the degrees of freedom

*f()   the F-distribution with parameters of
       numerator degrees of freedom and denominator
       degrees of freedom. We will use this
       distribution when we study ANOVA and simple
       linear regression.
```

We will only explore the binomial, Poisson, and the normal distributions in this assignment. Ask for help on the core R statistics package, "stats". Take some time to look at the documentation for the various distributions (go to the bottom of the R Documentation that appears in the lower right pane in RStudio, then click "index").

```
?'stats-package'
```

Running the script below will give a list of all of the distributions available in R.

```
library(help = "stats")
```

Exercises

The binomial distribution

Please run the indicated scripts and post the results in the space provided.

Example 1

Q1. Find the probability, value of the cumulative distribution function at (the probability to the left of) a given number.

Below we find the probability of getting 5 or fewer success in 20 trials with a probability of success of 0.2 on each trial. Check the results by using Table V in F & P.

```
x = 5
n = 20
p = 0.2
pbinom(x,size=n,prob=p)  # size = number of trials,
```

```
## [1] 0.8042078
```

```
                         # prob = probability of success on each trial.
```

probability of getting 5 or fewer successes is 0.8042078

Q2. Find the probability of more than 5 successes in 20 trials with a probability of success of 0.2 on each trial. Check this result using Table V in F & P.

```
x = 5
n = 20
p = 0.2
1-pbinom(x,size=n,prob=p)
```

```
## [1] 0.1957922
```

probability of more than 5 successes is 0.1957922

Q3. Why did we use the script 1-pbinom(x,size=n,prob=p) to answer this problem? Please reply in the space below.

Because the probability of all outcomes sum to 1. So, by subtracting the probability of 5 or fewer from the total probability, we are left with the probability of the other binary option, or the complement.

Example 2

Q1. Find the probability, value of the cumulative distribution function at (the probability to the left of) a set (vector) of numbers.

```
x = c(0,1,2,5,8,10,15,20)
n = 20
p = 0.2
pbinom(x,size=n,prob=p)
```

```
## [1] 0.01152922 0.06917529 0.20608472 0.80420779 0.99001821 0.99943659 0.99999999
## [8] 1.00000000
```

0.01152922 0.06917529 0.20608472 0.80420779 0.99001821 0.99943659 0.99999999 1.00000000

Example 3

Q1. Find the probability, that in 60 tosses of a fair coin the tail comes up (occurs) as follows.

20, 25, or 30 times.

```
x = c(20,25,30)
n =60
p= 0.5
pbinom(x,size=n,prob=p)
```

```
## [1] 0.006744647 0.122530417 0.551289087
```

$P\{20\} = 0.006744647$, $P\{25\} = 0.122530417$, $P\{30\} = 0.551289087$

Less than 20 times.

```
x = 19
n =60
p = 0.5
pbinom(x,size=n,prob=p)
```

```
## [1] 0.003108801
```

$P\{<20\} = 0.003108801$

20 times or less.

```r
x = 20
n = 60
p = 0.5
pbinom(x,size=n,prob=p)
```

```
## [1] 0.006744647
```

P{<=20} = 0.006744647

Between 20 and 30 times.

```r
x1 = 19
n = 60
p = 0.5
left_tail = pbinom(x1, n, p)

x2 = 30
right_tail = 1 - pbinom(x2, n, p)

between = 1 - right_tail - left_tail
print(between)
```

```
## [1] 0.5481803
```

P{20 <= i <= 30} = 0.5481803

Getting 4 or less tails in two equivalent ways.

```r
# First method.
x = 4
n = 60
p = 0.2
pbinom(x,n,p)
```

```
## [1] 0.003932593
```

P{<4} = 0.003932593

```r
# Second method.
n = 60
p = 0.2
dbinom(0,size=n,prob=p)+
  dbinom(1,size=n,prob=p)+
  dbinom(2,size=n,prob=p)+
  dbinom(3,size=n,prob=p)+
  dbinom(4,size=n,prob=p)
```

```
## [1] 0.003932593
```

P{<4} = 0.003932593

Example 4

Q1. Find the complete probability distribution when n = 5 and p = 0.1. Check this result by referring to Table V in F & P.

```r
x = c(0:5)
n = 5
p = 0.1
pbinom(x,n,p)
```

```
## [1] 0.59049 0.91854 0.99144 0.99954 0.99999 1.00000
```

0.59049 0.91854 0.99144 0.99954 0.99999 1.00000

Example 5

Generate 8 random values (i.e., generate a random sample of size 8) from a binomial distribution n = 150 and p = 0.4. These values a often called random deviates. This procedure is a fundamental part of creating widely used probabilistic simulations that you may have heard referred to as Monte Carlo or MCMC (Markov Chain Monte Carlo) models. Try a GOOGLE Search to learn a bit about these simulations.

Note:

The values generated are generated by a computer with a finite memory. The values you see printed out appear as integers. That is because the program that created the values either truncated or rounded the values actually computed. When you look at the histogram, it probably was generated from the values actually calculated and not the ones that are printed, e.g., if a 12 is displayed by the print function, may have been calculated as 11.65 or 12.2. Do not forget this!!

Also do a GOOGLE search on random number generators to learn a little about them. After you do that you should be able to answer the question: Are random numbers truly random?

Q1. What did you find out about the randomness of random numbers? Please put your conclusions in the space below.

Are you kidding me? You want me to go google something as an assignment? I thought this was a college course, I can't believe I'm being told to do self learning a THIRD OF THE WAY through this massive assignment.

I already know about random numbers. They are not truly random. There is always a seed and limitations to memory.

```
help(rbinom)
```

The script below will generate a different set of values from a binomial distribution with n = 150 and p = 0.4 each time it is run. To see this, run the script a few times in succession and see the results.Make sure to run the histogram script as well.

Q2. Did it change?

```
x = 8
n = 150
p = 0.4
rbinom(x,size=n,prob=p)
```

```
## [1] 61 51 59 60 57 61 61 60
```

```
binomial = rbinom(x,size=n,prob=p)
```

Yes the numbers change.

Q3. Create a quick histogram from this sample using the core R function "hist()".

```
hist(binomial, main="A histogram of a simulated binomial distribution")
```

What mean does the histogram appear to have? Please place your answer in the space below.

The mean appears to be between 55 and 60

```
mean(binomial)
```

```
## [1] 60.125
```

The following script will always return the same set of values anytime it is run. The "set.seed()" function is what causes that. It is telling R to start at the same point each time it draws the random sample.

Q4. Can you think of a situation where being able to always generate the same set of random deviates might be useful? Run the script a few times in succession and see the results.Make sure to run the histogram script as well.Did it change?

```
set.seed(5172013)
x = 8
n = 150
p = 0.4
rbinom(x,size=150,prob=0.4)
```

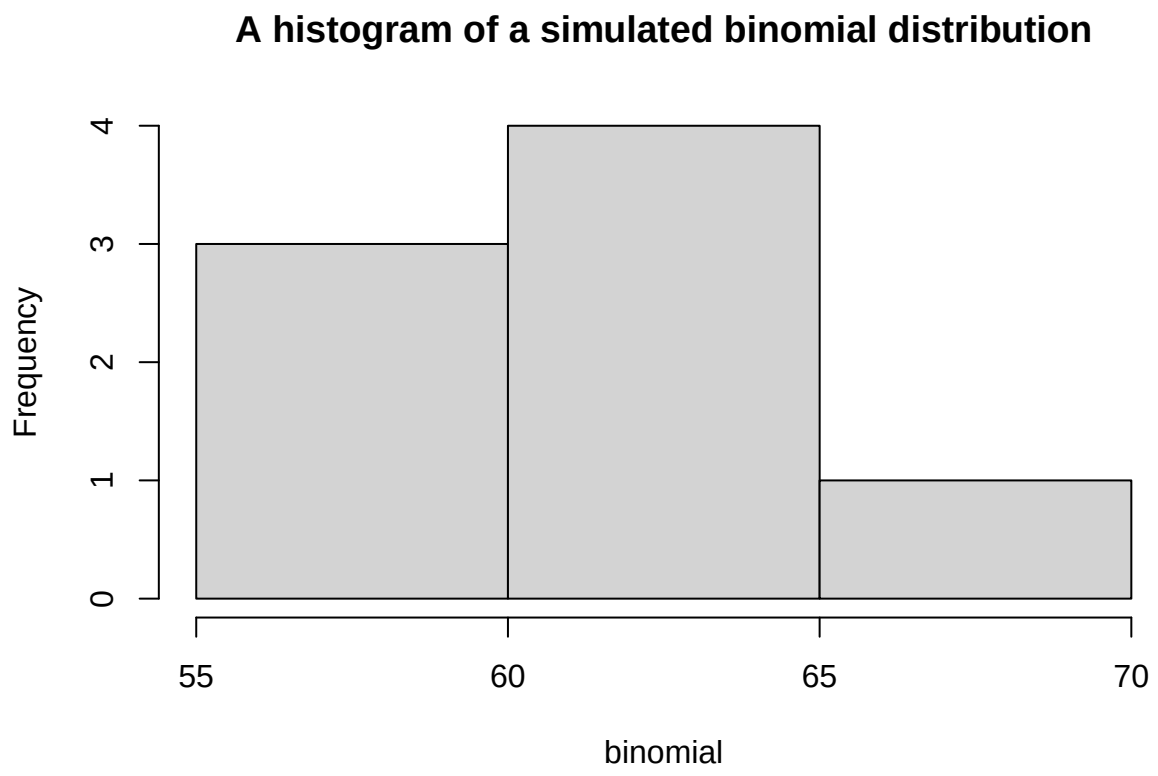
```
## [1] 73 60 61 69 67 69 57 52
```

```
binomial = rbinom(x,size=150,prob=0.4)
```

It did not change. It would be useful to save the seed if you are performing a bootstrap approach to solving a problem and needed the seed for reproducibility

Create a quick histogram from this sample using the core R function "hist()".

```
hist(binomial, main="A histogram of a simulated binomial distribution")
```



What mean does the histogram appear to have? Please place your answer in the space below.

```
mean(binomial)
```

```
## [1] 61.125
```

The mean appears to be between 60 and 65

Example 6

Just for fun! Run the following script.

We discussed the normal approximation to the binomial distribution. The R code below will give you an opportunity to explore binomial distribution and see how well he approximation does.

```
# Clear the memory
rm(list = ls())
```

Make the packages "dplyr" and "ggplot2" and their associated functions available to use.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(readr)
```

Load some necessary R packages.

```
#install.packages("Rtools")
#install.packages("manipulate")
library(manipulate)
```

Generate a binomial distribution using the core R function for a histogram, "hist()".

```
generate.binomial = function(nreps, ntrials, p){
  y = rbinom(n=nreps,size=ntrials,prob=p)
  hist(y, breaks=ntrials, xlim=c(0,ntrials), las=1,xlab="x",
       ylab="Frequency", col="red",
       main=NA)
}
```

The R script creates an interactive histogram graph of the binomial distribution. You can alter the variable "ntrials", "nreps", and "p" and change the histogram.

