Unit Six Final Project

Reviewable Project

Capstone Option 2: Biodiversity for the National Parks To stay on track, submit by **MAR 3**

About 1 minute

For this project, you'll act as a data analyst for the National Park Service. You'll be helping them analyze data on endangered species from several different parks.

The National Parks Service would like you to perform some data analysis on the conservation statuses of these species and to investigate if there are any patterns or themes to the types of species that become endangered. During this project, you will analyze, clean up, and plot data, pose questions and seek to answer them in a meaning way.

After you perform analysis, you'll be creating a presentation to share your findings with the National Park Service.

TASK 1: ANALYZING DATA

There are two possible ways of completing the analysis for this project:

- You can complete the project on Codecademy.com . You won't need to install Python and you won't need to do anything on the Command Line.
- If you'd like an extra challenge, you can complete this project on your own computer. To do this, you'll need to install Python.

Reconciling

ON YOUR COMPUTER

Setting Up Python

1. If you've never used the command line, we recommend taking the Learn the Command Line course.
2. Install Python by following the directions in this article on Installing Python for Data Analysis.
3. Learn about Jupyter Notebooks, a cool way of combining Python code with explanations or instruction in a web terminal.

Analysis

1. Download biodiversity.zip
2. Double click on it to "unzip" the folder. It should contain several items: **observations.csv**, **species_info.csv**, **biodiversity.ipynb**, **biodiversity-solutions.ipynb**
3. In the command line, navigate into the biodiversity directory
4. Type the following into the command line:

```
jupyter notebook
```

5. This should open a browser tab. Click on **biodiversity.ipynb** in the browser tab. This will open up your Jupyter Notebook.
6. Follow the steps in the Jupyter Notebook. It will help you do your analysis.

ON CODECADEMY.COM

You will complete the exercises for the project on Codecademy.com. There's no need to download and install Python!

While you work through the exercises, take notes on any interesting quantities that you calculate. You'll need these later.

Also be sure to download any charts or graphs that you create. You can do this by right-clicking on the chart, and clicking "save as".

TASK 2: CREATE A SLIDE DECK

Once you've performed your analysis, either on your computer or on Codecademy, you're ready to create your slide deck.

Create a slide deck using Google Drive, Microsoft Powerpoint, or some other presentation software. Your presentation should include the following:

- A title slide
- A section describing the data in **species_info.csv**. Be sure to include some (or all) of what you noticed while working through the notebook.
- A section describing the significance calculations that you did for endangered status between different categories of species.
- A recommendation for conservationists concerned about endangered species, based on your significance calculations
- A section describing the sample size determination that you did for the foot and mouth disease study
- All of the graphs that you created in the notebook

If you like, you can also record a video of yourself giving the presentation and upload it to YouTube.

TASK 3: SUBMIT YOUR PROJECT

1. Create a folder containing the following files
2. Save your presentation as a PDF using "save as"
3. Save your iPython notebook as a ".py" file using File > Download as > Python (.py)
4. If you created a video, create a file called video.txt with a link to the video
5. Compress the file into a .zip
6. Upload it to Codecademy

Submit

Finished? Submit your project for personalized code review.

Review

We'll review your project and get back to you within 3-5 days. You'll receive an email when your review is ready.

ELECTIVE READING: **Github Issues**

Feedback

Leave us your feedback about your project review experience.

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Biodiversity Project

Welcome to the Introduction to Data Analysis Biodiversity Capstone project!

You are a biodiversity analyst working for the National Parks Service. You have been given a CSV file species_info.csv with data about different species in our National Parks, including:

- The scientific name of each species
- The common names of each species
- The species conservation status

(CSV data based on data from the National Parks Service.)

The National Parks Service would like you to perform some data analysis on the conservation statuses of these species and to investigate if there are any patterns or themes to the types of species that become endangered. During this project, you will analyze, clean up, and plot data, pose questions and seek to answer them in a meaning way.

Let's get started!

1. You'll need `pandas` and `matplotlib` for this project. Import both modules.

Hint

The standard is to import pandas as `pd` and import pyplot from matplotlibs as `plt`.

2. Load species_info.csv into a DataFrame called `species`.

Stuck? Get a hint

3. Print and inspect the DataFrame by using `.head()`.

   What kind of information is contained in this DataFrame?

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt
species = pd.read_csv('species_info.csv')
print species.head()
```

	category	scientific_name	common_names	conservation_status
0	Mammal	Clethrionomys gapperi gapperi	Gapper's Red-Backed Vole	nan
1	Mammal	Bos bison	American Bison, Bison	nan
2	Mammal	Bos taurus	Aurochs, Aurochs, Domestic Cattle (Feral), Domesticated Cattle	nan
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	nan
4	Mammal	Cervus elaphus	Wapiti Or Elk	nan

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Inspected the DataFrame

You've loaded the CSV into the Dataframe `species` and inspected it using `.head()`:

```
1    Mammel    Clethrionomys gapperi gapperi  Gapper's Red-Backed Vole
2    Mammel    Bos bison                      American Bison, Bison
3    Mammel    Bos taurus                     Aurochs, Aurochs, Domestic Ca
4    Mammel    Ovis aries                     Domestic Sheep, Mouflon, Red
...
```

Let's take a minute and learn a little about the data. Answer each of the following questions.

1. How many different species are in the `species` DataFrame?

   Save your answer to the variable `species_count`.

Hint

You can determine the number of unique entries in a DataFrame using `.nunique()`.

2. What are the different values of `category` in the DataFrame `species`?

   Save your answer to `species_type`.

Hint

You can call `.unique()` on a column to find all of its unique values.

3. What are the different values of `conservation_status`?

   Save your answer to `conservation_statuses`.

