Lec 8: Intro to STATA and R

1 What is STATA, and what is R?

		STATA	**R**
Commonalities		Tools used to perform statistical operations on computers. Available across platforms. Actively used by researchers across many disciplines.	
Differences	Nature of tool	Software	Programming language
	License	Proprietary	Open source
	Price	Costs money; UW has free access	Free
	Better for ...	Already cleaned data Little data manipulation	Data still needs to be cleaned Complex data manipulation
	Speed	Faster	OK

- You can pretty much achieve the same outcome by using either STATA or R, though the level of difficulty and the amount of time it would take differ between the two.
- I personally recommend:
  - Using R to do the data cleaning (i.e., getting data into the shape that you want, such as converting numbers coded as strings back to numerical formats, renaming columns / variable names, removing observations following certain criteria due to poor data collection, etc.)
  - Using STATA to obtain the statistics of interest.
- I don't expect you to become an expert in STATA and/or R by the end of today's lecture. If anything, our lecture today will only scratch the surface of what STATA and R can accomplish.
  - Some classes that you can take at UW:
    - SSCC training class: `https://sscc.wisc.edu/sscc_jsp/training/`
  - Online classes that you have access to:
    - SSCC online curriculum: `https://www.sscc.wisc.edu/statistics/training/`
    - LinkedIn Learning: `https://lnkd.in/eFrnwki`
      - Introduction to STATA 15
      - Learning R
      - R for Data Science: Lunch Break Lessons
  - What if you need help?
    - I will show you how to use the built-in help function in both STATA and R in today's lecture. Often times, this should be the first step to try when troubleshooting your code.
    - For STATA only, the company that made STATA offers instruction manual that contains some code examples. I will show you how to access such manual from the help function.
    - Google and Stack Overflow is always your friend.

2 Get ready

- Before proceeding with the rest of this worksheet (which will be us practicing using STATA and R together), you need to have STATA and R installed on your computer.
- Given that R itself is only a programming language, and we need some software to help us more easily run and troubleshoot our R code, we need to install an additional software called **R Studio**, which is an IDE (Integrated Development Environment).
  - If you need any help installing STATA, here's a quick installation guide: `https://go.wisc.edu/8crw4k`
  - If you need any help installing R and R Studio, here's a quick video: `https://youtu.be/3s57Swwoj-A`
- We will be working with a set of data that records college student's GPA. You'll need to first download this data onto somewhere on your computer. Remember where you saved the data.

  The data is available for download at the following link:

  `https://go.wisc.edu/q018t8`

  The data variable codebook is available here:

  `https://go.wisc.edu/tsy7p1`

3 Practice using STATA

1. Launch STATA. Let's go through the user interface together.
2. Create a new Do file, and use this Do file to run all of your commands below.
3. Before running any commands, let's set up the working directory to tell STATA which folder on your laptop should STATA read data and save graphs or results to.

   The easiest way to do so is to go to the menu bar, and select

   ```
   File > Change working directory...
   ```

   In the system file explorer window that popped up, navigate to the folder where you saved the GPA data, and then hit the `Choose` button.
4. We are now ready to import the data into STATA. Since the data file is named as `GPA.csv` , where the `csv` extension indicates that the data is formatted as Comma-Separated Values, we need to import the data into STATA environment as a delimited file.

   Since we don't really know what commands we should use to import such file, let's start utilizing the `help` function. Try the following command first:

   ```
   help import
   ```

   Upon examining the help document related to import commands, see if you can arrive at the final command that we need to import this file, which should be

   ```
   import delimited "GPA.csv", clear
   ```
5. Let's quickly browse the data set. Try the following two commands:

```
browse
describe
```

6. Find the mean, standard deviation, and 75th percentile of high school GPA:

   ```
   summarize hsgpa, detail
   ```

   You should see the following in the Results panel after running the command above:

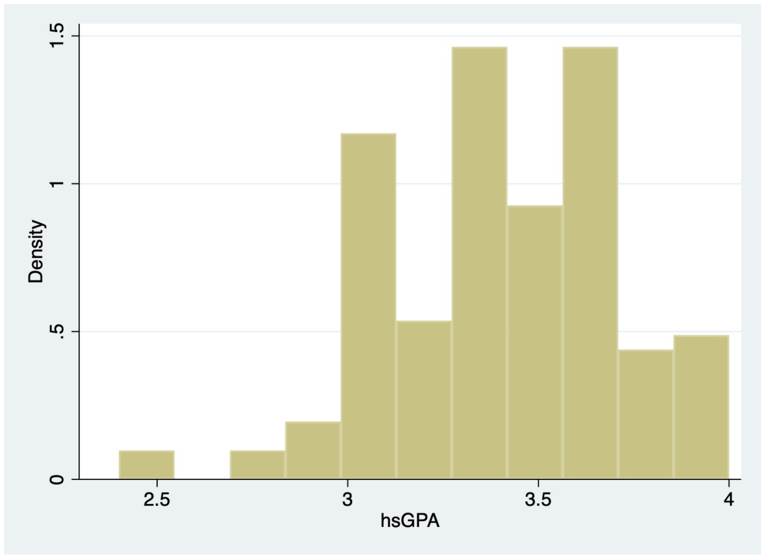
```
. summarize hsgpa, detail

                            hsGPA
-------------------------------------------------------------
      Percentiles      Smallest
 1%          2.5            2.4
 5%          2.9            2.5
10%            3            2.7       Obs                 141
25%          3.2            2.8       Sum of wgt.         141

50%          3.4                      Mean           3.402128
                        Largest       Std. dev.      .3199259
75%          3.6              4
90%          3.8              4       Variance       .1023526
95%          3.9              4       Skewness      -.3109566
99%            4              4       Kurtosis       2.890035
```

   This means that the mean of high school GPA is 3.402128, the standard deviation is 0.3199259, and the 75th percentile is 3.6.

7. Generate a histogram of high school GPA:

   ```
   histogram hsgpa
   ```

   Your histogram should look like the following:



8. Calculate the correlation between high school GPA and ACT score:

```
correlate hsgpa act
```

Interpret the correlation.

You should see the following in the Results panel:

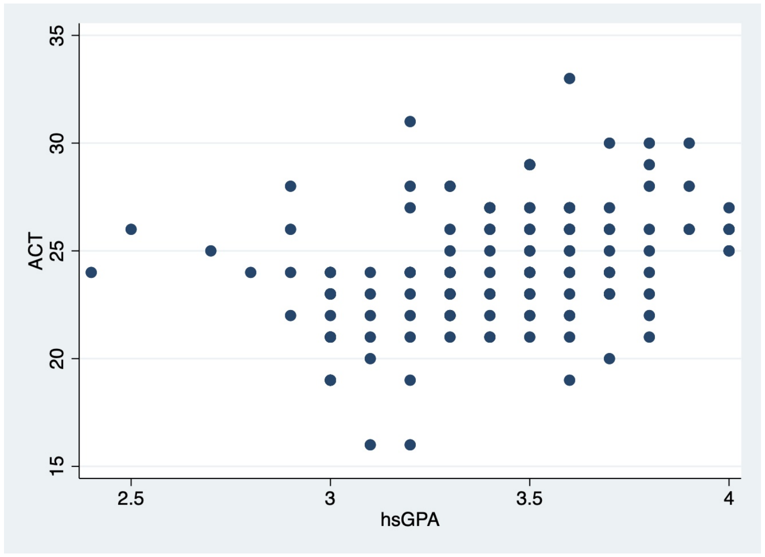
```
. correlate hsgpa act
(obs=141)

             |    hsgpa      act
-------------+------------------
       hsgpa |   1.0000
         act |   0.3458   1.0000
```

This means that the correlation between high school GPA and ACT score is 0.3458. This means that high school GPA and ACT score are weakly positively correlated.

9. Create a scatterplot of high school GPA and ACT score, with ACT score on the vertical axis:

```
scatter act hsgpa
```

Your scatterplot should look like the following:



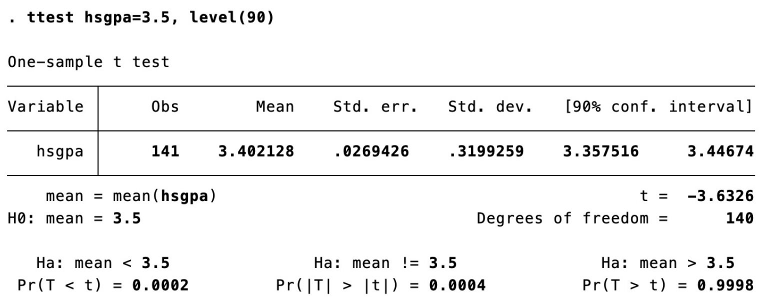
10. Perform t-test to test whether the population mean of high school GPA is equal to 3.5 using a 10% size of test:

```
ttest hsgpa=3.5, level(90)
```