To work with this graph interactively, click on the wheel-like symbol in the upper left corner of the plot window and use the sliders to change the number of trials ("ntrials"), the number of times to run the experiment ("nreps"), and the probability of success ("p").

```
# manipulate(generate.binomial(nreps, ntrials, p),
#            nreps=slider(100000 - 1, 100000, initial=100000 - 1),
#            ntrials=slider(1000 - 1,1000,initial=1000 ),
#            p=slider(0, 1, initial=.5))
```

Q1. What is the expected value (center of the distribution)? Does the graph you create show the correct mean? To make this easy to see, set "ntrials" and "nreps" to their maximums and pick "p" = 0.5.

500

Q2. Does the expected value agree with what we learned in class about the binomial distribution?

Yes

End of binomial distribution Examples.

Now it is your turn.Please do the following exercises writing and running the appropriate R scripts. Do not forget to draw pictures of the problem!!

```
 # Clear the memory
rm(list = ls())
```

Problem 1

Note: You have done this problem before, so you should be able to verify that your answers to these questions are correct!

A medical doctor finds that 5% of mothers admit to having one or more glasses of wine per day during pregnancy.Fifteen (15) pregnant mothers are randomly selected (with replacement) from admission at a hospital. If X = a random variable that represents the number of mothers who admit to having one or more glasses of wine per day during pregnancy, then:

Q1. What probability distribution would you assume as a model for the outcomes of this experiment, i.e., what is the probability distribution of ?

I would assume a binomial distribution, number of mothers along the x and frequency of admittance on the y axis.

In the scripts you write to answer these questions, assigning the variables as follows:

x = 5 n = 15 p = 0.05

may make your work easier.

Q2. Using that distribution, what is the probability that exactly five (5) mothers will admit to drinking one or more glasses of wine during their pregnancy?

```
x = 5
n = 15
p = 0.05
pbinom(x, n, p) - pbinom(4,n,p)
```

```
## [1] 0.0005618772
```

Q3. Using that distribution, what is the probability that none of the mothers will admit to drinking one or more glasses of wine during their pregnancy?

```
x = 5
n = 15
p = 0.05
pbinom(0,n,p)
```

```
## [1] 0.4632912
```

Q4. Based on this sample of 15 pregnant mothers, what is the expected number of mothers who admit to drinking one or more glasses of wine during their pregnancy?

```
0.05 * 15
```

```
## [1] 0.75
```

Less than likely that one of them will admit it.

Q5. What is the standard deviation about this expected (mean) value?

```
sigma = sqrt(15 * .05 * 0.95)
print(sigma)
```

```
## [1] 0.8440972
```

The following are examples on Poisson distribution. Please run the indicated scripts.

```
# Clear the memory.
rm(list = ls())
```

Example 1

Q1. Find the probability, value of the cumulative distribution function at (the probability to the left of) a set (vector) of numbers.

```
x = c(0,1,2,5,8,10,15,20)
lambda = 6
```

Write the code that calculates the probability. Make sure to display the result.

```
dpois(x,lambda)
```

```
## [1] 2.478752e-03 1.487251e-02 4.461754e-02 1.606231e-01 1.032577e-01
## [6] 4.130309e-02 8.912556e-04 3.725062e-06
```

Example 2

A random variable, X, has a Poisson distribution with mean (lambda,the parameter of the distribution) of 7.

F &P do not provide a tables for the Poisson distribution.So use the calculator found at https://stattrek.com/online-calculator/poisson.aspx to check your results.

Q2.Find the probability that:

X is less than or equal to 5.

```
ppois(5,lambda)
```

```
## [1] 0.4456796
```

X is less than 5.

```
sum(dpois(0:4, lambda))
```

```
## [1] 0.2850565
```

X is strictly greater than 10.

```
sum(dpois(10:999, lambda))
```

```
## [1] 0.08392402
```

X is between 4 and 16.

```
left_tail = ppois(3,lambda)
right_tail = 1 - ppois(16, lambda)

between = 1 - left_tail - right_tail
print(between)
```

```
## [1] 0.8486212
```

Example 3

Generate 30 random values (i.e., generate a random sample of size 30, n) from a Poisson distribution lambda = 7.

```
rand_pois = rpois(30, 7)
print(rand_pois)
```

```
##  [1]  7  9 12 12  9  4 10  7  6  7  6 12  7 11  7 10  4 11  2  6  5  5  7  8  6
## [26]  8  4  7  4  9
```

Q1. The script below will generate a different set of values from a Poisson distribution with n = 30 and lambda = 7 each time it is run. To see this, running the script a few times in succession and see the results. Make sure to run the histogram script as well. Did it change?

```
events = 30
lambda = 7
rpois(events,lambda)
```

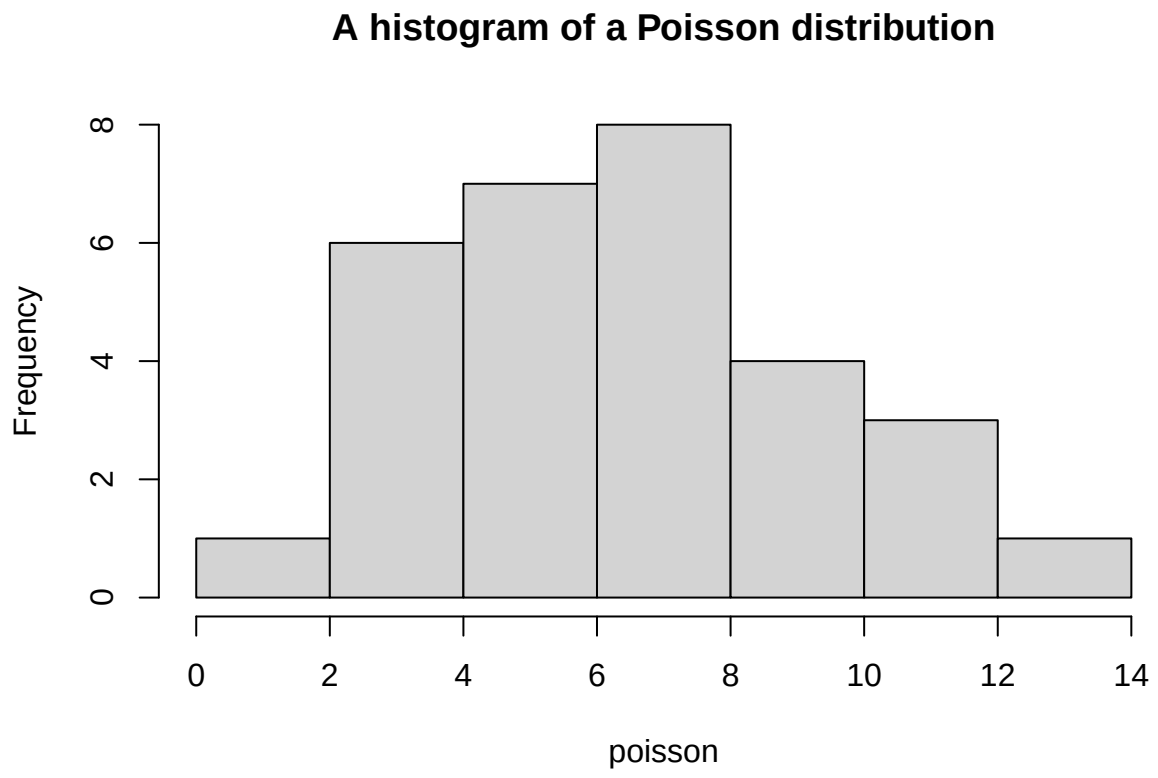
```
##  [1]  5  7  7 11  6  8 10  6  5  5  7  5 10  7  8  7  4  5 10  6  4  8 12 11  9
## [26]  4  7 12  8  8
```

```
poisson = rpois(events,lambda)
```

yes it changes.

Create a quick histogram from this sample using the core R function "hist()".

```
hist(poisson, main="A histogram of a Poisson distribution")
```



Q2 What mean does the histogram appear to have? Please place your answer in th espace below.

The mean appears to be around 6

The following script will always return the same set of values anytime it is run. The "set.seed()" function is what causes

that. It is telling R to start at the same point each time it draws the random sample.

```
set.seed(5172013)
events = 30
lambda = 7
rpois(events,lambda)
```

```
##  [1] 12  7  9 12  6  9  6  7 10  6 10 10  6  4  8  9  7  7  7  4  5  9  2  9  6
## [26]  6  4  8  4  4
```

```
poisson = rpois(events,lambda)
```

Create a quick histogram from this sample using the core R function "hist()".

```
hist(poisson, main="A histogram of a simulated Poisson distribution")
```

Q3. What mean does the histogram appear to have? Please place your answer in the space below.

7

Example 4

The following data represent the total number of aberrant crypt foci (abnormal growths in the colon) observed in 7 rats after being tested the carcinogen azoxymethane.

87, 53, 72, 90, 78, 85, 83.

Q1. Compute the sample mean and variance.

```
Data_Values = c(87, 53, 72, 90, 78, 85, 83)
```

Your code here:

```
data_mean = mean(Data_Values)
data_var = var(Data_Values)

print(data_mean)
```

```
## [1] 78.28571
```

```
print(data_var)
```

```
## [1] 159.9048
```

Q2. Is the Poisson distribution a good model for the population from which these data come?

We will perform a simulation experiement to try to answer this question.Later in the course, we will study a statistical procedure for answering such questions.

Notice the use of the "c()" to both do the calculation and print the results in on command. You could do this in multiple steps just as well. For example, the following would work also, but it just does not print the results in columns with headings without the added "print()". Run the first three (3) lines of the code, see what results, then run the two (2) "print()" functions and paste the results in the space after the script.

```
Data_Values = c(87, 53, 72, 90, 78, 85, 83)
Mean = mean(Data_Values)
Variance = var(Data_Values)
```

Show the results using code

```
print(Mean)
```

```
## [1] 78.28571
```

```
print(Variance)
```

```
## [1] 159.9048
```

[1] 78.28571 [1] 159.9048

We know that the mean and variance of the Poisson distribution are equal. From these results, it is unlikely that these data are from a Poisson distribution, So, it is doubtful that the Poisson distribution is a good model for this data.

```
# Clear the memory.
rm(list = ls())
```

Problem 1

Assume that average car traffic crossing the Tacoma Narrows bridge during a normal weekday morning is 25 cars per minute.

Q1. What probability distribution would you assume as a model for the outcomes of this experiment, i.e., what is the probability distribution of the number of cars crossing the brige per minute? Please add the units as well.

