Duncan McKinnon

West

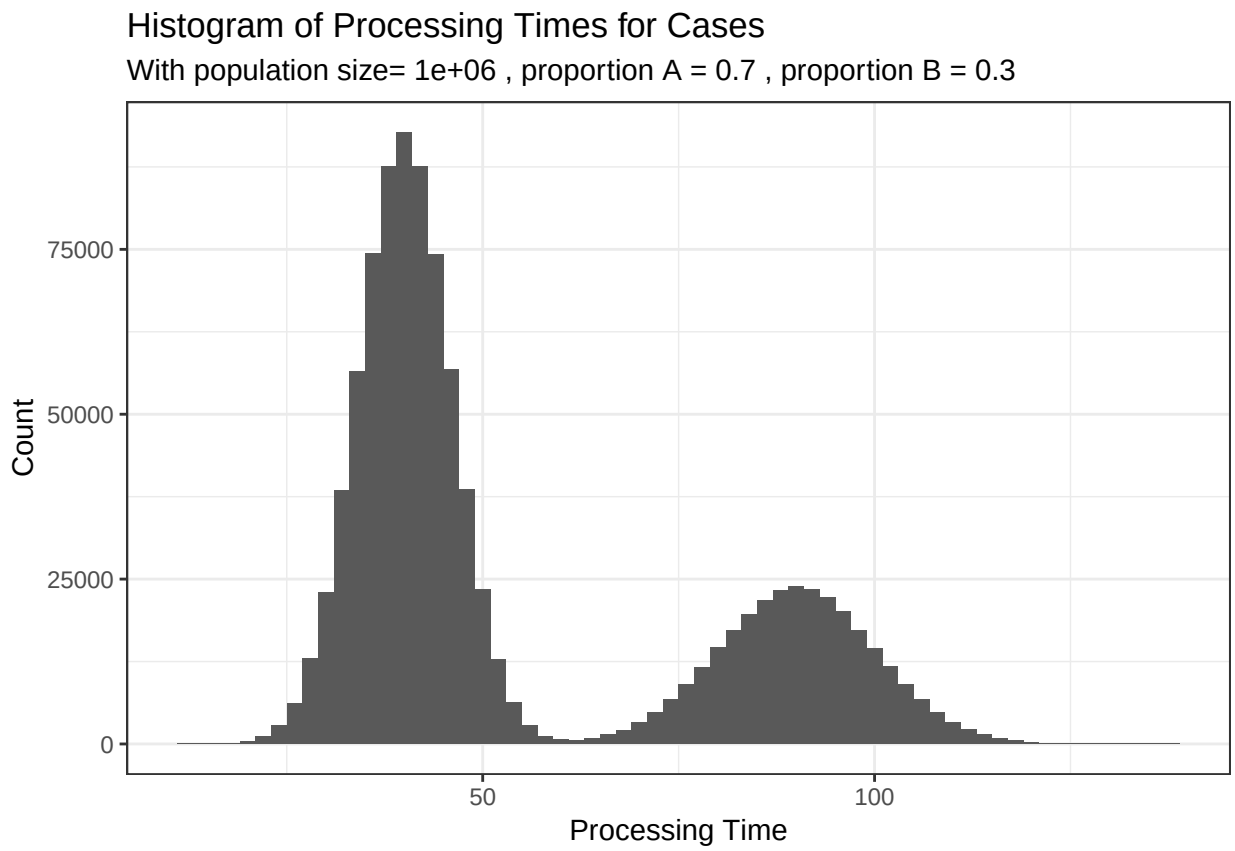
HW 5

QA

```
# load libraries and dependencies
suppressPackageStartupMessages( library(tidyverse) )
```