Since we are performing a t-test on whether the population mean of high school GPA is equal to 3.5, our null and alternative hypothesis are the following:

$$H_0 : \mu = 3.5$$
$$H_1 : \mu \neq 3.5$$

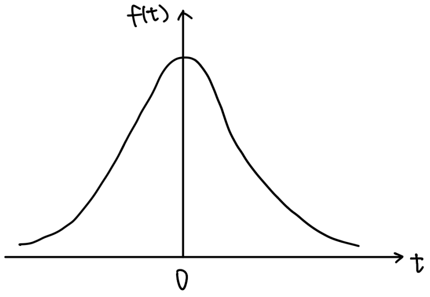
After running the STATA code provided above, you should see the following result:



```
. ttest hsgpa=3.5, level(90)

One-sample t test
------------------------------------------------------------------------------
Variable |     Obs        Mean    Std. err.   Std. dev.   [90% conf. interval]
---------+--------------------------------------------------------------------
   hsgpa |     141    3.402128    .0269426    .3199259    3.357516     3.44674
------------------------------------------------------------------------------
    mean = mean(hsgpa)                                            t =  -3.6326
H0: mean = 3.5                                   Degrees of freedom =      140

   Ha: mean < 3.5              Ha: mean != 3.5               Ha: mean > 3.5
 Pr(T < t) = 0.0002         Pr(|T| > |t|) = 0.0004          Pr(T > t) = 0.9998
```

With the result provided, we can interpret the t-test in three different ways:

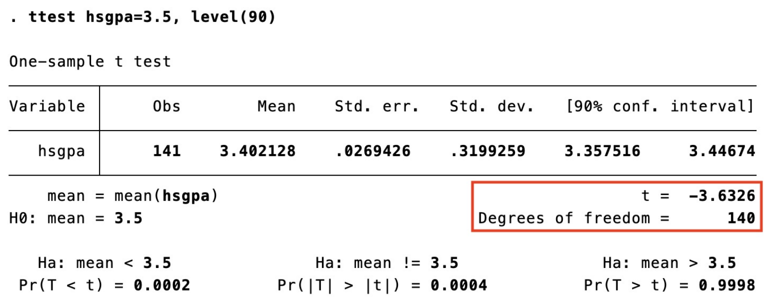
(a) **Test statistic & rejection region method**:
Taking H_0 as the default, we should have a sampling t-distribution, that looks like the following:



where the test statistic (i.e., the t-score) is calculated as the following:

$$t = \frac{\bar{X} - \mu_0}{s/\sqrt{n}} \sim t_{n-1}$$

the sampling t-distribution that t-score follows has degree of freedom equals to $n-1$.
Now, from the STATA result, we can find the t-score and the necessary degree of freedom in forming our rejection region:



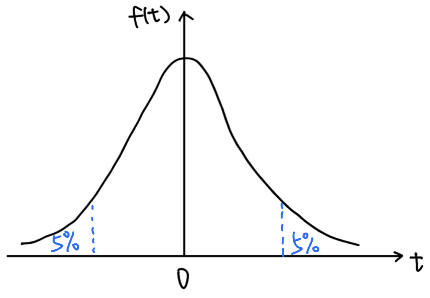
```
. ttest hsgpa=3.5, level(90)

One-sample t test
------------------------------------------------------------------------------
Variable |     Obs        Mean    Std. err.   Std. dev.   [90% conf. interval]
---------+--------------------------------------------------------------------
   hsgpa |     141    3.402128    .0269426    .3199259    3.357516     3.44674
------------------------------------------------------------------------------
    mean = mean(hsgpa)                                            t =  -3.6326
H0: mean = 3.5                                   Degrees of freedom =      140

   Ha: mean < 3.5              Ha: mean != 3.5               Ha: mean > 3.5
 Pr(T < t) = 0.0002         Pr(|T| > |t|) = 0.0004          Pr(T > t) = 0.9998
```

This means that the t-score is -3.6326, and the degree of freedom is 140.
Remember,