```python
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt
species = pd.read_csv('species_info.csv')
print species.head()
species_count = species.scientific_name.nunique()

# category = 7, scientific name = 5541

species_type = species.category.unique()

# species_type are 'Mammal' 'Bird' 'Reptile' 'Amphibian' 'Fish' 'Vascular
Plant'  'Nonvascular Plant'

conservation_statuses = species.conservation_status.unique()

# different values of conservation are nan 'Species of Concern' 'Endangered'
'Threatened' 'In Recovery'
```

Vascular Plant	Hypochaeris radicata	Cat's Ear, Spotted Cat's-Ear	
Vascular Plant	Hypochaeris radicata	Spotted Cats-Ear, Hairy Cats-Ear, Gosmore	
Vascular Plant	Hypochaeris radicata	Common Cat's-Ear, False Dandelion, Frogbit, Gosmore, Hairy Cat's Ear, Hairy Catsear, Spotted Catsear	

Mammal

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Analyze Species Conservation Status

Now it's time for some analysis!

You found that the column `conservation_status` has several possible values:

- `Species of Concern`: declining population or appears to be in need of conservation.
- `Threatened`: vulnerable to endangerment in the near future.
- `Endangered`: seriously at risk of extinction.
- `In Recovery`: formerly `Endangered`, but currently not in danger of extinction throughout all or a significant portion of its inhabitable range.

Now it would be interesting to count how many of each species fall into these conservation statuses.

1. Use `groupby` to count how many `scientific_name` falls into each `conversation_status` criteria.

   Save your results into the variable `conversation_counts`.

Hint

Remember, the syntax for doing this kind of groupby has the form `df.groupby('column_A').column_B.command().reset_index()`

2. Print `conversation_counts` and take a minute to think about what the DataFrame is telling you.

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt
species = pd.read_csv('species_info.csv')
print species.head()
species_count = species.scientific_name.nunique()

# category = 7, scientific name = 5541

species_type = species.category.unique()

# species_type are 'Mammal' 'Bird' 'Reptile' 'Amphibian' 'Fish' 'Vascular
Plant'  'Nonvascular Plant'

conservation_statuses = species.conservation_status.unique()
print conservation_statuses
# different types of conservation status are nan 'Species of Concern'
'Endangered' 'Threatened' 'In Recovery'

conservation_counts =
 species.groupby('conservation_status').scientific_name.nunique().reset_index()

# there are 151 species of concern, 10 threatened, 15 endangered, and 4 in
recovery
print conservation_counts
```

```
  category                scientific_name  \
0   Mammal  Clethrionomys gapperi gapperi
1   Mammal                      Bos bison
2   Mammal                     Bos taurus
3   Mammal                     Ovis aries
4   Mammal                  Cervus elaphus

                                      common_names conservation_status
0                         Gapper's Red-Backed Vole                  NaN
1                            American Bison, Bison                  NaN
2  Aurochs, Aurochs, Domestic Cattle (Feral), Dom...                  NaN
3  Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)                  NaN
4                                    Wapiti Or Elk                    NaN
[nan 'Species of Concern' 'Endangered' 'Threatened' 'In Recovery']
  conservation_status  scientific_name
0          Endangered               15
1         In Recovery                4
2  Species of Concern              151
3          Threatened               10
```

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Analyze Conservation Status II

When you counted the number of species that fall into our defined `conservation_status` buckets, you got the following:

	conservation_status	scientific_name
0	Endangered	15
1	In Recovery	4
2	Species of Concern	151
3	Threatened	10

But recall the initial `species` DataFrame, there are far more than 200 species! Clearly, only a small number of them are categorized as needing some sort of protection. The rest have `conservation_status` equal to None, or `NaN`. Because `groupby` does not include `NaN`, we will need to fill in the null values to get an accurate representation of the species conservation status.

We can do this uses `.fillna`, which is run on our DataFrame `species` and fills in all of the `NaN` values with an argument of our choice.

1. Paste the following code into the workspace and run it to replace `NaN` in our DataFrame with `'No Intervention'`:

```
species.fillna('No Intervention', inplace = True)
```

2. Great! Now run the same `groupby` as before to see how many species require `No Intervention`.

   Save your results into the variable `conversation_counts_fixed`.

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt
species = pd.read_csv('species_info.csv')
print species.head()
species_count = species.scientific_name.nunique()

# category = 7, scientific name = 5541

species_type = species.category.unique()

# species_type are 'Mammal' 'Bird' 'Reptile' 'Amphibian' 'Fish' 'Vascular
Plant'  'Nonvascular Plant'

conservation_statuses = species.conservation_status.unique()
print conservation_statuses
# different types of conservation status are nan 'Species of Concern'
'Endangered' 'Threatened' 'In Recovery'

conservation_counts =
 species.groupby('conservation_status').scientific_name.nunique().reset_index()

# there are 151 species of concern, 10 threatened, 15 endangered, and 4 in
recovery

species.fillna('No Intervention', inplace = True)
conservation_counts_fixed =
species.groupby('conservation_status').scientific_name.nunique().reset_index()

# there are 5,363 no interventions
print conservation_counts_fixed
```

```
  category                scientific_name  \
0   Mammal  Clethrionomys gapperi gapperi
1   Mammal                      Bos bison
2   Mammal                     Bos taurus
3   Mammal                     Ovis aries
4   Mammal                  Cervus elaphus

                                      common_names conservation_status
0                         Gapper's Red-Backed Vole                  NaN
1                            American Bison, Bison                  NaN
2  Aurochs, Aurochs, Domestic Cattle (Feral), Dom...                  NaN
3  Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)                  NaN
4                                     Wapiti Or Elk                  NaN
[nan 'Species of Concern' 'Endangered' 'Threatened' 'In Recovery']
  conservation_status  scientific_name
0          Endangered               15
1         In Recovery                4
2     No Intervention             5363
3  Species of Concern              151
4          Threatened               10
```

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Plotting Conservation Status by Species

We've determined the number of species that fall into each level of `conservation_status` and generated the following table:

	conservation_status	scientific_name
0	Endangered	15
1	In Recovery	4
2	No Intervention	5363
3	Species of Concern	151

Now let's make a visual of this data. We will use `plt.bar` to create a bar chart. Notice that the counts in the table above are in alpabetical order by `conservation_status` which is particularly helpful for visualizing the data, so our first step is to sort the columns by the number of species in each category.

We can do this using `.sort_values`. We use the keyword `by` to indicate which column we want to sort by.

1. Paste the following code and run it to create a new DataFrame called `protection_counts`, which is sorted by `scientific_name`:

```
protection_counts = species.groupby('conservation_status')\
    .scientific_name.nunique().reset_index()\
    .sort_values(by='scientific_name')
```

2. Now let's create a bar chart!

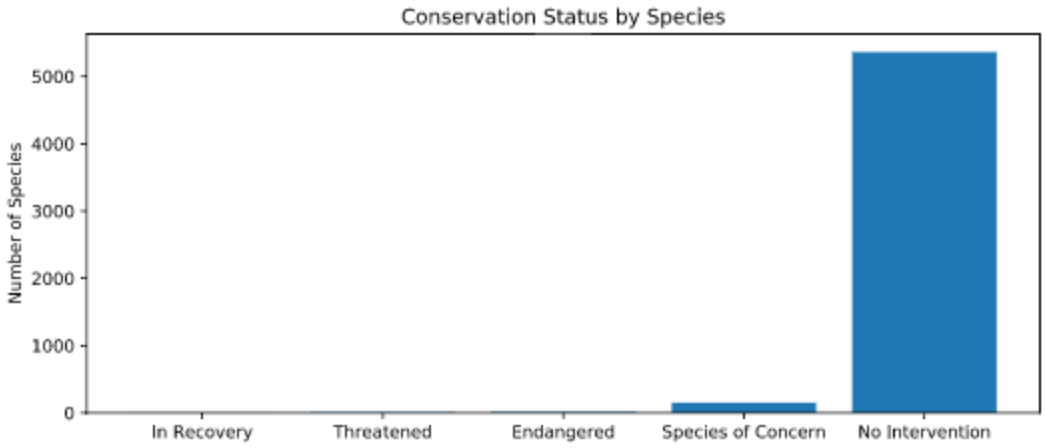
    1. Start by creating a wide figure with `figsize = (10,4)`
    2. Create an axes objected called `ax` using `plt.subplot`.
    3. Create a bar chart whose heights are equal to the `scientific_name` column of `protection_counts`.
    4. Create an x-tick for each of the bars.
    5. Label each x-tick with the label from `conservation_status` in `protection_counts`.
    6. Label the y-axis `Number of Species`.
    7. Title the graph `Conservation Status by Species`
    8. Plot the graph using `plt.show()`.

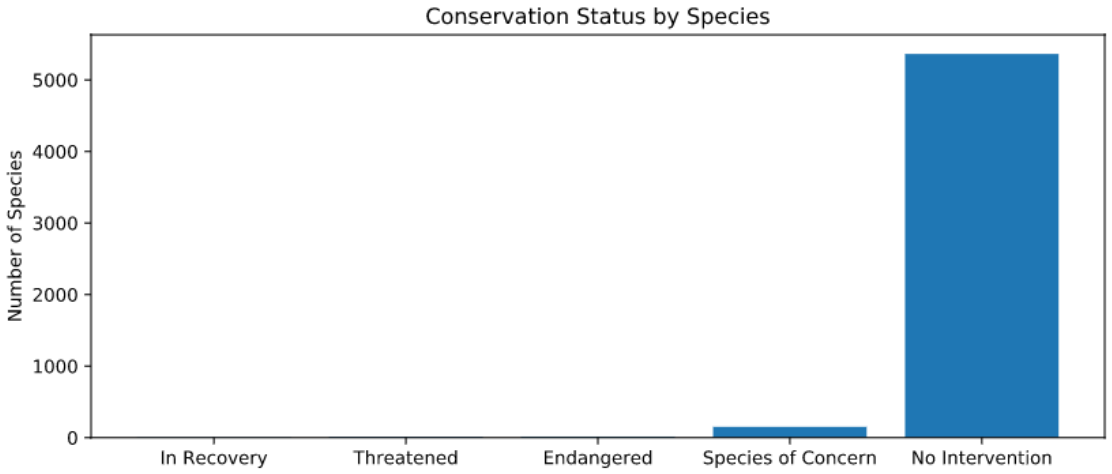
Hint

All of these can be accomplished by running commands on the plot object, `plt`, or the axes object, `ax` (once you have defined it). Feel free to search matplotlib documentation or return to the matplotlib lesson for a refresher.

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

species = pd.read_csv('species_info.csv')

species.fillna('No Intervention', inplace = True)

protection_counts = species.groupby('conservation_status').scientific_name.nunique().reset_index().sort_values(by='scientific_name')
print protection_counts

conservation_status = ['In Recovery', 'Threatened', 'Endangered', 'Species of Concern', 'No Intervention']
heights = [4, 10, 15, 151, 5363]
plt.figure(figsize = (10,4))
plt.bar(range(len(conservation_status)), heights)
ax = plt.subplot()
ax.set_xticks(range(len(conservation_status)))
ax.set_xticklabels(conservation_status)
plt.ylabel('Number of Species')
plt.title('Conservation Status by Species')
plt.show()
```

	conservation_status	scientific_name
1	In Recovery	4
4	Threatened	10
0	Endangered	15
3	Species of Concern	151
2	No Intervention	5363




Conservation Status by Species
Number of Species
0
1000
2000
3000
4000
5000
In Recovery
Threatened
Endangered
Species of Concern
No Intervention

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Investigating Endangered Species

Combing through the endangered species DataFrame raises an interesting question. **Are certain types of species more likely to be endangered?**

Let's investigate.