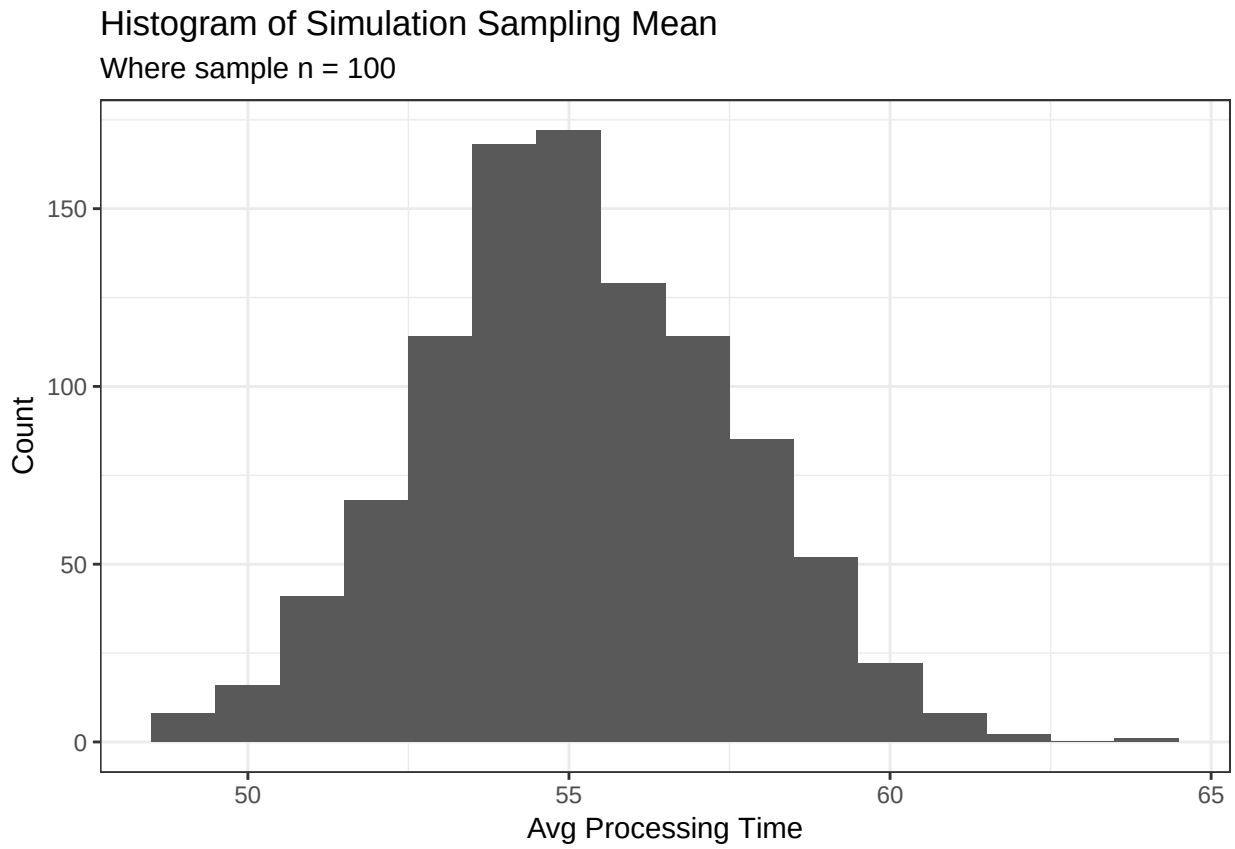
1).

```
# simulate population
set.seed(42) # set seed for replications

m <- 1e6
pA <- 0.7
pB <- 1 - pA

# A is normally distributed with mean 40, sd 6
A <- rnorm(m * pA, 40, 6)

# B is normally distributed with mean 90, sd 10
B <- rnorm(m * pB, 90, 10)

# population is made up of A and B
sim_pop <- c( A, B )

# histogram of simulated population
ggplot() +
  geom_histogram(aes(x = sim_pop), binwidth = 2) +
  labs( title = 'Histogram of Processing Times for Cases',
        subtitle = paste('With population size=', m, ', proportion A =', pA, ', proportion B =', pB),
        x = 'Processing Time',
        y = 'Count') +
  theme_bw()
```