Poisson distribution (cars / minute)

Q2. Find the probability that 30 or fewer cars will cross the bridge in any particular minute.

```
ppois(30, 25)
```

```
## [1] 0.8633089
```

Q3. Find the probability that more than 30 or cars will cross the bridge in any particular minute.

```
1 - ppois(30,25)
```

```
## [1] 0.1366911
```

Normal distribution

Examples: Please run the indicated scripts.

```
# Clear the memory
rm(list = ls())
```

Example 1

Find the probability, value of the cumulative distribution function at (the probability to the left of) a set (vector) of numbers.

```
x = c(-2.1,-1.6,0,1.8,2.3,5.0,6.1,13.5)
Mean = 22
variance = 25
```

Show the results using code

```
pnorm(x,Mean,variance)
```

```
## [1] 0.1675230 0.1725848 0.1894297 0.2095453 0.2153484 0.2482522 0.2623882
## [8] 0.3669283
```

Example 2

A random variable, X, has a normal distribution with mean = 22 and variance = 25. Find the probability that:

X lies between 16.2 and 27.5.

```
Mean = 22
variance = 25
sd = sqrt(variance)
Std_Dev = sqrt(variance)
xupper = 1 - pnorm(27.5, Mean, sd)
xlower = pnorm(16.2, Mean, sd)
```

Show the results using code

```
1 - xupper - xlower
```

```
## [1] 0.7413095
```

X > 29.

```
Mean = 22
variance = 25
Std_Dev = sqrt(variance)
x = 29
```

Show the results using code

```
pnorm(x-1, Mean, Std_Dev)
```

```
## [1] 0.8849303
```

X < 17.

```
Mean = 22
variance = 25
Std_Dev = sqrt(variance)
x = 17
```

Show the results using code

```
pnorm(x-1, Mean, Std_Dev)
```

```
## [1] 0.1150697
```

X less than 15 OR greater than 25.

```
Mean = 22
variance = 25
sd = sqrt(variance)
xupper = 1-pnorm(25, Mean, sd)
xlower = pnorm(15, Mean, sd)
```

Show the results using code

```
1 - xupper - xlower
```

```
## [1] 0.6449902
```

Example 3

Generate 30 random values (i.e., generate a random sample of size 30, n) from a normal distribution with mean = 5 and variance = 1.

```
# Get help.
help(rnorm)
```

The script below will generate a different set of 30 values from a normal distribution with mean = 5 and variance = 1 each time it is run. To see this, run the script a few times in succession and see the results. Make sure to run the histogram

script as well. Did it change?

```
events = 30
Mean = 5
variance = 1
Std_Dev = sqrt(variance)
rnorm(events,Mean,Std_Dev)
```

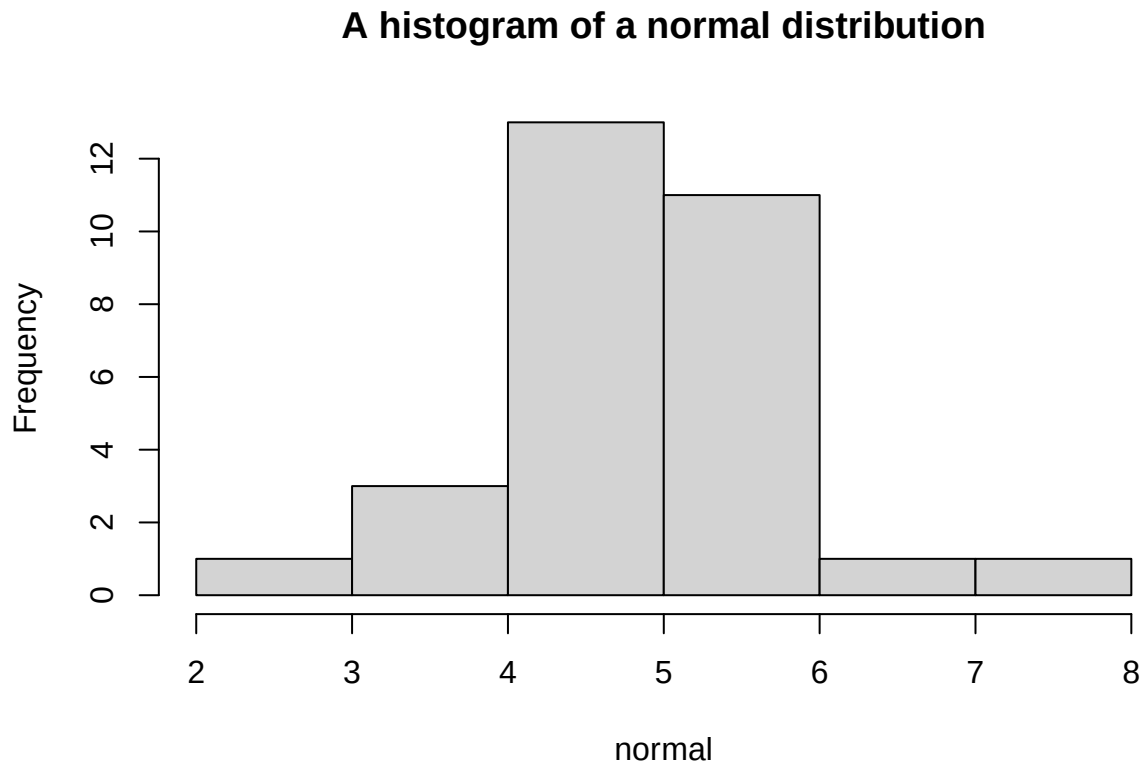
```
##  [1] 4.377839 5.060023 4.595939 3.740225 3.807574 4.181845 4.959602 4.711864
##  [9] 6.031924 4.453320 5.208427 6.010354 5.385390 3.904802 6.015521 3.679307
## [17] 6.736186 5.810683 5.169641 5.587027 4.978661 3.800848 4.272091 3.681215
## [25] 3.392807 4.912227 4.517569 5.230486 4.464720 3.681855
```

```
normal = rnorm(events,Mean,Std_Dev)
```

[1] 4.377839 5.060023 4.595939 3.740225 3.807574 4.181845 4.959602 4.711864 6.031924 4.453320 5.208427 6.010354 5.385390 3.904802 6.015521 3.679307 6.736186 5.810683 5.169641 5.587027 [21] 4.978661 3.800848 4.272091 3.681215 3.392807 4.912227 4.517569 5.230486 4.464720 3.681855

Create a quick histogram from this sample using the core R function "hist()".

```
hist(normal, main="A histogram of a normal distribution")
```



What mean does the histogram appear to have?

The mean appears to be between 4 and 5

The following script will always return the same set of values anytime it is run. The "set.seed()" function is what causes that. It is telling R to start at the same point each time it draws the random sample. Make sure to run the histogram script as well. Did it change?

```
set.seed(5172013)
events = 30
```

```
Mean = 5
variance = 1
Std_Dev = sqrt(variance)
rnorm(events,Mean,Std_Dev)
```

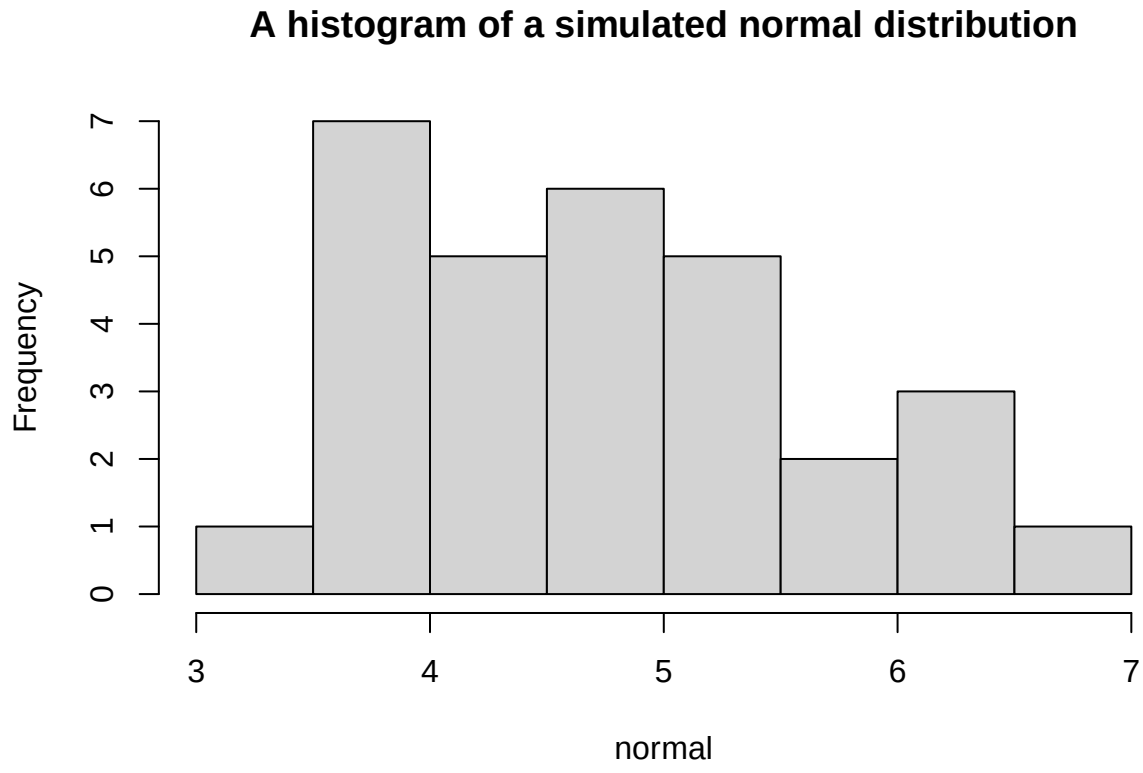
```
##  [1] 6.733120 5.825392 4.730318 4.611784 6.005063 6.016212 4.705863 5.360595
##  [9] 5.127265 4.905688 4.266713 2.960217 4.734936 3.926599 3.656979 4.339508
## [17] 4.654845 4.874448 3.177615 3.993504 5.060515 6.903521 5.815027 6.050127
## [25] 4.554848 4.512242 5.249195 4.903121 3.773400 2.686277
```

```
normal = rnorm(events,Mean,Std_Dev)
```

No changes

Create a quick histogram from this sample using the core R function "hist()".

```
hist(normal, main="A histogram of a simulated normal distribution")
```



What mean does the histogram appear to have?

4.745332

Example 4

A very important probability distribution is standard normal distribution, i.e., the normal distribution with mean = 0 and variance = 1. In other words, a random variable

$$X$$

is normally distributed as

$X \sim \mathcal{N}(0, 1)$

Generate 1000 random values (i.e., generate a random sample of size 1000, n) from a normal distribution with mean = 0 and variance = 1.