Instructions

1. Create a new column in `species` called `is_protected`, which is `True` if `conservation_status` is not equal to `'No Intervention'`, and `False` otherwise.

Hint

You can create a new column by running `df['new_column'] = *logical argument*`

2. Now group by *both* `category` and `is_protected`.

   Save your results to `category_counts`.

3. Examine `category_counts.head()`.

4. It's going to be easier to view this data if we pivot it. Using `pivot`, rearrange `category_counts` so that:

   - `columns` is `conservation_status`
   - `index` is `category`
   - `values` is `scientific_name`

   Save your pivoted data to `category_pivot`. Remeber to include `.reset_index()` at the end.

Hint

The syntax to pivoting takes the form `df.pivot()` where the arguments are setting columns equal to `columns`, `index`, and `values`.

5. Examine `category_pivot`.

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

species = pd.read_csv('species_info.csv')

species.fillna('No Intervention', inplace = True)

species['is_protected'] = species.conservation_status != 'No Intervention'
# do not understand line 9
category_counts = species.groupby(['category',
'is_protected']).scientific_name.nunique().reset_index()

print category_counts.head()

category_pivot = category_counts.pivot(columns='is_protected',
index='category', values='scientific_name').reset_index()
print category_pivot
```

```
    category  is_protected  scientific_name
0  Amphibian         False               72
1  Amphibian          True                7
2       Bird         False              413
3       Bird          True               75
4       Fish         False              115
is_protected          category  False  True
0                    Amphibian     72     7
1                         Bird    413    75
2                         Fish    115    11
3                       Mammal    146    30
4             Nonvascular Plant    328     5
5                      Reptile     73     5
6                 Vascular Plant   4216    46
```

species.conservation_status will take that whole column of conservation statuses, and will compare each of those conservation status values to 'No Intervention', and gives back a True or False based on the results of the comparison.
!= is an operator that checks if the values on either side are not equal. If the left value is not equal to the right side, it gives back a result of True. If the left value is equal to the right value, it gives back False. Does that make sense?

OK! Another possible way to accomplish this is using a lambda. It is a little bit more wordy/complicated, but it still achieves the same result. "species['is_protected'] = species['conservation_status'] .apply(lambda x: True if x != 'No Intervention' else False)"

	park_name	observations
0	Bryce National Park	250
1	Great Smoky Mountains National Park	149
2	Yellowstone National Park	507
3	Yosemite National Park	282

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Investigating Endangered Species II

You've just pivoted the species donservation data to make it more legible and now have the following table:

is_protected	category	False	True
0	Amphibian	72	7
1	Bird	413	75
2	Fish	115	11
3	Mammal	146	30

There are a couple more things you can do to make this table more readable AND more useful for addressing our guiding question, **are certain types of species more likely to be endangered?**.

☑ Instructions

1. `True` and `False` are pretty vague column names. Let's use the `.columns` to rename the categories `True` and `False` to something more descriptive:
   - Rename `False` to `not_protected`.
   - Rename `True` to `protected`.

Hint

An easy way to rename columns (if you know the order of the columns in your DataFrame) is `df.columns = ['column_A', 'column_B', 'column_C']`

- [x] 2. Let's create a new column in `category_pivot` called `percent_protected`, which is equal to `protected` (the number of species that are protected) divided by `protected` plus `not_protected` (the total number of species).

- [x] 3. Examine `category_pivot`.

  What does the new `percent_protected` column seem to indicate?

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

species = pd.read_csv('species_info.csv')

species.fillna('No Intervention', inplace = True)

species['is_protected'] = species.conservation_status != 'No Intervention'

category_counts = species.groupby(['category',
'is_protected']).scientific_name.nunique().reset_index()

print category_counts.head()

category_pivot = category_counts.pivot(columns='is_protected',
index='category', values='scientific_name').reset_index()

print category_pivot

category_pivot.info()

category_pivot.columns = ['category', 'not_protected', 'protected']
category_pivot['percent_protected'] = (category_pivot.protected /
(category_pivot.protected + category_pivot.not_protected)) * 100
print category_pivot
```

```
    category  is_protected  scientific_name
0  Amphibian         False               72
1  Amphibian          True                7
2       Bird         False              413
3       Bird          True               75
4       Fish         False              115
is_protected           category  False  True
0                     Amphibian     72     7
1                          Bird    413    75
2                          Fish    115    11
3                        Mammal    146    30
4             Nonvascular Plant    328     5
5                       Reptile     73     5
6                 Vascular Plant   4216    46
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7 entries, 0 to 6
Data columns (total 3 columns):
category    7 non-null object
False       7 non-null int64
True        7 non-null int64
dtypes: int64(2), object(1)
memory usage: 240.0+ bytes
            category  not_protected  protected  percent_protected
0          Amphibian             72          7           8.860759
1               Bird            413         75          15.368852
2               Fish            115         11           8.730159
3             Mammal            146         30          17.045455
4  Nonvascular Plant            328          5           1.501502
5            Reptile             73          5           6.410256
6     Vascular Plant           4216         46           1.079305
```

	category	not_protected	protected	percent_protected
0	Amphibian	72	7	8.860759
1	Bird	413	75	15.368852
2	Fish	115	11	8.730159
3	Mammal	146	30	17.045455
4	Nonvascular Plant	328	5	1.501502
5	Reptile	73	5	6.410256
6	Vascular Plant	4216	46	1.079305

	category	not_protected	protected	percent_protected
0	Amphibian	72	7	0.088608
1	Bird	413	75	0.153689
2	Fish	115	11	0.087302
3	Mammal	146	30	0.170455
4	Nonvascular Plant	328	5	0.015015
5	Reptile	73	5	0.064103
6	Vascular Plant	4216	46	0.010793

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Chi-Squared Test for Significance

You now have the endangered species data pivoted in the table below. Let's see if we can use it to answer the question "are certain types of species more likely to be endangered?".