2).

```
set.seed(42) # set seed for replications

# simulated sampling mean
m <- 1000
n <- 100

# collect samples and run means
samples <- replicate(m, sample_n(tibble(sim_pop), n))
sample_means <- map_dbl(samples, mean)

# histogram of simulated sampling mean
ggplot() +
  geom_histogram(aes(x = sample_means), binwidth = 1) +
  labs( title = 'Histogram of Simulation Sampling Mean',
        subtitle = paste('Where sample n =', n),
        x = 'Avg Processing Time',
        y = 'Count') +
  theme_bw()
```



3).

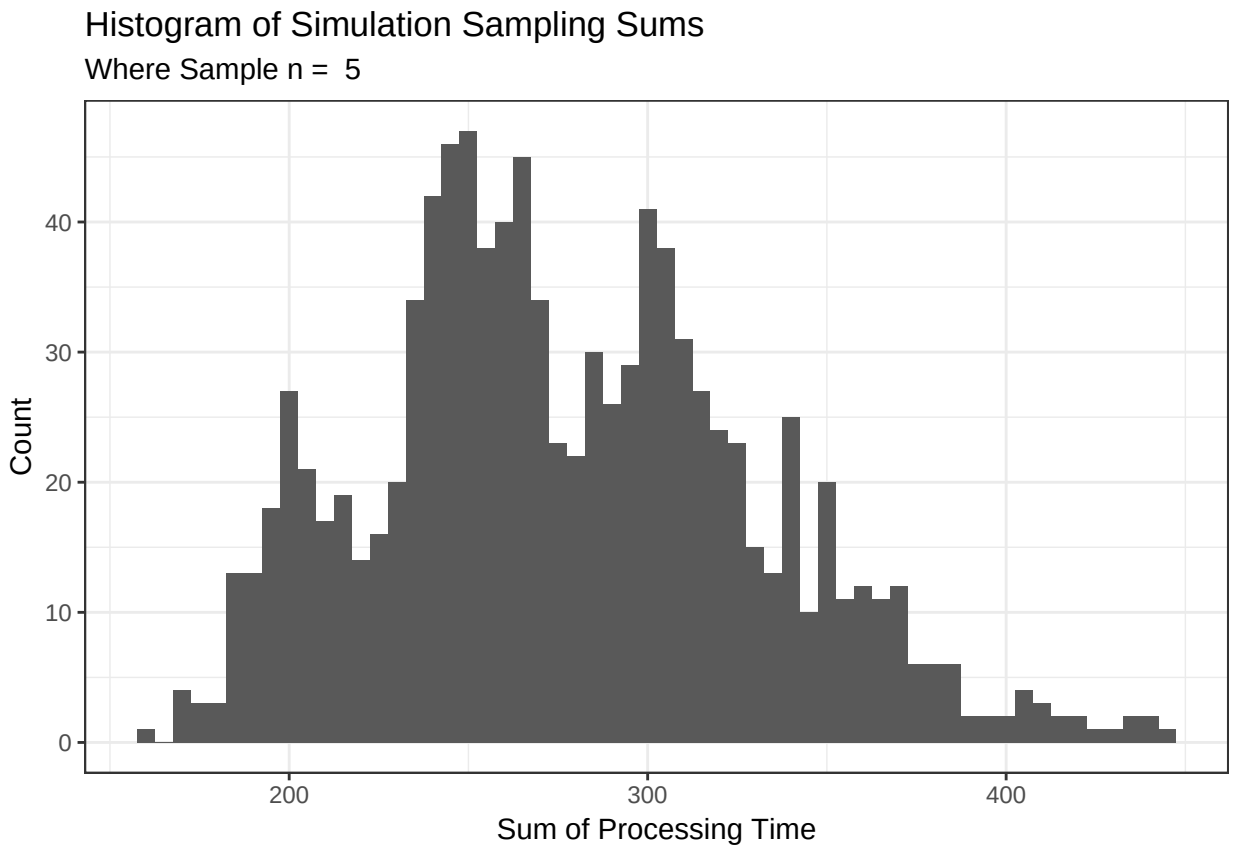
```
# standard error of the sample mean
se <- sd(sample_means)
se
```

```
## [1] 2.39964
```