The script below will generate a different set of 1000 values from a normal distribution with mean = 0 and variance = 1 each time it is run. To see this, run the script a few times in succession and see the results. Make sure to run the histogram script as well. Did it change?

```
events = 1000
Mean = 0
variance = 1
Std_Dev = sqrt(variance)
rnorm(events,Mean,Std_Dev)
```

```
##    [1]  0.563262463  0.038349446 -0.394189884  0.546440749  2.230517805
##    [6]  0.646772596 -1.178923431 -0.319674310 -0.404179113 -0.275508519
##   [11] -1.815643727  0.863477229  0.164020376  0.413672874 -0.771656921
##   [16] -0.435233978  0.663370787 -0.807841981 -0.172731873  1.495299054
##   [21] -0.029552860  0.415483014 -0.729224571 -1.971397529 -0.447759623
##   [26]  0.792574037  0.754157615 -2.033769533 -0.794520183 -0.776064187
##   [31]  1.514560431  0.007656137  1.058553448  0.200537549 -0.718758363
##   [36] -0.571570729  0.503413152 -1.657705390 -1.085187707 -0.264691863
##   [41]  0.029542950 -0.220285394  2.337799425  0.274700837  0.119059400
##   [46]  1.328827365  0.149824788 -0.917801010  0.094575297  0.833080289
##   [51]  0.098316475 -0.850662534 -0.831499767  2.360490042  0.290309050
##   [56]  0.239852425  1.237677417  0.053843141 -0.247429963  0.184858571
##   [61] -0.549282344  1.040517980 -0.576927857  0.008718357 -0.997901024
##   [66]  0.048705424  0.594405775 -1.239227683 -0.204887296  0.377378763
##   [71]  0.084150053 -1.449683555  0.405027240  0.081042475 -0.078917169
##   [76]  0.736744340  0.234041463  0.115159113 -0.269995469 -0.082874219
##   [81]  1.115770649 -1.610119106  2.576430238  1.035053921  0.899511164
##   [86] -0.897752908  0.783362910  0.559415081 -0.407310641 -0.150601677
##   [91] -0.296076860  0.060418192 -1.036451141 -0.444323741  2.374690813
##   [96]  0.186056232  0.528416828  0.871402948  0.337428057  0.437414549
##  [101]  0.020537328  0.913113416 -0.106290509  0.815810048  1.164478816
##  [106]  1.506990320  1.377995500  1.551042584 -1.171350999 -1.056272291
##  [111]  0.790193070  1.082754078 -0.253653569 -0.134700781  1.321386770
##  [116]  0.572004575  0.234483876 -1.024679094  1.249156846 -0.447837180
##  [121] -1.354169837  0.206654578 -1.062883054  1.191308157 -1.734457671
##  [126]  0.606556811  1.423421820  0.267140293 -0.975320591  0.122610508
##  [131] -0.074729399 -0.835336705 -0.335254497 -0.388071127 -0.622837840
##  [136]  0.501442570 -0.563608159 -0.351592267 -0.516233425  0.374596980
##  [141] -0.637684479 -0.152455634  0.981021941  1.260291342 -2.846717069
##  [146] -0.568731118  0.920832799  0.501440522  0.041317815  0.790415560
##  [151] -0.303480115  0.359466972  0.711941972 -1.652004809 -0.794397294
##  [156]  0.899351374  1.150330762  0.130985316 -0.901304062  0.892941851
##  [161] -0.063307073  0.859736907 -0.453042027 -0.309443471 -1.471314612
##  [166]  0.240765778 -0.306563148 -0.029763291 -1.586043480  0.017740323
##  [171] -1.039489955 -0.703504585  0.160019050 -1.070150464 -0.092563299
##  [176] -0.845289068 -1.356755236  0.453795070 -0.937117135 -0.686663831
##  [181] -0.730446571  0.629517376 -0.329326788  0.728806983  0.874276071
##  [186] -0.820132217  0.574763727 -0.493666818  0.280489346  0.985976035
##  [191] -1.336725439  2.537232964  1.169316930  0.126455847  0.759666503
##  [196]  0.454172833  1.016352160  0.978562959 -0.600448616 -0.566073594
```

```
##   [201]  0.422719772 -0.309385047  0.563210503  1.124430985 -0.314828590
##   [206]  2.008176900 -0.768667068 -0.936650562  0.532141563  0.717802309
##   [211] -0.668508090 -0.520455770  0.141262967 -0.402598395  0.479873251
##   [216] -0.256949536  1.047468714 -1.529142699 -0.640864181  0.368530980
##   [221] -0.258755964 -0.022461773 -1.046891137  0.329187032  0.696323910
##   [226]  1.293772610  0.298729552 -0.780861012 -0.713166980 -0.247465120
##   [231]  0.276956943  0.123784298 -0.021774577 -0.611667009 -1.721408426
##   [236]  0.960492697  0.504561150 -1.141840364 -0.354315956 -1.088704000
##   [241] -1.405880234  0.306737811 -0.660905250  0.610349530  2.565888003
##   [246]  0.230408791  0.340177610 -0.094606827 -0.216600001 -0.646080843
##   [251] -1.171184572  0.949665939 -0.404060139 -0.070712581 -0.470802858
##   [256] -0.639615173  0.965822942 -0.727380724 -0.190906445 -1.771082591
##   [261] -0.049783482 -0.410914153 -0.216376524 -0.270278455 -0.201700787
##   [266]  1.268907161  1.328245082  0.720639466  0.468838088 -1.302703213
##   [271]  2.219976486  0.427054501 -0.917718212  1.405681140  0.459371522
##   [276] -0.872121241  0.848502732  0.642441151 -1.052377905 -2.023459952
##   [281]  0.326222169  0.428079929 -0.896359394  2.114822967 -0.874664868
##   [286]  0.284396513 -1.713935036 -0.936093160 -0.811809567 -1.088700578
##   [291] -0.644771699 -0.207318177 -1.524693691  0.442024991  1.681147196
##   [296]  0.426409935 -0.358392980 -0.496622905  1.020704513 -1.588191723
##   [301] -0.458600810  1.003486224  0.719936945  0.140440279 -1.196370906
##   [306]  1.426831769 -0.169451033 -0.875360125 -1.675447502  1.936260761
##   [311] -2.024011606 -0.042250364 -0.166362051  0.134780154  0.216814824
##   [316] -0.060075944  0.937802535  1.063995955  0.392301461  0.975921821
##   [321] -0.791466324  1.785468284 -0.104755087 -0.544367522 -0.153509869
##   [326]  0.168173927 -0.176419562  1.521159611 -0.931476622 -1.391788457
##   [331]  0.922704984  0.001136486  1.257013774  0.022403874 -1.303732307
##   [336] -2.668729208  0.407002526 -0.422253556  0.808156450  0.081645129
##   [341] -0.576292032 -0.498985149 -1.123543439 -0.587544570 -0.292205175
##   [346]  0.471686682  0.622921647  0.977794713 -1.020839233 -1.531822823
##   [351]  0.507759490  0.360751200  0.214776273 -0.522418485 -1.578881707
##   [356] -1.239323215 -0.116652668  0.247358852  0.421509210 -0.430903033
##   [361] -0.780500049 -0.008130924 -0.648115426 -1.215960166  0.929133493
##   [366] -1.297349599  2.851104072 -2.397189328 -0.336185882  1.424697920
##   [371]  0.641876121 -0.067593656 -0.576442356  0.268989377 -0.401161184
##   [376] -0.938794571  1.861867681 -0.223815223  0.658806565 -0.090952191
##   [381]  0.846985608  0.042490755 -1.496871737 -1.758724812 -1.129626773
##   [386]  1.318225960  1.019701236 -0.492460262 -1.556050933 -0.510256512
##   [391] -0.562749467  1.036366098 -0.429299319 -0.565037669 -0.237207714
##   [396] -0.672857587 -1.261131011  0.459161361 -1.705344349 -0.028333685
##   [401]  2.023355010  0.841108926  0.266550149 -1.638534442  1.454609260
##   [406] -0.029961094  1.116567954 -0.833782110  0.026473540 -2.996894988
##   [411] -0.435017994 -1.122187665  2.973451161 -0.863567978  0.983207685
##   [416] -1.166578375  1.486888820 -0.119624045 -0.092771299  0.944993022
##   [421]  0.414851859 -1.434024678  0.132574579 -2.074581520 -2.104292967
##   [426] -0.376750043 -0.619417831  1.909916018 -1.081108499  0.387338430
##   [431] -0.123201800 -0.750089260  1.314344166 -1.239271330  1.271321760
##   [436]  0.419152906 -0.615082595  0.182357674 -0.985674716 -0.955375289
##   [441]  0.898641213 -1.036227856  0.720357223  0.021832600 -1.801471868
##   [446] -0.337362778  1.038185307 -0.693266658 -0.839724617  0.405047884
##   [451]  0.388723058  1.015871606  0.398745583 -0.448474769  1.772227388
##   [456]  0.523052766 -0.415216306 -0.080045758 -0.393108467 -0.098358239
##   [461] -0.951500688  0.494354458  1.132670987 -0.307574843  0.123127094
##   [466]  1.752661630  0.563597537 -0.392979601  0.895876890  0.121926050
##   [471]  2.465837040 -0.718309410  0.301562641 -0.761336692  1.597869493
##   [476] -0.210424399 -0.331200535 -0.422293824  0.159050951  0.868034269
##   [481]  1.279494898 -1.304696992  0.340977961  0.845895033  0.042368397
##   [486]  1.593828639 -0.253614246  0.984010105 -0.442718734 -0.061011384
##   [491] -1.599862730 -2.461367586 -0.248867486 -0.857698523  0.708879183
```

```
##  [496]  0.304586664 -2.050648313 -1.451080971 -0.145370544 -0.129888698
##  [501] -0.527671023 -1.063485450 -1.379761793 -1.990932965 -0.206404526
##  [506]  1.347695440  0.453879885 -1.595931106  0.618339280 -1.383419535
##  [511]  0.043877643  0.291727915 -0.198421732 -0.384138702 -0.564411062
##  [516] -0.010634953  0.532181365  0.262565809  0.592239492 -1.021230387
##  [521]  0.153403295  1.742891395  0.506219839  1.926202710 -0.263244303
##  [526]  1.369467080 -1.106348818 -0.907291949  0.141580839  1.082226874
##  [531]  1.177486203  0.913628565 -0.539294273  0.078538725 -1.216870730
##  [536]  1.559792591 -0.629651138 -1.132493580  0.448308666 -1.625083714
##  [541] -0.040955926  0.448179196 -0.544062494  0.259204307 -0.004184179
##  [546] -1.030032432  0.781341307 -0.771544971  0.935065357 -0.369595078
##  [551] -0.765464795 -1.502875792 -0.921682765 -1.068270152  0.678917425
##  [556]  1.154558342  1.447163320  0.412587822 -0.711802614 -1.387909812
##  [561]  2.139350734  0.539292617  1.087242206 -0.719727651  0.807199167
##  [566] -0.490969660 -1.283994784 -2.578299884  0.826218220  0.201479549
##  [571] -0.537254530  0.633945039 -0.386065707 -0.210327212 -0.113280608
##  [576]  0.483126456 -0.153484635  0.705233209  2.263778347 -0.874046258
##  [581] -0.316370559  0.781529448  1.662461994  0.945847470  1.214731259
##  [586]  0.689561562  0.702013634  0.350819010  1.997448326 -0.678021159
##  [591] -0.498926241  0.812115877 -1.894206027 -0.865416750 -0.206580620
##  [596]  2.499107295 -0.277843330  0.168598085  0.644460220  0.095622243
##  [601] -0.133736701  0.593339804  0.422740211  1.084583560 -1.399973416
##  [606] -0.842729623 -0.143321414 -0.551209029  0.234299142 -2.001511313
##  [611]  1.036481374 -0.622126137 -0.180213431  1.168527999  0.611674165
##  [616] -1.366490258 -0.633517281 -0.577067714  0.246290026  0.321938924
##  [621] -1.057579188  2.907925060 -0.051813687 -1.153344797 -1.339080492
##  [626] -1.132454844 -1.060376736  0.499452947 -2.297580707 -0.527706514
##  [631]  0.996176044 -0.035879003 -0.656867012  1.170773654 -0.551985960
##  [636]  1.181656586 -0.089296651  0.892374438  0.588917254  0.417525927
##  [641] -1.743774518  0.556808857 -0.364523149 -0.476649695 -0.858782455
##  [646]  0.551455789 -1.602364795  0.383456230 -0.228933440  0.284183305
##  [651]  0.665667159 -0.555603158  1.671516234 -0.020434596  0.542297504
##  [656]  0.356491047  0.602944728  0.921183542 -0.437381259  0.610870727
##  [661] -0.399096852 -0.464150122 -1.073173583 -0.795678322  0.266440953
##  [666]  0.307885748 -0.416882921  1.681835191 -2.904808770 -0.330882948
##  [671] -0.315518920 -1.076052566 -0.126297977  0.077397149 -0.742052694
##  [676] -0.353730355 -1.634899266  0.781551332  0.020101748 -0.851810361
##  [681] -1.402008335  0.930028463 -1.710884007  1.011806606 -0.120616359
##  [686]  0.674012308  0.645682444  0.785231189  0.275118473  1.448262991
##  [691] -0.855582634  0.478717063  0.090866144 -0.525454984 -0.124760667
##  [696] -0.621155835  1.380152893 -0.495798251  0.712029029 -0.106350139
##  [701]  1.963881469  1.504560208 -0.747065148  1.277463010 -0.420200414
##  [706] -1.436334290 -0.267795143  1.525319139 -0.126903834 -0.422968992
##  [711] -0.426533606  0.997262577 -0.183364895 -0.009146357  1.316340562
##  [716]  1.128546724  0.478006440  0.228799231  0.093057976 -0.877827820
##  [721]  0.048979371  0.064343992  1.538962863  0.589276301  0.442379625
##  [726] -0.331723458  0.376192364 -1.227774555 -1.886678628  1.131456742
##  [731] -0.869025034 -0.124838831 -0.743509440 -0.571277971 -0.768158026
##  [736]  2.057924108  0.688604247 -1.279448203  0.747429190 -0.365992787
##  [741] -0.551664712 -0.046813424  0.286725735 -0.375928695 -0.397714923
##  [746] -0.353966279  0.018619651  0.847563349 -1.296365852  1.070645679
##  [751] -1.992321333  1.318410816  0.275764041  1.198626372  1.659208674
##  [756] -0.644539438 -0.316512971  1.053574996  0.744933366  0.787788153
##  [761] -1.389597736  0.403639883 -0.623502506 -1.311746188 -0.854343458
##  [766]  0.781480549  1.565653271  0.444423438  0.186534211  2.184116242
##  [771] -0.887723055  1.673119809  0.728259644  1.759104952  0.055893992
##  [776] -0.469769272  3.094726207  1.229452312  0.828249678  1.484529269
##  [781] -0.666763145  0.278958576  0.042517910 -0.859333041 -0.224658064
##  [786]  2.130776256  0.634135102 -0.542097461  0.370806286 -0.331925472
```

```
##  [791]  0.726550469 -0.759509597 -1.208397873  0.790393405 -0.214017178
##  [796]  2.062474718  1.304975324  0.013713446  0.721566930 -2.833245259
##  [801]  0.930565201  0.892672021 -1.827140873 -0.407793655  1.070693171
##  [806] -0.445919132 -0.047943323  0.955905621 -1.079535406 -0.208257413
##  [811]  2.357591681 -0.660094480  0.273064928 -0.040125147  2.014275565
##  [816] -2.811035880 -0.554861697  0.618211664 -0.199752025 -1.146449967
##  [821] -1.344676407 -0.925123283 -0.825794971  0.915376959  1.297829614
##  [826]  0.253106305 -1.125782859 -1.382566326 -2.712841699  1.637621677
##  [831]  0.310625139 -1.195908303  0.564035508  0.414000235  1.333120616
##  [836]  1.106497242  0.056351444  0.807333500  0.168735428 -0.740885575
##  [841]  0.625273194 -0.707059108  0.791180986 -1.208340451 -1.434493216
##  [846] -0.320133451  0.158650628  0.747515110  0.070069014 -1.038321420
##  [851] -0.212991318 -0.765120055  0.334768890 -0.267601779  1.097472750
##  [856] -0.111898463  1.872540998  0.311877423  1.912305274  1.472070228
##  [861] -0.930088864 -0.079430696  1.648768731 -0.022531775  1.999491668
##  [866]  0.734828373 -0.132264553 -1.204875718  0.474703386  0.561536845
##  [871]  0.196351030  0.522247894 -0.236938095 -0.306903519 -0.622261989
##  [876] -0.190945802  1.882900528 -1.989390673  0.064626708 -2.868773245
##  [881] -0.446391859 -0.567195815  0.889037362 -0.628433098  0.777755339
##  [886]  2.857411719  0.942647583  2.856699072 -1.479876913  0.192215199
##  [891]  0.204869704  1.606247155  1.212316649  1.327301968  0.762774617
##  [896] -0.164849389 -0.026372427 -0.981023327  0.241735094 -0.626874921
##  [901] -1.515478622  0.762193883 -1.423533459  0.100619560  0.129648929
##  [906] -0.203159766 -1.439973762  0.316022513  0.130782536 -0.476431527
##  [911]  0.936544598  2.040353041  0.776075068 -1.422122449 -0.918975449
##  [916] -0.011135651  1.181738199 -0.305606864  1.069190498 -1.777198681
##  [921] -1.059273297 -0.115985735  0.016405839 -0.555759815 -0.984570596
##  [926]  0.639176677  0.959308079  0.648927354 -0.033527912  0.635231305
##  [931] -0.277487225  0.901113061 -0.956942682 -1.035537093 -1.078261007
##  [936]  0.412359741 -1.059470448  0.105929819  1.057095435  0.888939760
##  [941] -1.251754741 -0.933042254 -0.891480473  0.347184168  0.748057167
##  [946]  1.003055731 -0.170032206 -1.388908626 -0.650461557  0.527495641
##  [951]  0.571768509  1.650844636 -1.261943436 -0.317015533  0.775455324
##  [956]  1.309861732 -0.466621332  1.276341053  0.819433805  0.459905547
##  [961] -1.046112757 -2.119458975 -0.203992965 -0.390587885 -1.000028815
##  [966]  1.409227186  0.587362881  0.471768795 -0.612869840 -0.190121957
##  [971]  0.174891106 -1.465165773 -0.679488387 -0.557301430  0.711724801
##  [976] -1.465969383 -1.063922899  1.851401094 -1.352140731  0.548042445
##  [981]  1.759810012  0.992437764 -1.917138389 -0.117239896  0.512163883
##  [986]  0.684670256  3.533580278 -0.252932030 -0.332733587  0.621159155
##  [991] -0.021885594 -0.683781752 -0.870337592  0.009805001 -0.083872344
##  [996]  0.116958858 -1.099970151  1.572849147  0.819062106 -0.224464089
```