2	Fish	115	11	0.087302
3	Mammal	146	30	0.170455
4	Nonvascular Plant	328	5	0.015015
5	Reptile	73	5	0.064103
6	Vascular Plant	4216	46	0.010793

It looks like Mammals are more likely to be endangered than Birds, but is it a significant difference? We can do a significance test to see if this statement is true. In this test, our **null hypothesis** is that this difference is due to chance.

But what kind of test are you going to use? Consider the following questions:

- Is the data numerical or categorical?
- How many pieces of data are you comparing?

Based on those answers, you should choose to do a *chi-squared test*. In order to run a chi-squared test, we'll need to create a contingency table. Our contingency table will have the form:

	protected	not-protected
Mammal	?	?
Bird	?	?

Instructions

1. Create a table called `contingency` and fill it with the correct values. You do not need to include column names in the contingency table.

2. In order to perform our chi-squared test, we'll need to import the correct function from `scipy`. Paste the following code and run it:

```
from scipy.stats import chi2_contingency
```

3. Run `chi2_contingency` on the `contingency` table.

   Save the p-value from this test to the variable `pval`.

Hint

Remember, `chi2_contingecy` returns a 4 element tuple, where the second element is the `p-value`.

4. It looks like this difference isn't significant!

   Let's test another. Is the difference between `Reptile` and `Mammal` significant?

   Save the p-value to `pval_reptile_mammal`.

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

contingency = [[30,146],[75,413]]
from scipy.stats import chi2_contingency
pval = chi2, pval, dof, expected = chi2_contingency(contingency)
print pval

contingency2 = [[5,73],[30,146]]
pval_reptile_mammal = chi2, pval_reptile_mammal, dof, expected =
chi2_contingency(contingency2)
print pval_reptile_mammal
```

```
0.687594809666
0.0383555902297
```

.68 means that we cannot reject the null hypothesis (or we cannot say there is a significant difference).

.03 is < .05 so we reject the null hypotheses and we say there is a significant difference.

A significantly large difference will allow us to reject the **null hypothesis**, which is defined as the prediction that there is no interaction between variables. Basically, if there is a big enough difference between the scores, then we can say something significant happened. If the scores are too close, then we have to conclude that they are basically the same.

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Final Thoughts on Protected Species

Now we can answer our initial question:

Are certain types of species more likely to be endangered?

We initially saw that there was a slight difference in the percentages of birds and mammals that fall into a protected category. Our **null hypothesis** here is that this difference was a result of chance.

When we ran our chi-squared test, we found a p-value of ~0.688, so we can conclude that the difference between the percentages of protected birds and mammals is not significant and is a result of chance.

But, when we compared the percentages of protected reptiles and mammals and ran the same chi-squared test, we calculated a p-value of ~0.038, which **is** significant.

Therefore, we can conclude that certain types of species *are* more likely to be endangered than others.

Congratulations on completing Part I of the Biodiversity at National Parks Data Analysis Project!

Please feel free to continue to play around with the data and see what other interesting questions and conclusions can be drawn. When you are ready, move onto the next exercise where we will tackle another question.

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Observations DataFrame

The National Parks Service sent over another dataset for you to analyze.

Conservationists have been recording sightings of different species at several national parks for the past 7 days. Their observations have been sent to you in a file called observations.csv.

☑ Instructions

- [x] 1. Load `observations.csv` into a variable called `observations`.
- [x] 2. Inspect the first couple rows of `observations` by printing it and using `.head()`.

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

species = pd.read_csv('species_info.csv')
species.fillna('No Intervention', inplace = True)
species['is_protected'] = species.conservation_status != 'No Intervention'

observations = pd.read_csv('observations.csv')
print observations.head()
```

	scientific_name	park_name	observations
0	Vicia benghalensis	Great Smoky Mountains National Park	68
1	Neovison vison	Great Smoky Mountains National Park	77
2	Prunus subcordata	Yosemite National Park	138
3	Abutilon theophrasti	Bryce National Park	84
4	Githopsis specularioides	Great Smoky Mountains National Park	85

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

In Search of Sheep

A team of ruminant-enthused scientists has been tracking the movements of various species of sheep across different national parks and have asked for your assistance in analyzing the `observation` and `species` DataFrames to help track sheep locations.

Because the `observation` DataFrame only contains the scientific names of species, you will have to use the `species` DataFrame to look for any names that refer to sheep.

The following code will tell us whether or not a word, such as "sheep", occurs in a string:

```
>>> # Does "Sheep" occur in this string?
>>> str1 = 'This string contains Sheep, baa'
>>> 'Sheep' in str1

True
```

```
>>>  # Does "Sheep" occur in this string?
>>> str2 = 'This string contains Cows, moo'
>>> 'Sheep' in str2

False
```

1. Use `apply` and a `lambda` function to create a new column in `species` called `is_sheep` which is `True` if the `common_names` contains `'Sheep'`, and `False` otherwise.

2. Select the rows of `species` where `is_sheep` is `True` and save it to the variable `species_is_sheep`.

3. Print `species_is_sheep` and inspect the results. You will want to browse through all the entries so don't use `.head()`.

4. Many of the results are actually plants. Select the rows of `species` where `is_sheep` is `True` and `category` is `Mammal`. Save the results to the variable `sheep_species`.