4).

```
set.seed(42) # set seed for replications

# simulated sample sums
m <- 1000
n <- 5

# collect samples and get sums
sum_samples <- replicate(m, sample_n(tibble(sim_pop), n))
sample_sums <- map_dbl(sum_samples, sum)

# histogram of simulated sampling sums
ggplot() +
  geom_histogram(aes(x = sample_sums), binwidth = n) +
  labs( title = 'Histogram of Simulation Sampling Sums',
```

```
       subtitle = paste('Where Sample n = ', n),
       x = 'Sum of Processing Time',
       y = 'Count') +
  theme_bw()
```



5).

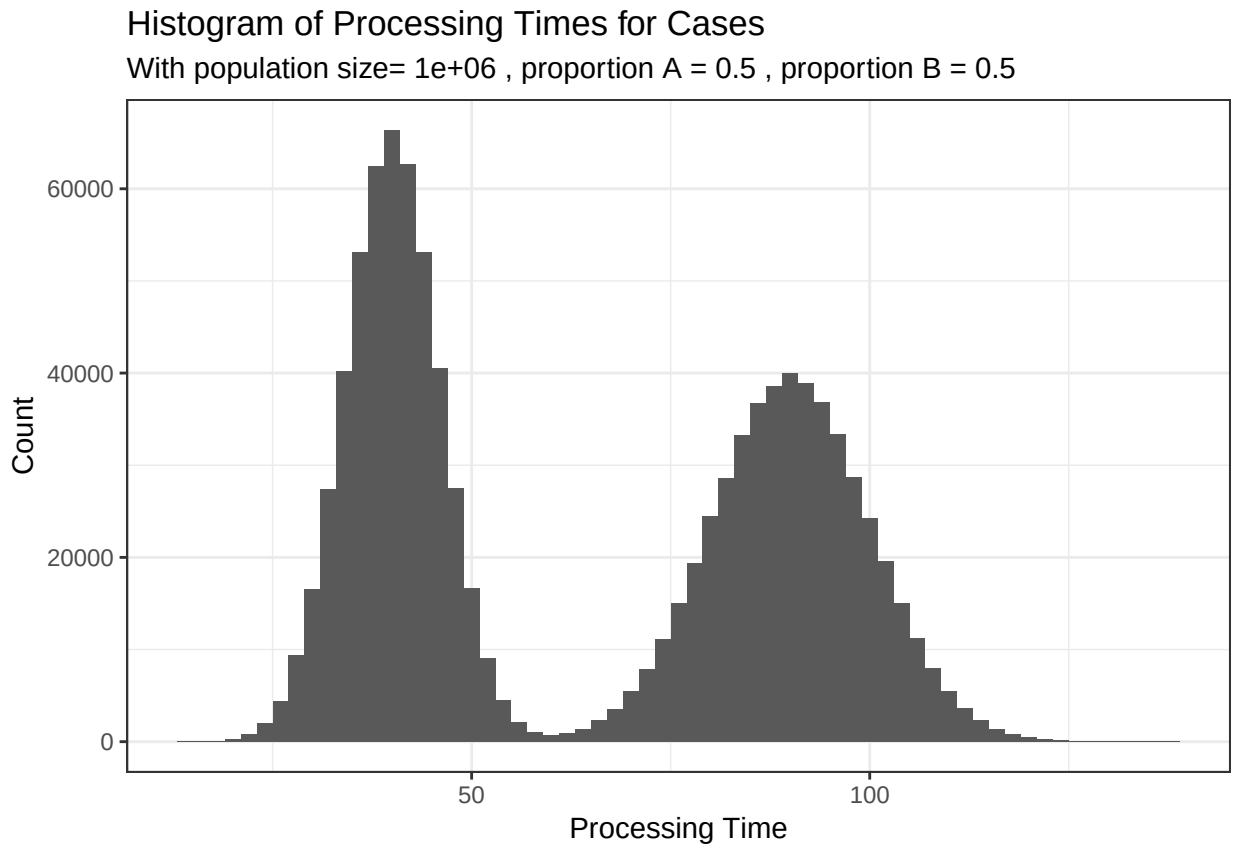
```
# P(sum > 8hrs)
P <- sum( sample_sums > 480 ) / m
P
```

```
## [1] 0
```