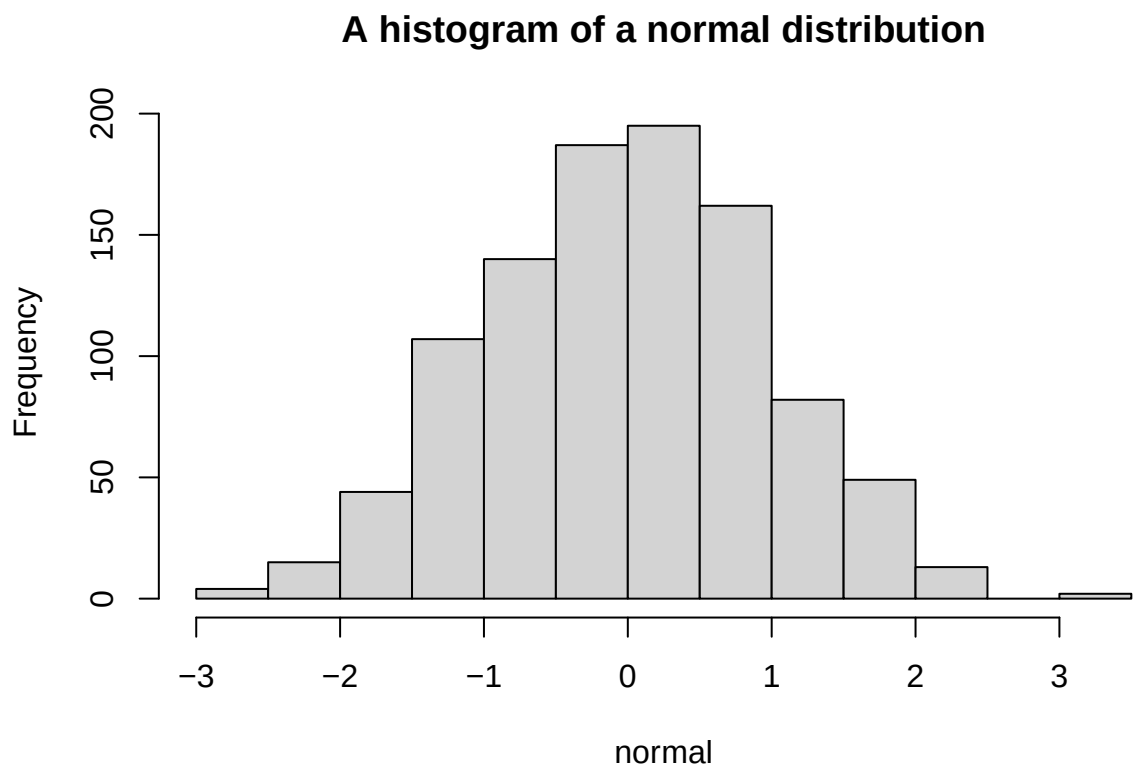
```
normal = rnorm(events,Mean,Std_Dev)
print('')
```

```
## [1] ""
```

There are 1,000 values generated, so there is no need to cut and paste them here. Notice how fast R generated them. Do review the histogram and think of how long it might take to do this with a calculator!

Create a quick histogram from this sample using the core R function "hist()".

```
hist(normal, main="A histogram of a normal distribution")
```



What mean does the histogram appear to have? Please place your answer in the space below.

The mean appears around 0.

The following script will always return the same set of values anytime it is run. The "set.seed()" function is what causes that. It is telling R to start at the same point each time it draws the random sample.

