Lab 2 Key

#1. Start R Studio and in the upper left pane, write code for creating a vector, x, with entries 0, 1, 2, ..., 10.
```r
x <- 0:10
```

## [1] 0 1 2 3 4 5 6 7 8 9 10

#2. Add the following code to the text processing pane and run all the code. What is plotted?
```r
plot(x, dbinom(x, size = 10, prob = 0.2), type = "h", ylim = c(0, 1))
```

#3. Make and compare "spike graphs"/"pin plots" for probabilities from the binomial distributions with n = 10 and p = .1, .2, .5, .8, and .9.
```r
plot(x, dbinom(x, size = 10, prob = 0.1), type = "h", ylim = c(0, 1))
```
plot(x, dbinom(x, size=10, prob=0.1), type="h", ylim=c(0,1))

plot(x, dbinom(x, size=10, prob=0.5), type="h", ylim=c(0,1))
\texttt{plot(x, dbinom(x, size=10, prob=.8), type="h", ylim=c(0,1))}

\texttt{plot(x, dbinom(x, size=10, prob=.9), type="h", ylim=c(0,1))}
# How do the plots for \( p = .2 \) and \( .8 \) ?

# 4. Add the following code to the text processing pane and run it. What has been produced?
```
pr <- dbinom(x, size=10, prob=.2)
sum(x*pr)
## [1] 2
```

# 5. Add the following code to the text processing pane and run it. What has been produced?
```
sqdev <- (x - sum(x*pr))^2
sum(sqdev*pr)
## [1] 1.6
```

# 6. Add the following code to the text processing pane and run it. What has been plotted?
```
x <- 0:25
plot(x, dpois(x, lambda=2), type="h", ylim=c(0,1))
#7. Compare your plot from part 6 to "spike graphs"/"pin plots" for probabilities from the Poisson distributions with lambda 3, 4, and 9. What happens with increasing lambda?

x <- 0:25
plot(x, dpois(x, lambda = 3), type = "h", ylim = c(0, 1))
plot(x, dpois(x, lambda=4), type="h", ylim=c(0, 1))

plot(x, dpois(x, lambda=9), type="h", ylim=c(0, 1))
# The plot will move toward right with increasing Lambda.

# 8. Make and compare "spike graphs"/"pin plots" for binomial distribution probabilities with
# n = 100 and p=.16 to Poisson distribution probabilities for Lambda=16 over a set of possible values from 0 to 40. Adjust the limits on the vertical axes so that (they are the same for the two plots and) you can see clearly what is going on (don't run them all the way to 1.0). What is the general shape of these plots? How do they compare?

```r
x <- 0:40
plot(x, dbinom(x, size=100, prob=.16), type="h", ylim=c(0,1))
```
plot(x, dpois(x, lambda=16), type="h", ylim=c(0, 1))

# These two plots are almost the same. And both of them are normal like distribution.
#9. Make and compare plots of geometric distribution probabilities for \( p = .1 \), .2, .5, and .8 over possible values 1, 2, \( , 30 \). The second of these plots can be made using code below.

\[
\begin{align*}
\text{x} &\leftarrow 1:30 \\
\text{plot}(x, \text{dgeom}(x-1, p=.2), \text{type}="h", \text{ylim}=c(0,1))
\end{align*}
\]
The R version of the geometric distribution is for the number of "F"s before the first "S," that is, for $Y-1$ in the notation used in Vardeman and Jobe.
Hence the argument of the dgeom #function above was changed from x to x-1.

#10. Add the following code to the text processing pane and run it.

```r
x<-c(rbinom(1000, size=100, prob=.16))
hist(x)
```

![Histogram of x](image)

#They are both like normal distribution.
#This generates and makes a histogram for 1000 (pseudo-) random binomial variables. How does the histogram compare to the binomial probability pin plot in part 8?

#11. Rerun the code in part 10. Do you get the same plot? Should you expect the same plot?
#Why?

```r
x<-c(rbinom(1000, size=100, prob=.16))
hist(x)
```
```r
x <- c(rbinom(1000, size=100, prob=.16))
hist(x)
```
No, we can not get the same plot, since in each run, the random values are generated from different random number generators.

For purposes of having simulation results that can be reproduced and checked, it is important to be able to redo exactly the same generation of (pseudo-) random values. Add the line of code below to the code of part 10, and run it twice. Do you now get the same plot in these two runs?

```r
set.seed(0)
x <- c(rbinom(1000, size = 100, prob = .16))
hist(x)
```

![Histogram of x](image-url)

```r
x <- c(rbinom(1000, size = 100, prob = .16))
hist(x)
```
Set the seed of R's random number generator, which is useful for creating simulations or random objects that can be reproduced.

(You could have used any seed you like, not just the choice 0.)