6).

```
# simulate population
set.seed(42) # set seed for replications

m <- 1e6
pA <- 0.5
pB <- 1 - pA

# A is normally distributed with mean 40, sd 6
A <- rnorm(m * pA, 40, 6)
```

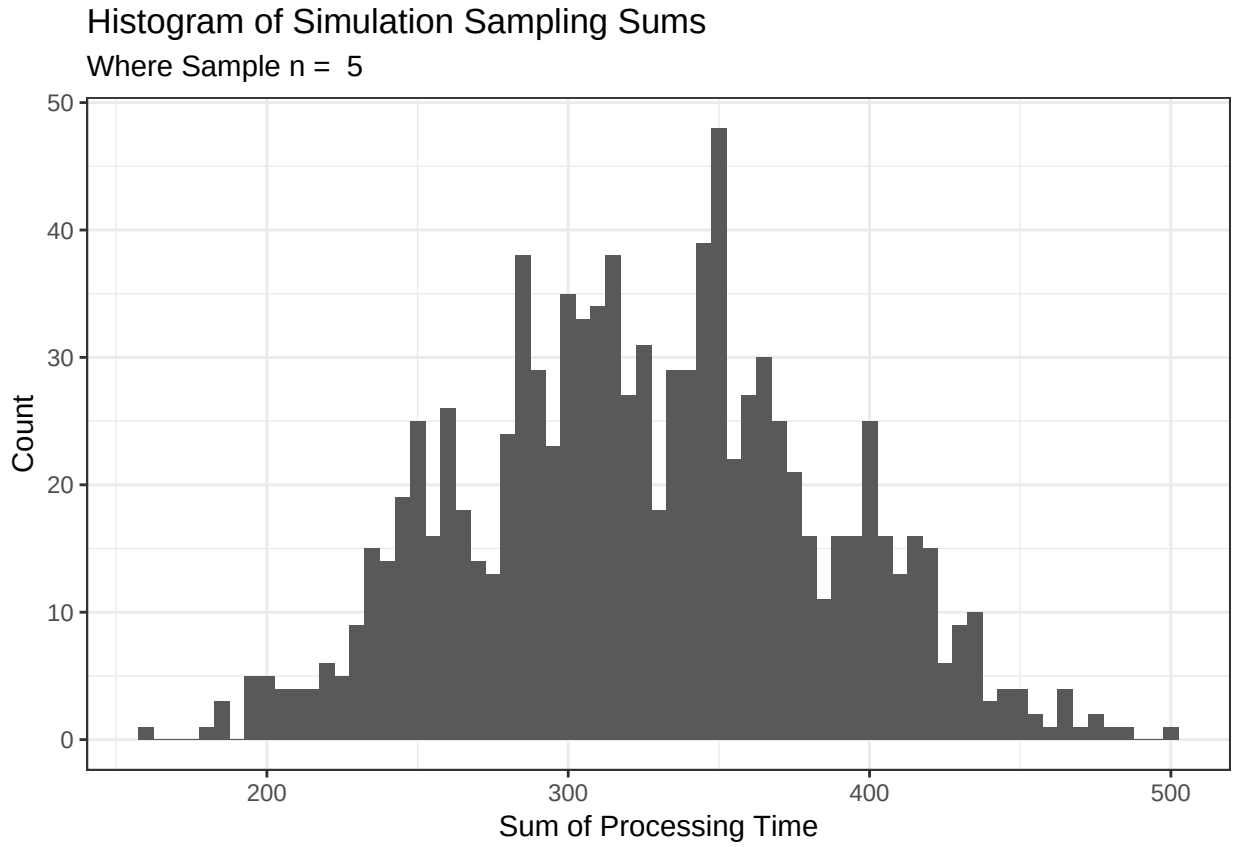
```r
# B is normally distributed with mean 90, sd 10
B <- rnorm(m * pB, 90, 10)

# population is made up of A and B
sim_pop <- c( A, B )

# histogram of simulated population
ggplot() +
  geom_histogram(aes(x = sim_pop), binwidth = 2) +
  labs( title = 'Histogram of Processing Times for Cases',
        subtitle = paste('With population size=', m, ', proportion A =', pA, ', proportion B =', pB),
        x = 'Processing Time',
        y = 'Count') +
  theme_bw()
```