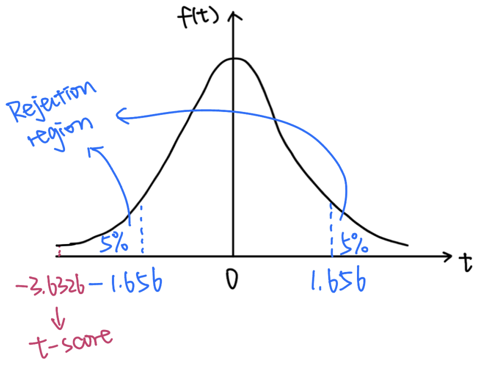
- The degree of freedom determines how the t-distribution looks like
- The t-score is about the position we are on the t-distribution for this specific sample
- If the t-score tells us that we are relatively close to the center of the t-distribution (i.e., outside of the rejection region formed with 10% size of the test), then we fail to reject the null, since our t-score tells us that we are not that far from the center of the t-distribution formed under the null hypothesis

- If the t-score tells us that we are relatively far from the center of the t-distribution (i.e., within the rejection region formed with 10% size of the test), then we reject the null, since our t-score tells us that we are very much far away from the center of the t-distribution formed under the null hypothesis

Since our alternative hypothesis, H_1, allows for inequality, this means that we will reject the null when we are far away in either the left tail or the right tail of the distribution. With a 10% size (i.e., 10% significance level), we should form equal size tail across the left side and the right side of the distribution:



Looking up t-table, we find that, with degree of freedom equal to 140, the left and right rejection region cutoff points being -1.656 and 1.656, respectively. This means that our t-score certain falls under the rejection region:



Thus, with 10% size of test, one rejects the null and conclude that the population mean of high school GPA is NOT equal to 3.5.

(b) **Confidence interval method**:

Recall that confidence interval is like the opposite of test statistic & rejection region method:

- Under test statistic & rejection region method, we end up constructing a size 10% rejection region, and see if the test statistic falls under the rejection region
- Under confidence interval method, we construct a $100\% - 10\% = 90\%$ confidence interval using the sample statistic to see if the hypothesized population mean falls within the confidence interval
  - If the hypothesized population mean falls within the confidence interval, then it is possible for us to have the hypothesized population mean. Thus, we FAIL to reject the null.
  - If the hypothesized population mean falls outside of the confidence interval, then the hypothesized population mean is NOT supported by this sample statistic. Thus, we reject the null.

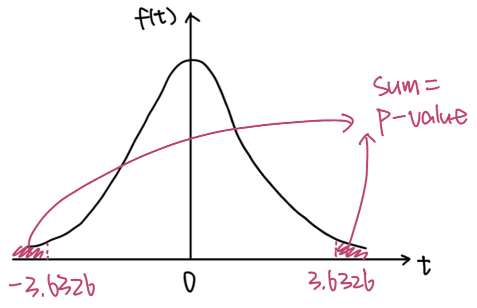
The 90% confidence interval can be found from the STATA code:

```
. ttest hsgpa=3.5, level(90)

One-sample t test
------------------------------------------------------------------------------
Variable |     Obs        Mean    Std. err.   Std. dev.   [90% conf. interval]
---------+--------------------------------------------------------------------
   hsgpa |     141    3.402128    .0269426    .3199259    3.357516     3.44674
------------------------------------------------------------------------------
    mean = mean(hsgpa)                                            t =  -3.6326
H0: mean = 3.5                                   Degrees of freedom =      140

   Ha: mean < 3.5              Ha: mean != 3.5               Ha: mean > 3.5
 Pr(T < t) = 0.0002         Pr(|T| > |t|) = 0.0004          Pr(T > t) = 0.9998
```

Thus, since the 90% confidence interval does NOT contain the hypothesized mean of 3.5, this means that we reject the null under 10% size, and conclude that the population mean of high school GPA is NOT equal to 3.5.

(c) **p-value method**:

The last method that we can perform this t-test is using p-value. Recall that p-value is constructed by using the test statistic (i.e., t-score) as if it is the appropriate cutoff for rejection:



In terms of rejection:

- If p-value is SMALLER than the size of the test, this means that our intended rejection region (constructed using size of the test) is bigger than the rejection region allowed by the t-score, so we reject the null.
- If p-value is BIGGER than the size of the test, this means that our intended rejection region (constructed using size of the test) is smaller than the rejection region allowed by the t-score, so we FAIL to reject the null.