5. Print and inspect `sheep_species`.

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

species = pd.read_csv('species_info.csv')
species.fillna('No Intervention', inplace = True)
species['is_protected'] = species.conservation_status != 'No Intervention'
print species.head()
observations = pd.read_csv('observations.csv')
print observations.head()

species['is_sheep'] = species.common_names.apply(lambda x: True if 'Sheep' in x else False)
print species.head()
species_is_sheep = species[species.is_sheep == True]
print species_is_sheep
sheep_species = species[(species.is_sheep == True) & (species.category == 'Mammal')]
print sheep_species
```

	category	scientific_name	common_names	conservation_status	is_protected
0	Mammal	Clethrionomys gapperi gapperi	Gapper's Red-Backed Vole	No Intervention	False
1	Mammal	Bos bison	American Bison, Bison	No Intervention	False
2	Mammal	Bos taurus	Aurochs, Aurochs, Domestic Cattle (Feral), Domesticated Cattle	No Intervention	False
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False
4	Mammal	Cervus elaphus	Wapiti Or Elk	No Intervention	False

	scientific_name	park_name	observations
0	Vicia benghalensis	Great Smoky Mountains National Park	68
1	Neovison vison	Great Smoky Mountains National Park	77
2	Prunus subcordata	Yosemite National Park	138
3	Abutilon theophrasti	Bryce National Park	84
4	Githopsis specularioides	Great Smoky Mountains National Park	85

	category	scientific_name	common_names	conservation_status	is_protected	is_sheep
0	Mammal	Clethrionomys gapperi gapperi	Gapper's Red-Backed Vole	No Intervention	False	False
1	Mammal	Bos bison	American Bison, Bison	No Intervention	False	False
2	Mammal	Bos taurus	Aurochs, Aurochs, Domestic Cattle (Feral), Domesticated Cattle	No Intervention	False	False
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True
4	Mammal	Cervus elaphus	Wapiti Or Elk	No Intervention	False	False

	category	scientific_name	common_names	conservation_status	is_protected	is_sheep
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True
1139	Vascular Plant	Rumex acetosella	Sheep Sorrel, Sheep Sorrell	No Intervention	False	True
2233	Vascular Plant	Festuca filiformis	Fineleaf Sheep Fescue	No Intervention	False	True
3014	Mammal	Ovis canadensis	Bighorn Sheep, Bighorn Sheep	Species of Concern	True	True
3758	Vascular Plant	Rumex acetosella	Common Sheep Sorrel, Field Sorrel, Red Sorrel, Sheep Sorrel	No Intervention	False	True
3761	Vascular Plant	Rumex paucifolius	Alpine Sheep Sorrel, Fewleaved Dock, Meadow Dock	No Intervention	False	True

	category	scientific_name	common_names	conservation_status	is_protected	is_sheep
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True
3014	Mammal	Ovis canadensis	Bighorn Sheep, Bighorn Sheep	Species of Concern	True	True
4446	Mammal	Ovis canadensis sierrae	Sierra Nevada Bighorn Sheep	Endangered	True	True

	category	scientific_name	common_names	conservation_status	is_protected	is_sheep
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True
3014	Mammal	Ovis canadensis	Bighorn Sheep, Bighorn Sheep	Species of Concern	True	True
4446	Mammal	Ovis canadensis sierrae	Sierra Nevada Bighorn Sheep	Endangered	True	True

Merging Sheep and Observation DataFrames

You've determined what species in `species` are sheep, but now you need to determine where these sheep are locating by combining the data in `sheep_species` and `observations`.

Instructions

1. Now merge `sheep_species` with `observations` to get a DataFrame with observations of sheep. Save this DataFrame as `sheep_observations`.

2. Print and inspect the first couple rows of `sheep_observations` using `.head()`.

3. How many total sheep sightings (across all three species) were made at each national park? Use `groupby` to get the `sum` of `observations` for each `park_name`. Save your answer to `obs_by_park`.

4. Print `obs_by_park`.

   This is the total number of sheep observed in each park over the past 7 days

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

species = pd.read_csv('species_info.csv')
species.fillna('No Intervention', inplace = True)
species['is_protected'] = species.conservation_status != 'No Intervention'
print species.head()
observations = pd.read_csv('observations.csv')
print observations.head()

species['is_sheep'] = species.common_names.apply(lambda x: True if 'Sheep' in x else False)
print species.head()
species_is_sheep = species[species.is_sheep == True]
print species_is_sheep
sheep_species = species[(species.is_sheep == True) & (species.category == 'Mammal')]
print sheep_species

sheep_observations = pd.merge(sheep_species, observations)
print sheep_observations
print sheep_observations
obs_by_park = sheep_observations.groupby('park_name').observations.sum().reset_index()
print obs_by_park
```

	category	scientific_name	common_names	conservation_status	is_protected	is_sheep	park_name	observations
0	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True	Yosemite National Park	126
1	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True	Great Smoky Mountains National Park	76
2	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True	Bryce National Park	119
3	Mammal	Ovis aries	Domestic Sheep, Mouflon, Red Sheep, Sheep (Feral)	No Intervention	False	True	Yellowstone National Park	221
4	Mammal	Ovis canadensis	Bighorn Sheep, Bighorn Sheep	Species of Concern	True	True	Yellowstone National Park	219
5	Mammal	Ovis canadensis	Bighorn Sheep, Bighorn Sheep	Species of Concern	True	True	Bryce National Park	109

	park_name	observations
0	Bryce National Park	250
1	Great Smoky Mountains National Park	149
2	Yellowstone National Park	507
3	Yosemite National Park	282

	park_name	observations
0	Bryce National Park	250
1	Great Smoky Mountains National Park	149
2	Yellowstone National Park	507
3	Yosemite National Park	282