7).

```r
set.seed(42) # set seed for replications

# simulated sample sums
m <- 1000
n <- 5

# collect samples and get sums
```

```r
sum_samples <- replicate(m, sample_n(tibble(sim_pop), n))
sample_sums <- map_dbl(sum_samples, sum)

# histogram of simulated sampling sums
ggplot() +
  geom_histogram(aes(x = sample_sums), binwidth = n) +
  labs( title = 'Histogram of Simulation Sampling Sums',
        subtitle = paste('Where Sample n = ', n),
        x = 'Sum of Processing Time',
        y = 'Count') +
  theme_bw()
```



8).

```r
# P(sum > 8hrs)
P <- sum( sample_sums > 480 ) / m
P
```

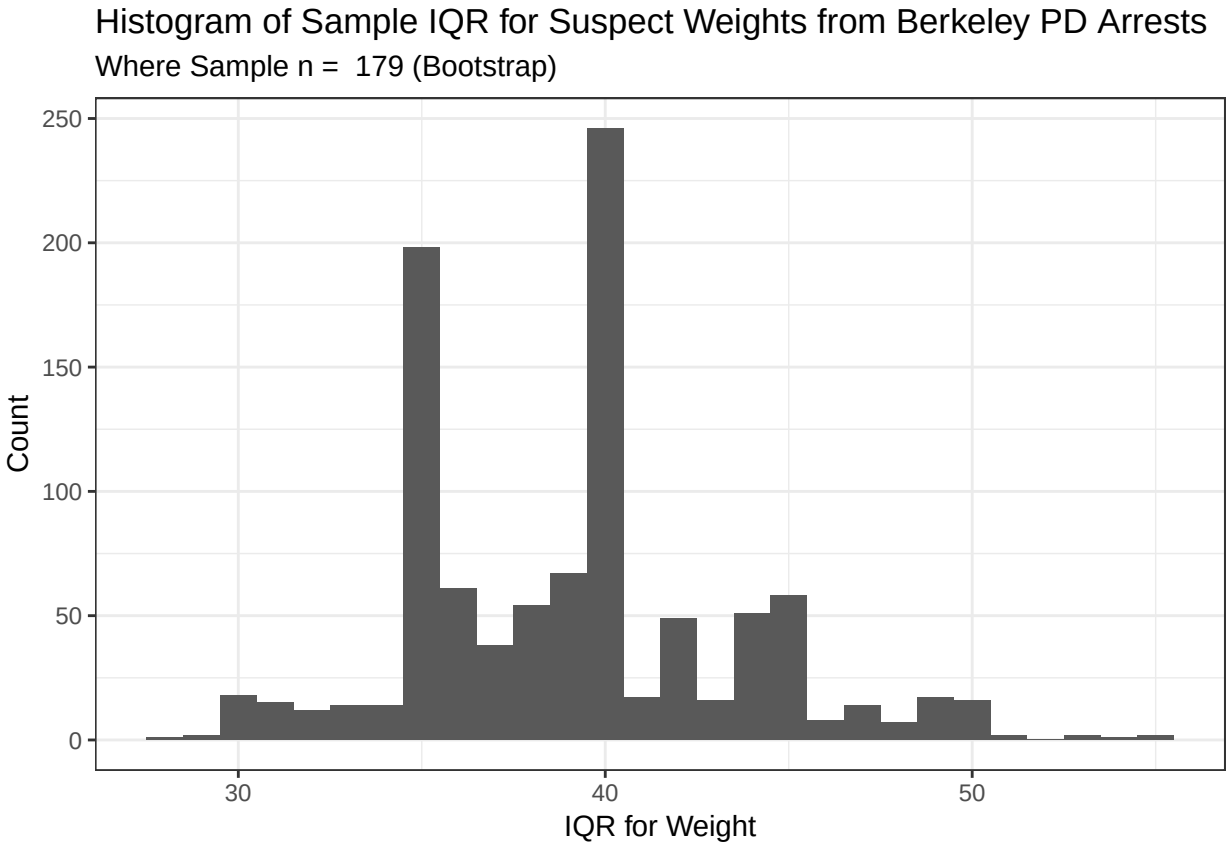
```
## [1] 0.002
```

QB

```r
# load libraries and dependencies
setwd("C:/Users/dunca/OneDrive/Desktop/Course Work/Intro to Data Science")
data <- read_csv("Data/Berkeley_PD_Log_-_Arrests.csv")
```

1).

```r
set.seed(42) # set seed for replications

# remove NA's from data
data <- data %>% filter(!is.na(Weight))

# create bootstrap distribution for sample IQR of body weight

m <- 1000
n <- length(data$Weight)

bs_samples <- replicate(m, sample_n(tibble(data$Weight), n, replace = TRUE))
bs_iqr <- map_dbl(bs_samples, IQR)

# histogram of bootstrap inter-quartile ranges
ggplot() +
  geom_histogram(aes(x = bs_iqr), binwidth = 1) +
  labs( title = 'Histogram of Sample IQR for Suspect Weights from Berkeley PD Arrests',
        subtitle = paste('Where Sample n = ', n, '(Bootstrap)'),
        x = 'IQR for Weight',
        y = 'Count') +
  theme_bw()
```



2).

```
# get 95% CI using standard error
CI_95p.SE <- mean(bs_iqr) + qnorm(c(0.025, 0.975)) * sd(bs_iqr)
