In [2]:
```
from datascience import *
import numpy as np

import matplotlib.pyplot as plots
plots.style.use('fivethirtyeight')
%matplotlib inline
```

Problem 1

Mr. Cho is a 6th-grade teacher with 14 girls and 11 boys in their class. Each day, they choose someone for milk duty, and an astute student notes that in the last 30 days, Mr. Cho has chosen 22 girls and 8 boys. They suspect Mr. Cho is biased against boys. Please test this with an appropriate hypothesis test.

What is your null hypothesis?

p_{girl} = 14/25

What is your test statistic?

proportion of the girls chosen in the 30 days.

Observed = 22/30

How will you simulate sampling under the null?

sample_proportions(number of samples I want to take, proportions of the actual data)

- sample_proportions(30, make_array(14/25, 11/25))

Please proceed to generate 5000 simulations under the null, determine a p-value, and draw a conclusion.

In [3]:
```
sample_stat = make_array()

for i in np.arange(5000):
    sample = sample_proportions(30, make_array(14/25, 11/25))
    girl_prop = sample.item(0)
    sample_stat = np.append(sample_stat, girl_prop)

p_value = sum(sample_stat > 22/30)/5000
p_value
```

Out[3]:
```
0.0166
```

Since the p-value is smaller than 0.05, we have an evidence to reject the null hypothesis. We can conclude that the probability for Mr. Cho to choose a student is not based on randomness.

Problem 2

Zitzapia is a new medication for the treatment of acne. The table acne.csv reports the number of new pimples a person experienced in one month while on the drug and how many new pimples while off the drug for 100 participants. Please conduct an appropriate hypothesis test to determine whether the drug reduces the incidence of pimples.

In [4]:
```
acne = Table().read_table('~/DS_113_S23/Labs/Review_Lab/acne.csv')
acne
```

Out[4]:

ID	On	Off
1	3	6
2	3	4
3	2	6
4	2	10
5	4	7
6	3	10
7	2	3
8	5	11
9	1	8
10	1	5

... (90 rows omitted)

What is your null hypothesis?

The number of new pimples increase and decrease were sampled at random whether or not the person used the drug.

What is your test statistic?

difference in average of the new pimples when the person is using drug and not.

How will you simulate sampling under the null?

avg(acne[1])-avg(acne[2])

Please proceed to generate 5000 simulations under the null, determine a p-value, and draw a conclusion.

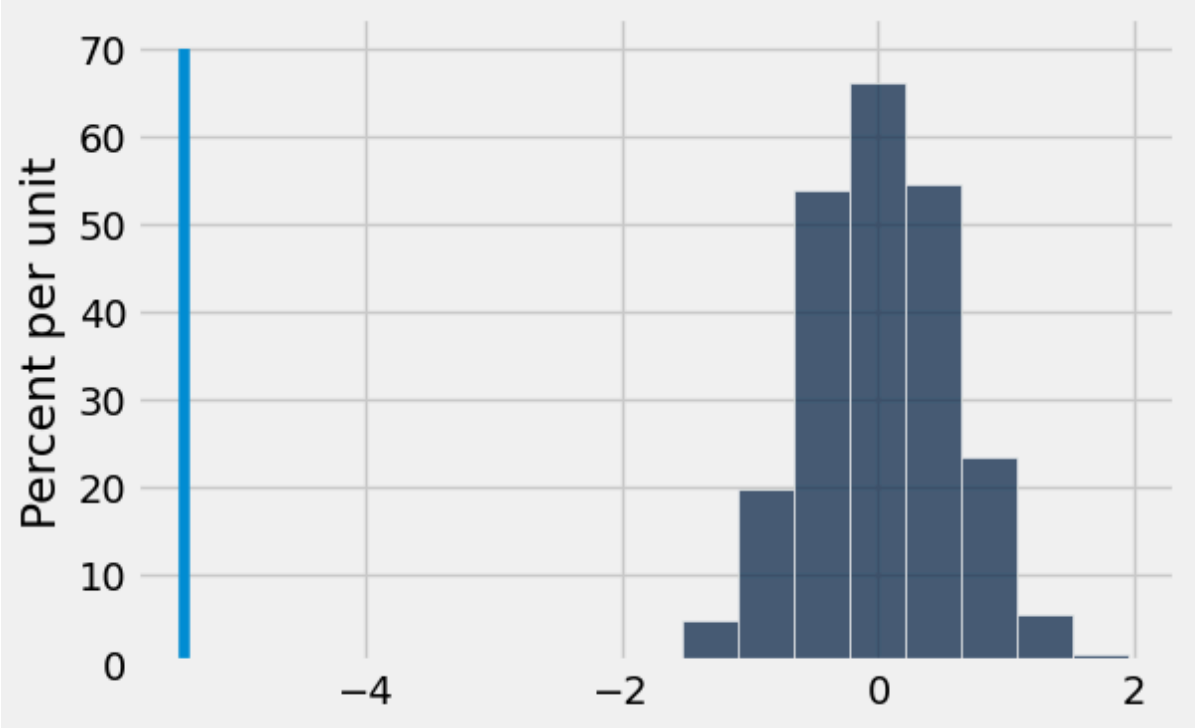
In [5]:
```
observed = np.average(acne[1])-np.average(acne[2])
observed
```