Make sure to run the histogram script as well. Did it change?

```
set.seed(5172013)
events = 1000
Mean = 0
variance = 1
Std_Dev = sqrt(variance)
rnorm(events,Mean,Std_Dev)
```

```
##    [1]  1.733119938  0.825392436 -0.269682429 -0.388215573  1.005062677
##    [6]  1.016212316 -0.294137317  0.360594665  0.127264966 -0.094311705
##   [11] -0.733287024 -2.039783203 -0.265063562 -1.073400818 -1.343021003
##   [16] -0.660492139 -0.345155240 -0.125552160 -1.822385279 -1.006495716
##   [21]  0.060515319  1.903520677  0.815027369  1.050127201 -0.445152006
##   [26] -0.487758426  0.249195174 -0.096879448 -1.226600481 -2.313722986
##   [31] -0.622160633  0.060022688 -0.404060720 -1.259775014 -1.192426024
##   [36] -0.818154935 -0.040398136 -0.288135810  1.031924489 -0.546680424
##   [41]  0.208426999  1.010354020  0.385389810 -1.095197892  1.015520622
##   [46] -1.320692955  1.736186030  0.810683247  0.169640869  0.587026969
##   [51] -0.021338845 -1.199152261 -0.727909300 -1.318785355 -1.607192846
##   [56] -0.087773322 -0.482430763  0.230486245 -0.535280127 -1.318145251
##   [61]  0.563262463  0.038349446 -0.394189884  0.546440749  2.230517805
##   [66]  0.646772596 -1.178923431 -0.319674310 -0.404179113 -0.275508519
##   [71] -1.815643727  0.863477229  0.164020376  0.413672874 -0.771656921
```

```
##   [76] -0.435233978  0.663370787 -0.807841981 -0.172731873  1.495299054
##   [81] -0.029552860  0.415483014 -0.729224571 -1.971397529 -0.447759623
##   [86]  0.792574037  0.754157615 -2.033769533 -0.794520183 -0.776064187
##   [91]  1.514560431  0.007656137  1.058553448  0.200537549 -0.718758363
##   [96] -0.571570729  0.503413152 -1.657705390 -1.085187707 -0.264691863
##  [101]  0.029542950 -0.220285394  2.337799425  0.274700837  0.119059400
##  [106]  1.328827365  0.149824788 -0.917801010  0.094575297  0.833080289
##  [111]  0.098316475 -0.850662534 -0.831499767  2.360490042  0.290309050
##  [116]  0.239852425  1.237677417  0.053843141 -0.247429963  0.184858571
##  [121] -0.549282344  1.040517980 -0.576927857  0.008718357 -0.997901024
##  [126]  0.048705424  0.594405775 -1.239227683 -0.204887296  0.377378763
##  [131]  0.084150053 -1.449683555  0.405027240  0.081042475 -0.078917169
##  [136]  0.736744340  0.234041463  0.115159113 -0.269995469 -0.082874219
##  [141]  1.115770649 -1.610119106  2.576430238  1.035053921  0.899511164
##  [146] -0.897752908  0.783362910  0.559415081 -0.407310641 -0.150601677
##  [151] -0.296076860  0.060418192 -1.036451141 -0.444323741  2.374690813
##  [156]  0.186056232  0.528416828  0.871402948  0.337428057  0.437414549
##  [161]  0.020537328  0.913113416 -0.106290509  0.815810048  1.164478816
##  [166]  1.506990320  1.377995500  1.551042584 -1.171350999 -1.056272291
##  [171]  0.790193070  1.082754078 -0.253653569 -0.134700781  1.321386770
##  [176]  0.572004575  0.234483876 -1.024679094  1.249156846 -0.447837180
##  [181] -1.354169837  0.206654578 -1.062883054  1.191308157 -1.734457671
##  [186]  0.606556811  1.423421820  0.267140293 -0.975320591  0.122610508
##  [191] -0.074729399 -0.835336705 -0.335254497 -0.388071127 -0.622837840
##  [196]  0.501442570 -0.563608159 -0.351592267 -0.516233425  0.374596980
##  [201] -0.637684479 -0.152455634  0.981021941  1.260291342 -2.846717069
##  [206] -0.568731118  0.920832799  0.501440522  0.041317815  0.790415560
##  [211] -0.303480115  0.359466972  0.711941972 -1.652004809 -0.794397294
##  [216]  0.899351374  1.150330762  0.130985316 -0.901304062  0.892941851
##  [221] -0.063307073  0.859736907 -0.453042027 -0.309443471 -1.471314612
##  [226]  0.240765778 -0.306563148 -0.029763291 -1.586043480  0.017740323
##  [231] -1.039489955 -0.703504585  0.160019050 -1.070150464 -0.092563299
##  [236] -0.845289068 -1.356755236  0.453795070 -0.937117135 -0.686663831
##  [241] -0.730446571  0.629517376 -0.329326788  0.728806983  0.874276071
##  [246] -0.820132217  0.574763727 -0.493666818  0.280489346  0.985976035
##  [251] -1.336725439  2.537232964  1.169316930  0.126455847  0.759666503
##  [256]  0.454172833  1.016352160  0.978562959 -0.600448616 -0.566073594
##  [261]  0.422719772 -0.309385047  0.563210503  1.124430985 -0.314828590
##  [266]  2.008176900 -0.768667068 -0.936650562  0.532141563  0.717802309
##  [271] -0.668508090 -0.520455770  0.141262967 -0.402598395  0.479873251
##  [276] -0.256949536  1.047468714 -1.529142699 -0.640864181  0.368530980
##  [281] -0.258755964 -0.022461773 -1.046891137  0.329187032  0.696323910
##  [286]  1.293772610  0.298729552 -0.780861012 -0.713166980 -0.247465120
##  [291]  0.276956943  0.123784298 -0.021774577 -0.611667009 -1.721408426
##  [296]  0.960492697  0.504561150 -1.141840364 -0.354315956 -1.088704000
##  [301] -1.405880234  0.306737811 -0.660905250  0.610349530  2.565888003
##  [306]  0.230408791  0.340177610 -0.094606827 -0.216600001 -0.646080843
##  [311] -1.171184572  0.949665939 -0.404060139 -0.070712581 -0.470802858
##  [316] -0.639615173  0.965822942 -0.727380724 -0.190906445 -1.771082591
##  [321] -0.049783482 -0.410914153 -0.216376524 -0.270278455 -0.201700787
##  [326]  1.268907161  1.328245082  0.720639466  0.468838088 -1.302703213
##  [331]  2.219976486  0.427054501 -0.917718212  1.405681140  0.459371522
##  [336] -0.872121241  0.848502732  0.642441151 -1.052377905 -2.023459952
##  [341]  0.326222169  0.428079929 -0.896359394  2.114822967 -0.874664868
##  [346]  0.284396513 -1.713935036 -0.936093160 -0.811809567 -1.088700578
##  [351] -0.644771699 -0.207318177 -1.524693691  0.442024991  1.681147196
##  [356]  0.426409935 -0.358392980 -0.496622905  1.020704513 -1.588191723
##  [361] -0.458600810  1.003486224  0.719936945  0.140440279 -1.196370906
##  [366]  1.426831769 -0.169451033 -0.875360125 -1.675447502  1.936260761
```

```
##   [371] -2.024011606 -0.042250364 -0.166362051  0.134780154  0.216814824
##   [376] -0.060075944  0.937802535  1.063995955  0.392301461  0.975921821
##   [381] -0.791466324  1.785468284 -0.104755087 -0.544367522 -0.153509869
##   [386]  0.168173927 -0.176419562  1.521159611 -0.931476622 -1.391788457
##   [391]  0.922704984  0.001136486  1.257013774  0.022403874 -1.303732307
##   [396] -2.668729208  0.407002526 -0.422253556  0.808156450  0.081645129
##   [401] -0.576292032 -0.498985149 -1.123543439 -0.587544570 -0.292205175
##   [406]  0.471686682  0.622921647  0.977794713 -1.020839233 -1.531822823
##   [411]  0.507759490  0.360751200  0.214776273 -0.522418485 -1.578881707
##   [416] -1.239323215 -0.116652668  0.247358852  0.421509210 -0.430903033
##   [421] -0.780500049 -0.008130924 -0.648115426 -1.215960166  0.929133493
##   [426] -1.297349599  2.851104072 -2.397189328 -0.336185882  1.424697920
##   [431]  0.641876121 -0.067593656 -0.576442356  0.268989377 -0.401161184
##   [436] -0.938794571  1.861867681 -0.223815223  0.658806565 -0.090952191
##   [441]  0.846985608  0.042490755 -1.496871737 -1.758724812 -1.129626773
##   [446]  1.318225960  1.019701236 -0.492460262 -1.556050933 -0.510256512
##   [451] -0.562749467  1.036366098 -0.429299319 -0.565037669 -0.237207714
##   [456] -0.672857587 -1.261131011  0.459161361 -1.705344349 -0.028333685
##   [461]  2.023355010  0.841108926  0.266550149 -1.638534442  1.454609260
##   [466] -0.029961094  1.116567954 -0.833782110  0.026473540 -2.996894988
##   [471] -0.435017994 -1.122187665  2.973451161 -0.863567978  0.983207685
##   [476] -1.166578375  1.486888820 -0.119624045 -0.092771299  0.944993022
##   [481]  0.414851859 -1.434024678  0.132574579 -2.074581520 -2.104292967
##   [486] -0.376750043 -0.619417831  1.909916018 -1.081108499  0.387338430
##   [491] -0.123201800 -0.750089260  1.314344166 -1.239271330  1.271321760
##   [496]  0.419152906 -0.615082595  0.182357674 -0.985674716 -0.955375289
##   [501]  0.898641213 -1.036227856  0.720357223  0.021832600 -1.801471868
##   [506] -0.337362778  1.038185307 -0.693266658 -0.839724617  0.405047884
##   [511]  0.388723058  1.015871606  0.398745583 -0.448474769  1.772227388
##   [516]  0.523052766 -0.415216306 -0.080045758 -0.393108467 -0.098358239
##   [521] -0.951500688  0.494354458  1.132670987 -0.307574843  0.123127094
##   [526]  1.752661630  0.563597537 -0.392979601  0.895876890  0.121926050
##   [531]  2.465837040 -0.718309410  0.301562641 -0.761336692  1.597869493
##   [536] -0.210424399 -0.331200535 -0.422293824  0.159050951  0.868034269
##   [541]  1.279494898 -1.304696992  0.340977961  0.845895033  0.042368397
##   [546]  1.593828639 -0.253614246  0.984010105 -0.442718734 -0.061011384
##   [551] -1.599862730 -2.461367586 -0.248867486 -0.857698523  0.708879183
##   [556]  0.304586664 -2.050648313 -1.451080971 -0.145370544 -0.129888698
##   [561] -0.527671023 -1.063485450 -1.379761793 -1.990932965 -0.206404526
##   [566]  1.347695440  0.453879885 -1.595931106  0.618339280 -1.383419535
##   [571]  0.043877643  0.291727915 -0.198421732 -0.384138702 -0.564411062
##   [576] -0.010634953  0.532181365  0.262565809  0.592239492 -1.021230387
##   [581]  0.153403295  1.742891395  0.506219839  1.926202710 -0.263244303
##   [586]  1.369467080 -1.106348818 -0.907291949  0.141580839  1.082226874
##   [591]  1.177486203  0.913628565 -0.539294273  0.078538725 -1.216870730
##   [596]  1.559792591 -0.629651138 -1.132493580  0.448308666 -1.625083714
##   [601] -0.040955926  0.448179196 -0.544062494  0.259204307 -0.004184179
##   [606] -1.030032432  0.781341307 -0.771544971  0.935065357 -0.369595078
##   [611] -0.765464795 -1.502875792 -0.921682765 -1.068270152  0.678917425
##   [616]  1.154558342  1.447163320  0.412587822 -0.711802614 -1.387909812
##   [621]  2.139350734  0.539292617  1.087242206 -0.719727651  0.807199167
##   [626] -0.490969660 -1.283994784 -2.578299884  0.826218220  0.201479549
##   [631] -0.537254530  0.633945039 -0.386065707 -0.210327212 -0.113280608
##   [636]  0.483126456 -0.153484635  0.705233209  2.263778347 -0.874046258
##   [641] -0.316370559  0.781529448  1.662461994  0.945847470  1.214731259
##   [646]  0.689561562  0.702013634  0.350819010  1.997448326 -0.678021159
##   [651] -0.498926241  0.812115877 -1.894206027 -0.865416750 -0.206580620
##   [656]  2.499107295 -0.277843330  0.168598085  0.644460220  0.095622243
##   [661] -0.133736701  0.593339804  0.422740211  1.084583560 -1.399973416
```

```
##  [666] -0.842729623 -0.143321414 -0.551209029  0.234299142 -2.001511313
##  [671]  1.036481374 -0.622126137 -0.180213431  1.168527999  0.611674165
##  [676] -1.366490258 -0.633517281 -0.577067714  0.246290026  0.321938924
##  [681] -1.057579188  2.907925060 -0.051813687 -1.153344797 -1.339080492
##  [686] -1.132454844 -1.060376736  0.499452947 -2.297580707 -0.527706514
##  [691]  0.996176044 -0.035879003 -0.656867012  1.170773654 -0.551985960
##  [696]  1.181656586 -0.089296651  0.892374438  0.588917254  0.417525927
##  [701] -1.743774518  0.556808857 -0.364523149 -0.476649695 -0.858782455
##  [706]  0.551455789 -1.602364795  0.383456230 -0.228933440  0.284183305
##  [711]  0.665667159 -0.555603158  1.671516234 -0.020434596  0.542297504
##  [716]  0.356491047  0.602944728  0.921183542 -0.437381259  0.610870727
##  [721] -0.399096852 -0.464150122 -1.073173583 -0.795678322  0.266440953
##  [726]  0.307885748 -0.416882921  1.681835191 -2.904808770 -0.330882948
##  [731] -0.315518920 -1.076052566 -0.126297977  0.077397149 -0.742052694
##  [736] -0.353730355 -1.634899266  0.781551332  0.020101748 -0.851810361
##  [741] -1.402008335  0.930028463 -1.710884007  1.011806606 -0.120616359
##  [746]  0.674012308  0.645682444  0.785231189  0.275118473  1.448262991
##  [751] -0.855582634  0.478717063  0.090866144 -0.525454984 -0.124760667
##  [756] -0.621155835  1.380152893 -0.495798251  0.712029029 -0.106350139
##  [761]  1.963881469  1.504560208 -0.747065148  1.277463010 -0.420200414
##  [766] -1.436334290 -0.267795143  1.525319139 -0.126903834 -0.422968992
##  [771] -0.426533606  0.997262577 -0.183364895 -0.009146357  1.316340562
##  [776]  1.128546724  0.478006440  0.228799231  0.093057976 -0.877827820
##  [781]  0.048979371  0.064343992  1.538962863  0.589276301  0.442379625
##  [786] -0.331723458  0.376192364 -1.227774555 -1.886678628  1.131456742
##  [791] -0.869025034 -0.124838831 -0.743509440 -0.571277971 -0.768158026
##  [796]  2.057924108  0.688604247 -1.279448203  0.747429190 -0.365992787
##  [801] -0.551664712 -0.046813424  0.286725735 -0.375928695 -0.397714923
##  [806] -0.353966279  0.018619651  0.847563349 -1.296365852  1.070645679
##  [811] -1.992321333  1.318410816  0.275764041  1.198626372  1.659208674
##  [816] -0.644539438 -0.316512971  1.053574996  0.744933366  0.787788153
##  [821] -1.389597736  0.403639883 -0.623502506 -1.311746188 -0.854343458
##  [826]  0.781480549  1.565653271  0.444423438  0.186534211  2.184116242
##  [831] -0.887723055  1.673119809  0.728259644  1.759104952  0.055893992
##  [836] -0.469769272  3.094726207  1.229452312  0.828249678  1.484529269
##  [841] -0.666763145  0.278958576  0.042517910 -0.859333041 -0.224658064
##  [846]  2.130776256  0.634135102 -0.542097461  0.370806286 -0.331925472
##  [851]  0.726550469 -0.759509597 -1.208397873  0.790393405 -0.214017178
##  [856]  2.062474718  1.304975324  0.013713446  0.721566930 -2.833245259
##  [861]  0.930565201  0.892672021 -1.827140873 -0.407793655  1.070693171
##  [866] -0.445919132 -0.047943323  0.955905621 -1.079535406 -0.208257413
##  [871]  2.357591681 -0.660094480  0.273064928 -0.040125147  2.014275565
##  [876] -2.811035880 -0.554861697  0.618211664 -0.199752025 -1.146449967
##  [881] -1.344676407 -0.925123283 -0.825794971  0.915376959  1.297829614
##  [886]  0.253106305 -1.125782859 -1.382566326 -2.712841699  1.637621677
##  [891]  0.310625139 -1.195908303  0.564035508  0.414000235  1.333120616
##  [896]  1.106497242  0.056351444  0.807333500  0.168735428 -0.740885575
##  [901]  0.625273194 -0.707059108  0.791180986 -1.208340451 -1.434493216
##  [906] -0.320133451  0.158650628  0.747515110  0.070069014 -1.038321420
##  [911] -0.212991318 -0.765120055  0.334768890 -0.267601779  1.097472750
##  [916] -0.111898463  1.872540998  0.311877423  1.912305274  1.472070228
##  [921] -0.930088864 -0.079430696  1.648768731 -0.022531775  1.999491668
##  [926]  0.734828373 -0.132264553 -1.204875718  0.474703386  0.561536845
##  [931]  0.196351030  0.522247894 -0.236938095 -0.306903519 -0.622261989
##  [936] -0.190945802  1.882900528 -1.989390673  0.064626708 -2.868773245
##  [941] -0.446391859 -0.567195815  0.889037362 -0.628433098  0.777755339
##  [946]  2.857411719  0.942647583  2.856699072 -1.479876913  0.192215199
##  [951]  0.204869704  1.606247155  1.212316649  1.327301968  0.762774617
##  [956] -0.164849389 -0.026372427 -0.981023327  0.241735094 -0.626874921
```