names(CI_95p.SE) <- c('2.5%', '97.5%')
CI_95p.SE
```

```
##    2.5%   97.5%
## 30.5975 47.8695
```

3).

```
# get 95% CI using quantiles
CI_95p.q <- quantile(bs_iqr, c(0.025, 0.975))
CI_95p.q
```

```
##  2.5% 97.5%
##  31.0  49.5
```

4).

In this situation, the quantile method is a more appropriate means of acquiring the 95% confidence interval. For the standard error to produce a confidence interval that even reflects the SAMPLE data, the sample

must already be normally distributed (or at least consistent with the expected distribution). Looking at the distribution of the sample IQRs produced using bootstraping, we can see that the resulting distribution is far from normal. On the other hand, the quantile method will always capture the real endpoints for the confidence interval within the sample, regardless of the distribution of the statistic. It's also significant that in this situation, the IQR function is calculated using the quantile function rather than the standard error, so using the quantile is consistent with the precedent set when capturing the original statistic.

5).

The confidence interval created in question B represents the upper and lower bounds of the values which the interquartile range of the suspects weights may take on in the population, with 95% certainty. This means that we can say with 95% certainty that the difference of weights between suspects at the 75th percentile and the 25th percentile in the population distribution is between the lower value and the upper value of the interval we calculated.