Out[5]:
```
-5.4199999999999999
```

In [6]:
```
sample_stat = make_array()
append_acne = np.concatenate([acne[1], acne[2]])

for i in np.arange(5000):
    shuffled = np.random.choice(append_acne, len(append_acne), replace=False
    on_drug_avg = np.average(shuffled[0:int(len(shuffled)/2)])
    off_drug_avg = np.average(shuffled[int(len(shuffled)/2):])
```

```
    sample_stat = np.append(sample_stat, on_drug_avg - off_drug_avg)

Table().with_column("", sample_stat).hist()
_ = plots.plot([observed, observed], [0, 0.7])

p_value = sum(sample_stat < observed)/5000
p_value
```

Out[6]: 0.0



The p-value is 0.0, meaning that nothing of the 5,000 permuted samples resulted in a difference of -5.42 or lower. According to our simulation, we have an evidence to reject the null hypothesis.

Problem 3

Now that you have determined that the drug zitzapia reduces incidents of acne, you would like to determine a confidence interval for the reduction in new pimples. Please report a 95% confidence interval for how many fewer pimples someone can expect when using the drug.

What is a bootstrap sample?

acne.sample()

What do you want to calculate from each bootstrap sample?

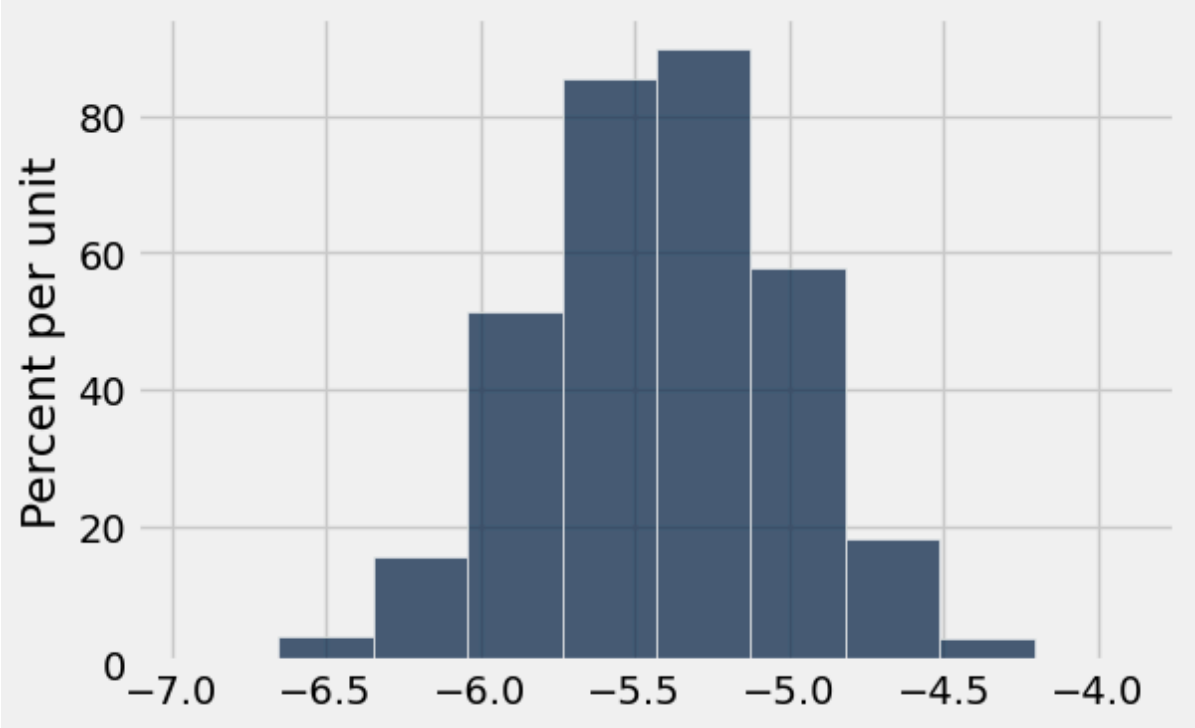
Answer here.

Please create 5000 bootstrap samples and represent the variability in our estimate for the reduction in the number of pimples with both a histogram and a confidence interval.