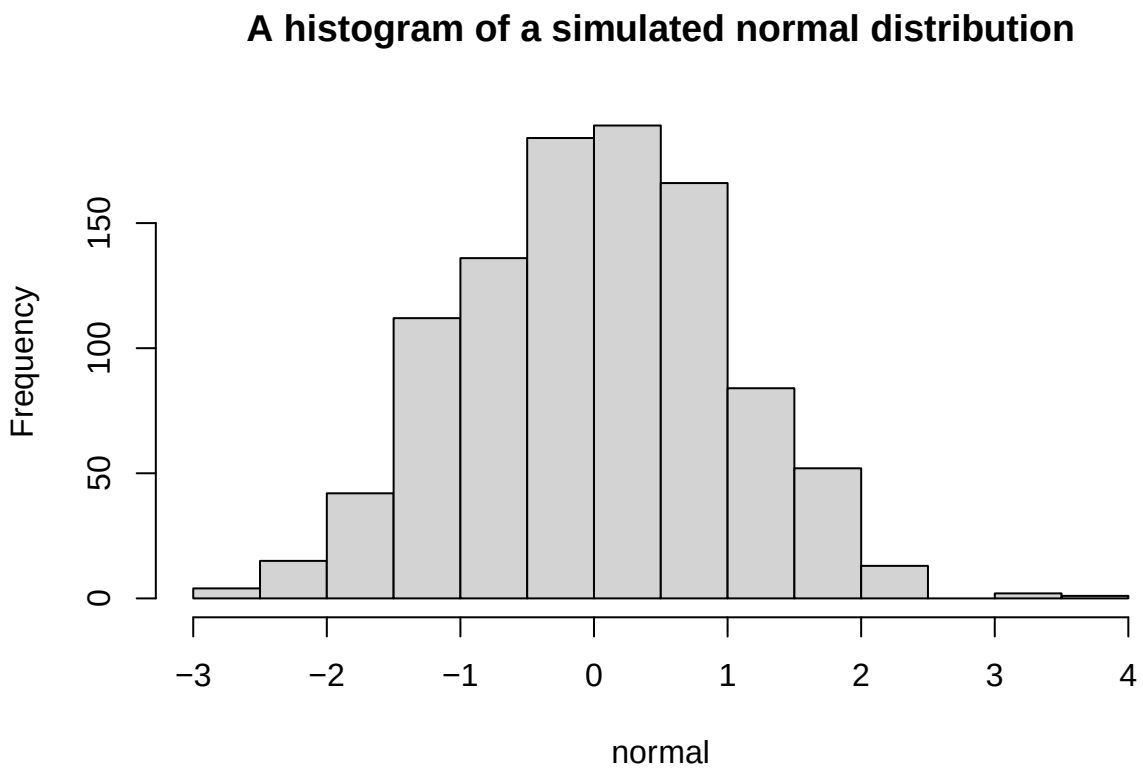
```
##   [961] -1.515478622  0.762193883 -1.423533459  0.100619560  0.129648929
##   [966] -0.203159766 -1.439973762  0.316022513  0.130782536 -0.476431527
##   [971]  0.936544598  2.040353041  0.776075068 -1.422122449 -0.918975449
##   [976] -0.011135651  1.181738199 -0.305606864  1.069190498 -1.777198681
##   [981] -1.059273297 -0.115985735  0.016405839 -0.555759815 -0.984570596
##   [986]  0.639176677  0.959308079  0.648927354 -0.033527912  0.635231305
##   [991] -0.277487225  0.901113061 -0.956942682 -1.035537093 -1.078261007
##   [996]  0.412359741 -1.059470448  0.105929819  1.057095435  0.888939760
```

```
normal = rnorm(events,Mean,Std_Dev)
```

No change

Create a quick histogram from this sample using the core R function "hist()".

```
hist(normal, main="A histogram of a simulated normal distribution")
```



What mean does the histogram appear to have?

Again around zero

Example 5

If a random variable is distributed as N(10,25), what is the probability that it will take on a value between 12 and 15?

```
mean = 10
variance = 25
```

```
sd = sqrt(variance)
xupper = 15
right_tail = 1 - pnorm(15, mean, sd)
xlower = 12
left_tail = pnorm(12, mean, sd)
```

Show the results using code

```
1 - right_tail - left_tail
```

```
## [1] 0.185923
```

Now, let's resolve this problem, but this time employ the method of converting to standard units. Please check this against Example 9.5 in F & P.