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Plotting Sheep Sightings

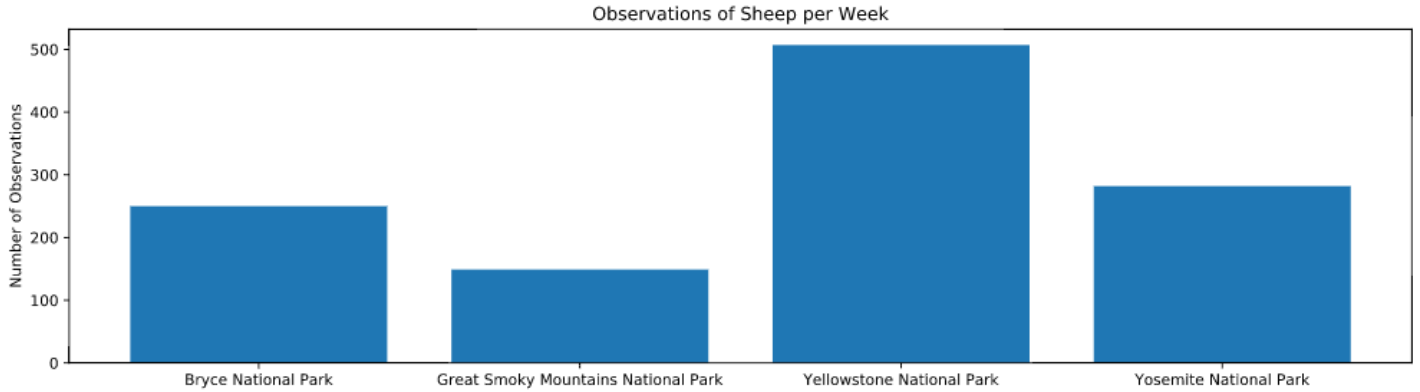
Now it's time to graph the sheep observation data. If we want the figure to easily show the number of sightings at each of the four national parks under investigation, a bar chart is probably the best bet.

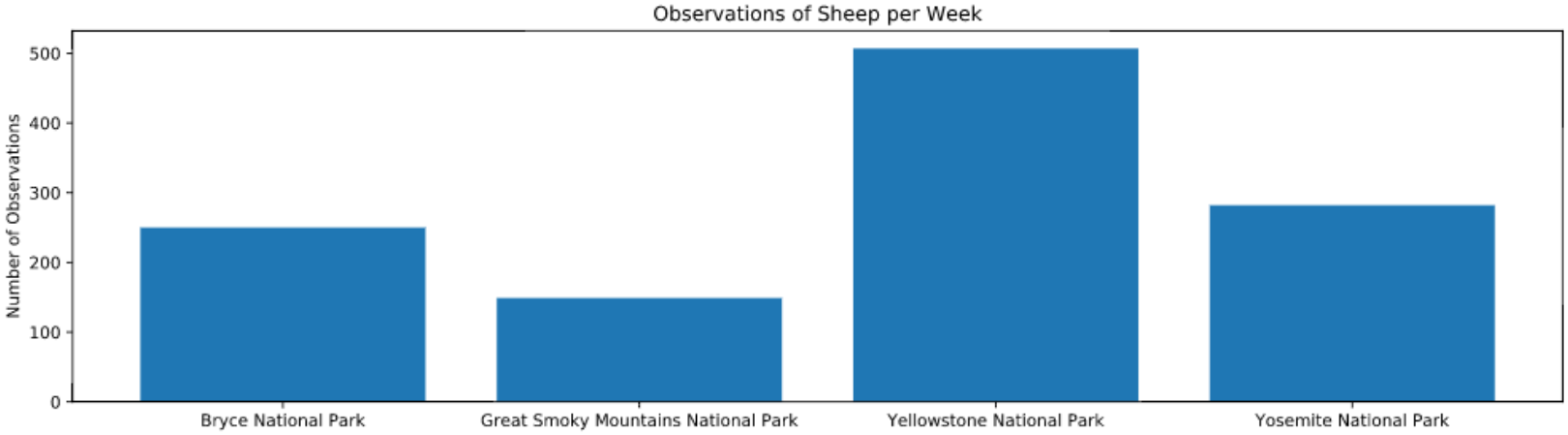
Instructions

1. Create a bar chart showing the different number of observations per week at each park.
    1. Start by creating a wide figure with `figsize=(16, 4)`
    2. Create an axes object called `ax` using `plt.subplot`.
    3. Create a bar chart whose heights are equal to `observations` column of `obs_by_park`.
    4. Create an x-tick for each of the bars.
    5. Label each x-tick with the label from `park_name` in `obs_by_park`
    6. Label the y-axis `Number of Observations`
    7. Title the graph `Observations of Sheep per Week`
    8. Plot the graph using `plt.show()`

```
import codecademylib
import pandas as pd
from matplotlib import pyplot as plt

species = pd.read_csv('species_info.csv')
species['is_sheep'] = species.common_names.apply(lambda x: 'Sheep' in x)
sheep_species = species[(species.is_sheep) & (species.category == 'Mammal')]

observations = pd.read_csv('observations.csv')

sheep_observations = observations.merge(sheep_species)

obs_by_park =
sheep_observations.groupby('park_name').observations.sum().reset_index()

print obs_by_park

plt.figure(figsize = (16, 4))
ax = plt.subplot()
plt.bar(range(len(obs_by_park)),obs_by_park.observations)
ax.set_xticks(range(len(obs_by_park)))
ax.set_xticklabels(obs_by_park.park_name)
plt.ylabel('Number of Observations')
plt.title('Observations of Sheep per Week')
plt.show()
```

	park_name	observations
0	Bryce National Park	250
1	Great Smoky Mountains National Park	149
2	Yellowstone National Park	507
3	Yosemite National Park	282




Observations of Sheep per Week
Number of Observations
0
100
200
300
400
500
Bryce National Park
Great Smoky Mountains National Park
Yellowstone National Park
Yosemite National Park

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Foot and Mouth Reduction Effort - Sample Size Determination

Park Rangers at Yellowstone National Park have been running a program to reduce the rate of foot and mouth disease at that park. The scientists want to test whether or not this program is working. They want to be able to detect reductions of at least 5 percentage point. For instance, if 10% of sheep in Yellowstone have foot and mouth disease, they'd like to be able to know this, with confidence.

The only information that the scientists currently have is that last year it was recorded that 15% of sheep at Bryce National Park have foot and mouth disease. Using this value and the sample size calculator in the browser window on the right, you will need to calculate the number of sheep that they would need to observe from each park to make sure their foot and mouth percentages are significant. Use the default level of significance (90%).