In [7]:
```
random_stat = make_array()
```

```
for i in np.arange(5000):
    bootstrap = acne.sample()
    on_drug_avg = np.average(bootstrap[1])
    off_drug_avg = np.average(bootstrap[2])
    random_stat = np.append(random_stat, on_drug_avg - off_drug_avg)

Table().with_column("", random_stat).hist()

c_lower_bound = percentile(2.5, random_stat)
c_upper_bound = percentile(97.5, random_stat)
print("Bootstrapped 95% confidence interval for the difference in average of
```

```
Bootstrapped 95% confidence interval for the difference in average of new a
cne with using drug and not in the population: [-6.260000, -4.630000]
```



Problem 4

The data set crickets.csv contains the air temperature and chirping rate for a sample of crickets on different summer nights.

In [8]:
```
crickets = Table().read_table('~/DS_113_S23/Labs/Review_Lab/crickets.csv')
crickets
```

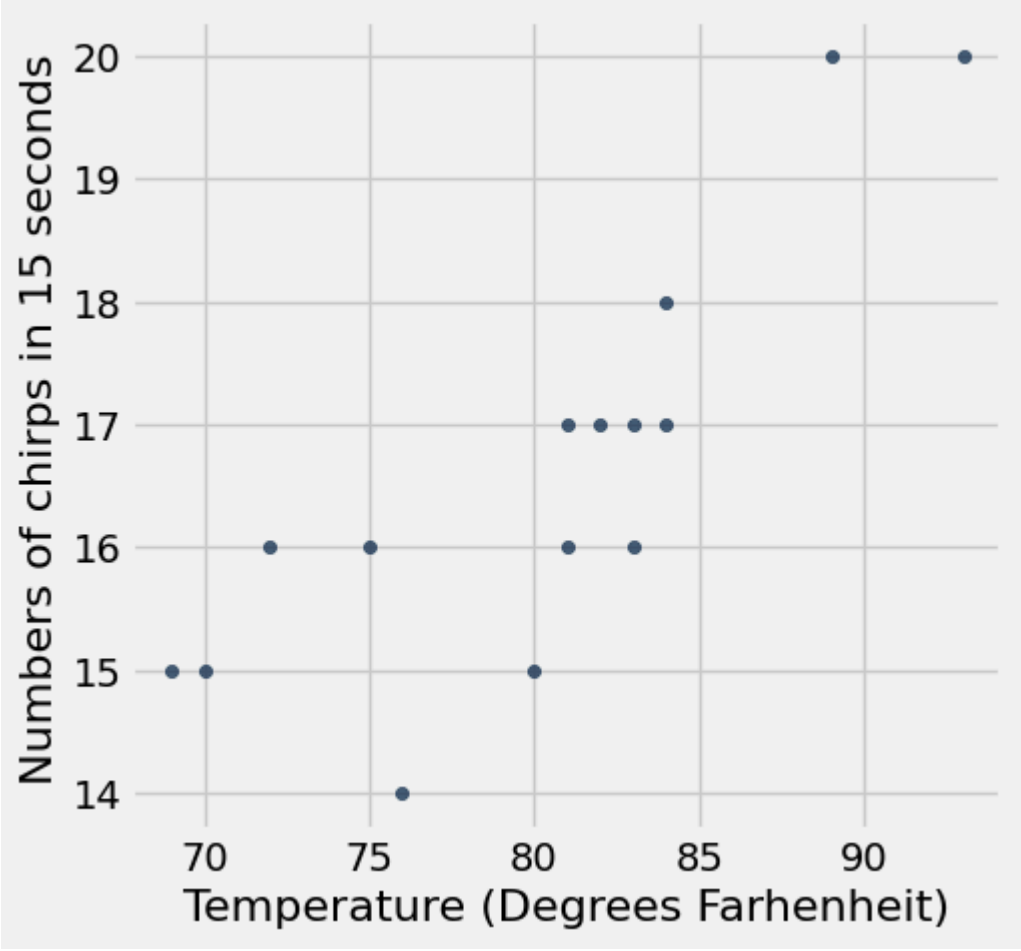
Out[8]:

Temperature (Degrees Farhenheit)	Numbers of chirps in 15 seconds
89	20
72	16
93	20
84	18
81	17
75	16
70	15
82	17
69	15
83	16

... (5 rows omitted)

Make a scatter plot of the data and interpret what you see.

In [9]:
```
crickets.scatter(0,1)
```



Calculate the correlation coefficient and explain how it is consistent with your scatter plot and observations above.