$z = \frac{\bar{x}-\mu}{\frac{\sigma}{\sqrt{n}}}$

where $z \sim \mathcal{N}(0, 1)$

Transform the data values and print the results.For this example,

$$z = \frac{x - 10}{\sqrt{25}}$$

Specifically,

$$z_{upper} = \frac{15 - 10}{5}$$

,

and

$$z_{lower} = \frac{12 - 10}{5}$$

Please place the results of the intermediate steps of this transformation in the spaces provided after the steps.

```
zupper = (xupper-mean)/sd
zlower = (xlower-mean)/sd
```

Show the results using code

```
zupper
```

```
## [1] 1
```

```
zlower
```

```
## [1] 0.4
```

Transform the population mean and standard deviation.

```
TransformedMean = 0
TransformedStd_Dev = 1
```

Calculate the probability.

```
right_tail = 1 - pnorm(zupper, TransformedMean, TransformedStd_Dev)
left_tail = pnorm(zlower, TransformedMean, TransformedStd_Dev)
1 - right_tail - left_tail
```

```
## [1] 0.185923
```

What do you notice about the two solutions? Is that surprising? Why or why not?

They are the same. Not surprising because all we did was normalize the data.

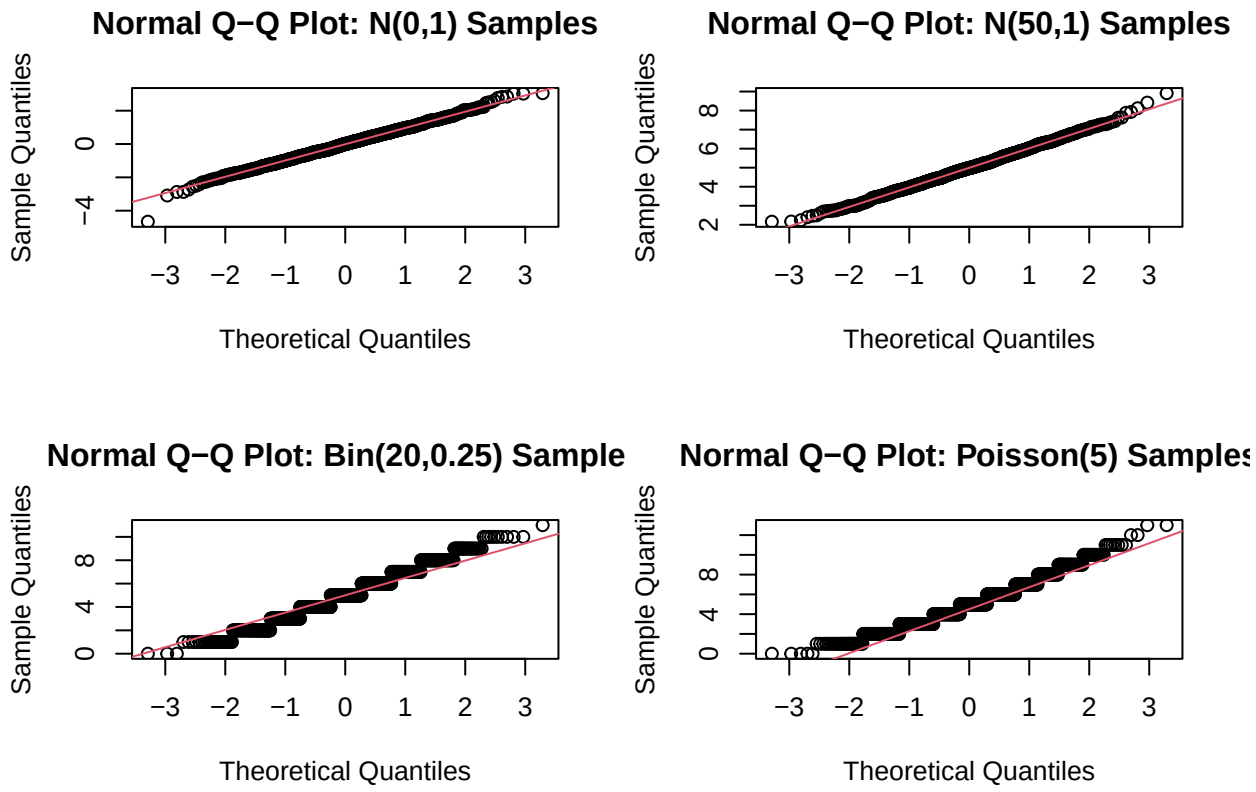
Example 6

Later in the course we will learn that the normal distribution is a key assumption of several important statistical procedures. Since we always explore our data instead of simply attacking it with various statistical procedures, it would be good to have a way to check that this assumption is at least approximately true. We will learn more about this later, but for now we will explore an R function that does this,"qqnorm()". This function produces what called a Q-Q plot.

It plots the quantiles of our sample data against the theoretical quantiles of any normal distribution,including the N(0,1) distribution. Our sample data can be assumed to come from any distribution and "qqnorm()" will help us see it our data is approximately normally distributed.

Below is an R script that creates this plot.

```
stdnormsample = rnorm(1000,mean=0)
normsample = rnorm(1000,mean=5)
binomsample = rbinom(1000,size=20,prob=0.25)
poissample = rpois(1000,5)

par(mfrow=c(2,2))   # array plots 2 x 2 on the page

qqnorm(stdnormsample, main="Normal Q-Q Plot: N(0,1) Samples")
qqline(stdnormsample,col=2)

qqnorm(normsample, main="Normal Q-Q Plot: N(50,1) Samples")
qqline(normsample,col=2)

qqnorm(binomsample, main="Normal Q-Q Plot: Bin(20,0.25) Samples")
qqline(binomsample,col=2)

qqnorm(poissample, main="Normal Q-Q Plot: Poisson(5) Samples")
qqline(poissample,col=2)
```



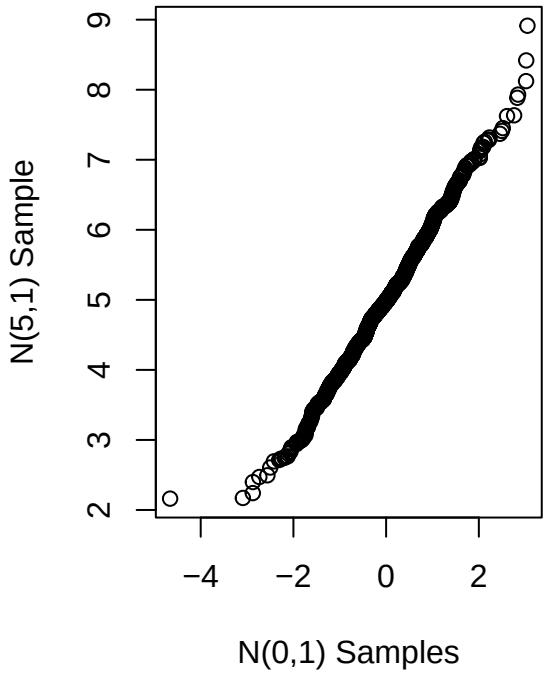
Departures from the straight red line (the red color is obtained by setting the parameter "col=2" in "qqline()") indicates the sample data departure from the assumption of normality.

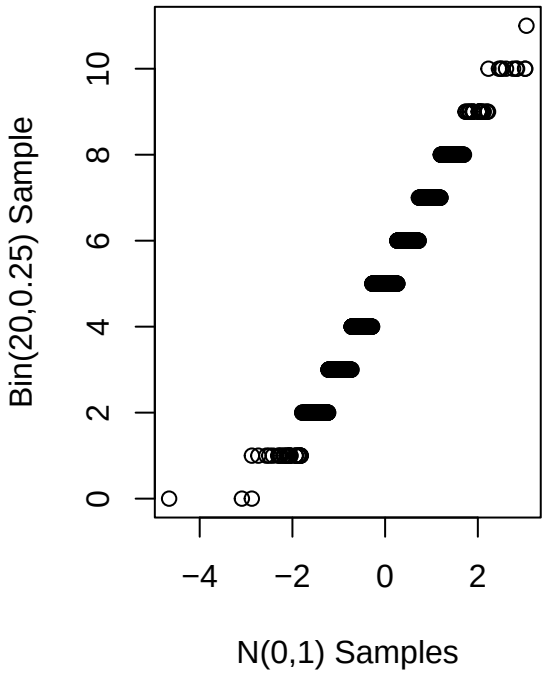
Knowing what you know about algebra, what does that straight line represent with regarded to the two variables plotted on the axes?

The two variables are equal along the straight line.

We can also use the R function "qqplot()" to plot sample quantiles for a given sample against the sample quantiles of another sample as follows:

```
?qqplot()

par(mfrow=c(1,2))

qqplot(stdnormsample,normsample,xlab='N(0,1) Samples',
   ylab='N(5,1) Sample')

qqplot(stdnormsample, binomsample,xlab='N(0,1) Samples',
   ylab='Bin(20,0.25) Sample')
```





We examine these plots to see if the data from the sample data lies along a straight line. Departure from that straight line relationship indicates the sample data is unlikely to be distributed as the theoretical distribution, which in these two cases is N(0,1).

Exercies

Now it is your turn.Please do the following exercises writing and running the appropriate R scripts.Do not forget to draw pictures of the problem!!

```
# Clear the memory.
rm(list = ls())
```

Problem 1

The mean and variance of the weights of NFL (National Football League) players are 248.5 lb and 278.89 square lb, respectively.Assume that the population is approximately normally distributed.

Q1. If you select a player at random, what is the probability that he is heavier than 270 lb.?

NOTE: I STRONGLY suggest you draw that picture first to help you solve the problem! Make sure to check your solution to this problem using Table I in F & P!

a. Define the random variable in the space below.

Random variable X is weight

b. What probability distribution would you assume as a model for the outcomes of this experiment?

Normal distribution

c. Find the probability that that the weight of an NFL player is greater than 270 lb. State the problem in probability terms.

```
1 - pnorm(270, 248.5, sqrt(278.89))
```

```
## [1] 0.09897307
```

P{w_NFL > 270} = 0.09897307

Solve this problem without and with using the standard normal transformation. Compare your results.

Place your code below. Make sure to show the numerical results.

```
z = (270 - 248.5) / sqrt(278.89)
print(z)
```

```
## [1] 1.287425
```

```
tx_mean = 0
tx_sd = 1
```

Calculate the desired probability.

```
1 - pnorm(z, tx_mean, tx_sd)
```

```
## [1] 0.09897307
```

Using the standard normal transformation, we seek to find $P(Z > z) $.

Place your code below. Make sure to show the numerical results.

Q2. Create an R script defining a population mean and standard deviation to be equal to 0 and 1, respectively, then demonstrate that what you did was correct. If you wish, you may call these variables (assigned objects) "StandardMean" and "StandardStd_Dev", respectively, but feel free to create your own variable (object) names.

Place your code below.

```
z = (270 - 248.5) / sqrt(278.89)
print(z)
```

```
## [1] 1.287425
```

```
tx_mean = 0
tx_sd = 1
```

Using the objects (variables) you created above, create an R script to calculate the desired probability and paste the result in the space below.

Place your code below.

```
1 - pnorm(z, tx_mean, tx_sd)
```

```
## [1] 0.09897307
```

Q3. In a few sentences, please write a comparison of the results in the space below.

The answers are the same. The probabilities do not change because the relationships within the data have not changed. They have only been transformed, which can be helpful when plotting large ranges of data.

The End of Assignment #3.