Now, p-value for the test we want can be found from the STATA output:

```
. ttest hsgpa=3.5, level(90)

One-sample t test
------------------------------------------------------------------------------
Variable |     Obs        Mean    Std. err.   Std. dev.   [90% conf. interval]
---------+--------------------------------------------------------------------
   hsgpa |     141    3.402128    .0269426    .3199259    3.357516     3.44674
------------------------------------------------------------------------------
    mean = mean(hsgpa)                                            t =  -3.6326
H0: mean = 3.5                                   Degrees of freedom =      140

   Ha: mean < 3.5              Ha: mean != 3.5               Ha: mean > 3.5
 Pr(T < t) = 0.0002         Pr(|T| > |t|) = 0.0004          Pr(T > t) = 0.9998
```

Thus, p-value is 0.0004, and size of the test is $10\% = 0.10$. Clearly, $0.0004 < 0.10$, so we reject the null under 10% size and conclude that the population mean of high school GPA is NOT equal to 3.5.

4 Practice using R (working through the same exercises)

1. Launch R Studio. Let's go through the user interface together.
2. Create a new R script, and use this script to run all of your commands below.
3. Before running any codes, let's set up the working directory to tell R which folder on your laptop should R read data and save graphs or results to.

   The easiest way to do so is to go to the menu bar, and select

   `Session > Set Working Directory > Choose Directory...`

   In the system file explorer window that popped up, navigate to the folder where you saved the GPA data, and then hit the `Open` button.
4. We are now ready to import the data into R. Since the data file is named as `GPA.csv`, where the `csv` extension indicates that the data is formatted as Comma-Separated Values, we need to import the data into R environment as a delimited file.

   Here is where R differs significantly from STATA. STATA always works with datasets, so there is no question that STATA expects you to import a set of data.

   However, R is a programming language, so it can work with all sorts of data structure (dataset, vector, matrix, etc.). To import a data set into R, we need to make sure that it is imported as a data structure called `data.frame`.

   It is a little tricky to look up help command directly within R at this stage, but a quick Google should direct us to the following webpage:

   https://www.statmethods.net/input/importingdata.html

   From here, see if you can arrive at the final line of code to use, which should be something like

   ```
   df = read.table("GPA.csv", header=TRUE, sep=",")
   ```
5. Let's quickly browse the data set. Before doing so, I want to highlight another difference between STATA and R. STATA comes with a lot of built-in functions, since STATA is meant as a software packages.

   R, on the other hand, is an open-source language, which means that, though R comes with a lot of built-in capabilities, its power is limited until calling third-party packages that enhance R's capability.

   To help us browse the data more efficiently, I'd like for us to install such a third-party package called `psych`. To install and load this package into R's environment, try

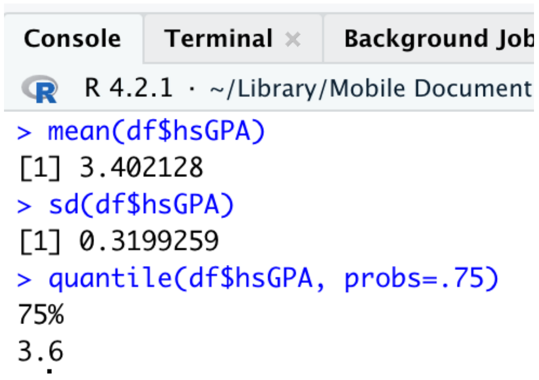
   ```
   install.packages("psych")
   library("psych")
   ```

   Let's now try the following two commands:

   ```
   View(df)
   describe(df)
   ```
6. Find the mean, standard deviation, and 75th percentile of high school GPA:

   ```
   mean(df$hsGPA)
   sd(df$hsGPA)
   quantile(df$hsGPA, probs=.75)
   ```

You should see the following in the Results panel after running the command above:



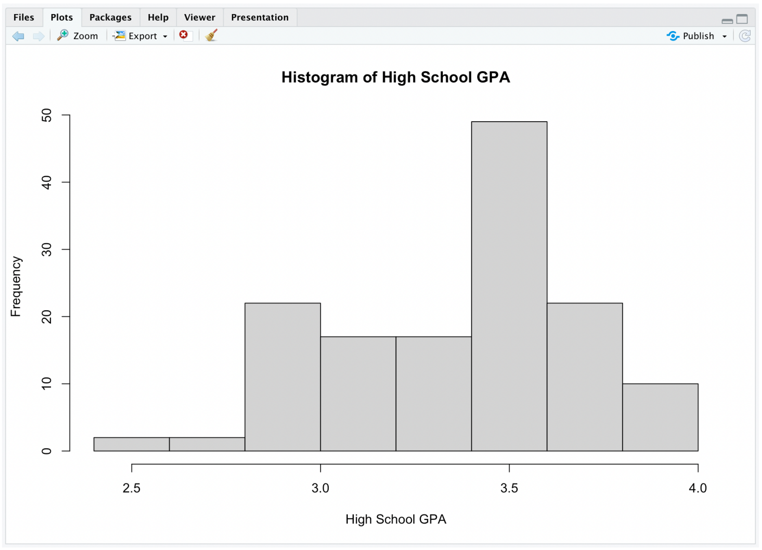
```
> mean(df$hsGPA)
[1] 3.402128
> sd(df$hsGPA)
[1] 0.3199259
> quantile(df$hsGPA, probs=.75)
75%
3.6
```