In [10]:
```
def standard_units(any_numbers):
    "Convert any array of numbers to standard units."
    return (any_numbers - np.mean(any_numbers)) / np.std(any_numbers)
```

```
def correlation(t, x, y):
    """Return the correlation coefficient (r) of two variables."""
    return np.mean(standard_units(t.column(x)) * standard_units(t.column(y))

def slope(t, x, y):
    """The slope of the regression line (original units)."""
    r = correlation(t, x, y)
    return r * np.std(t.column(y)) / np.std(t.column(x))

def intercept(t, x, y):
    """The intercept of the regression line (original units)."""
    return np.mean(t.column(y)) - slope(t, x, y) * np.mean(t.column(x))

def fit(table, x, y):
    """Return the height of the regression line at each x value."""
    a = slope(table, x, y)
    b = intercept(table, x, y)
    return a * table.column(x) + b
```

In [11]:
```
r = correlation(crickets, 0, 1)
r
```

Out[11]:
```
0.82535835436775895
```

Find the regression line equation and use it to predict the number of chirps in 15 second when the temperature is 77 degrees.

In [12]:
```
sl = slope(crickets, 0, 1)
it = intercept(crickets, 0, 1)
predict = 77*sl+it
predict
```

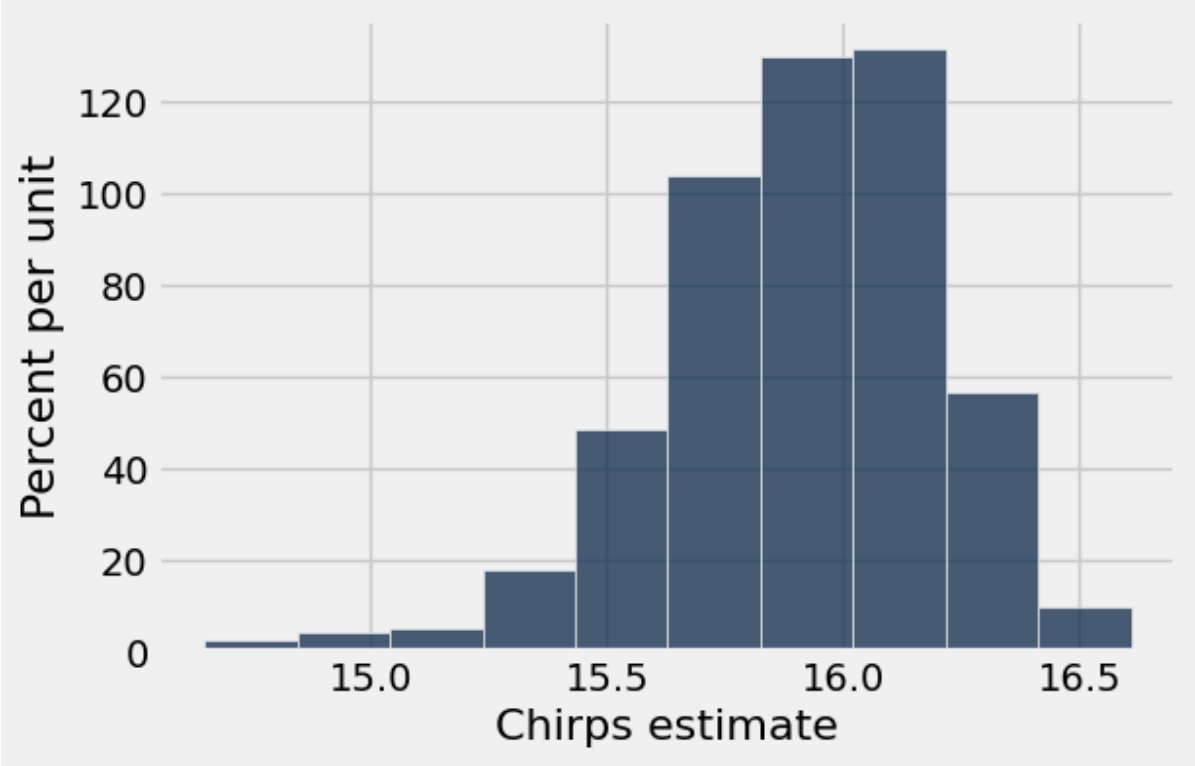
Out[12]:
```
15.936365555086535
```

Added challenge

Use the boot strap process to determine a confidence interval for the predicted number of chirps you found in the previous part.

In [13]:
```
bootstrap_chirps = make_array()
for i in np.arange(1000):
    boot_sample = crickets.sample()
    slopes = slope(boot_sample, 0, 1)
    intercepts = intercept(boot_sample, 0, 1)
    prediction = 77*slopes + intercepts
    bootstrap_chirps = np.append(bootstrap_chirps, prediction)

lower_end = percentile(2.5, bootstrap_chirps)
upper_end = percentile(97.5, bootstrap_chirps)

Table().with_column("Chirps estimate", bootstrap_chirps).hist()
print("95% confidence interval for the predicted number of chirps when the t
```

```
95% confidence interval for the predicted number of chirps when the tempera
ture is 77 degrees: [15.2419,16.3891]
```


Percent per unit
120
100
80
60
40
20
0
15.0
15.5
16.0
16.5
Chirps estimate