This means that the mean of high school GPA is 3.402128, the standard deviation is 0.3199259, and the 75th percentile is 3.6. All these numbers are identical to what we got from STATA.

7. Generate a histogram of high school GPA:

   `hist(df$hsGPA, xlab="High School GPA", main="Histogram of High School GPA")`

   You should see the following in the Plots panel:



   Notice that this histogram looks a bit different from the one generated by STATA, and that's OK. The difference here is the result of different bin selection. If you look closely, each bin in the R histogram is bigger than the bins in the STATA histogram.

   There are some customization on the number of bins that you can choose by including the `breaks` argument in R's `hist()` function, but that's not necessary at the end of the day. The histograms generated by both STATA and R sufficiently describe the general features of the data.

8. Calculate the correlation between high school GPA and ACT score:

   `cor(df$hsGPA, df$ACT)`

Interpret the correlation.

The following should appear in the Console after running the above code:

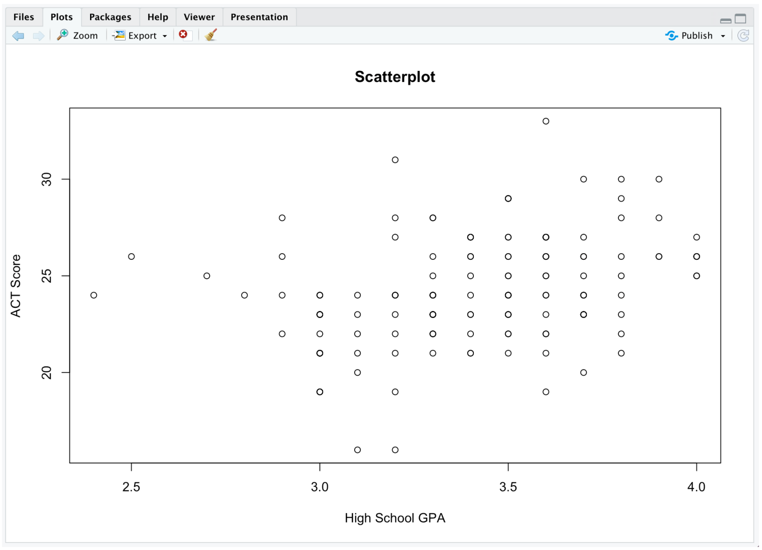
```
> cor(df$hsGPA, df$ACT)
[1] 0.3458056
```

This means that the correlation between high school GPA and ACT score is 0.3458056. This means that high school GPA and ACT score are weakly positively correlated.

9. Create a scatterplot of high school GPA and ACT score, with ACT score on the vertical axis:

   ```
   plot(df$hsGPA, df$ACT, xlab="High School GPA", ylab="ACT Score", main="Scatterplot")
   ```

   You should see the following in the Plots panel:



10. Perform t-test to test whether the population mean of high school GPA is equal to 3.5 using a 10% size of test:

    ```
    t.test(df$hsGPA, mu=3.5, conf.level=.90)
    ```

    Refer to the STATA section for all three different methods in performing the t-test. Here, the output in the R Console should look like the following:

```
> t.test(df$hsGPA, mu=3.5, conf.level=.90)

	One Sample t-test

data:  df$hsGPA
t = -3.6326, df = 140, p-value = 0.0003928
alternative hypothesis: true mean is not equal to 3.5
90 percent confidence interval:
 3.357516 3.446740
sample estimates:
mean of x
 3.402128
```

This means that

- t-score is -3.6326
- Degree of freedom (denoted as df in the output) is 140
- p-value is 0.0003928
- 90% confidence interval is $[3.357516, 3.446740]$

We will reach the same conclusion as in the STATA portion, which is that we reject the null under 10% size and conclude that the population mean of high school GPA is NOT equal to 3.5.