For reference, here is `obs_by_park` table from the previous exercise:

	park_name	observations
0	Bryce National Park	250
1	Great Smoky Mountains National Park	149
2	Yellowstone National Park	507
3	Yosemite National Park	282

1. What is the baseline percentage of this sample size determination?

   Save it to the variable `baseline`.

2. Calculate "Minimum Detectable Effect". Save the value to the variable `minimum_detectable_effect`.

Stuck? Get a hint

3. Plug the baseline and the minimum detectable effect into the sample size calculator. Set the level of significance to 90%.

   Save the sample size per variant from the calculator to the variable `sample_size_per_variant`.

4. Using the observation data calculated earlier, how many weeks would the scientists need to spend at Yellowstone National Park to observe enough sheep?

   Save your answer to the variable `yellowstone_weeks_observing`.

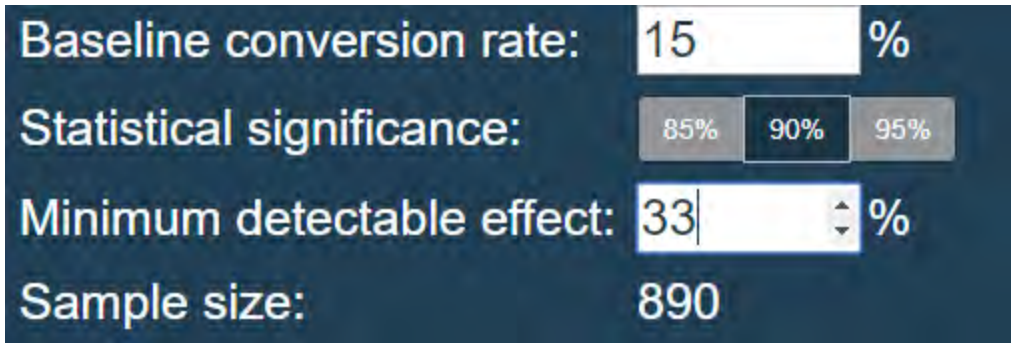
5. The scientists also want to repeat their measurements at Bryce National Park. How many weeks will they have to spend there to observe enough sheep?

   Save your answer to the variable `bryce_weeks_observing`.

```
baseline = 15.0
minimum_detectable_effect = 100.0 * 5.0 / baseline
print minimum_detectable_effect
sample_size_per_variant = 510
yellowstone_weeks_observing = 510 / 507.0
print yellowstone_weeks_observing
yellowstone_weeks_observing2 = 890 / 507.0
print yellowstone_weeks_observing2
bryce_weeks_observing = 510 / 250.0
print bryce_weeks_observing
bryce_weeks_observing2 = 890 / 250.0
print bryce_weeks_observing2
```

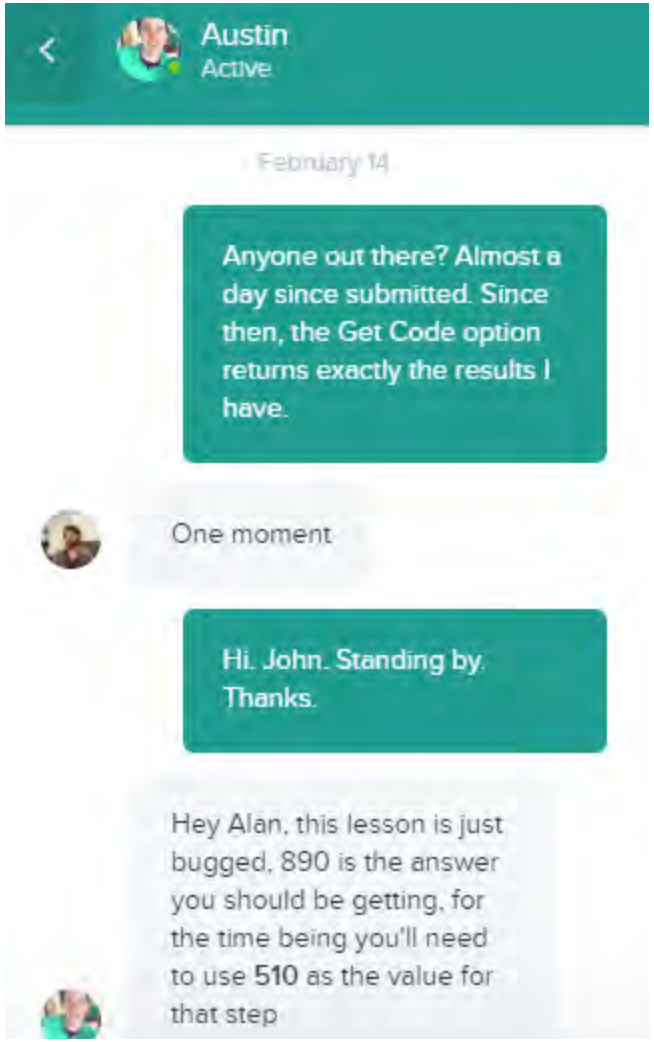
Run

```
33.3333333333
1.00591715976
1.75542406312
2.04
3.56
```



February 13

On page 14 of the final, I supposedly have the correct baseline (15%) and the correct min det eff (33%), but when plugged into the calculator using the required 90% stat sig, I get 890 which apparently is wrong. The model returns a message: "Did you set the level of significance to 90%?" I get the same answer whether I use 33% or -33%. I don't know how to proceed.


Austin
Active
February 14
Anyone out there? Almost a day since submitted. Since then, the Get Code option returns exactly the results I have.
One moment
Hi. John. Standing by. Thanks.
Hey Alan, this lesson is just bugged, 890 is the answer you should be getting, for the time being you'll need to use 510 as the value for that step

Okay. So I have the correct answer then. Thanks. I'll use 510 and hope to get to the finish line today.

No problem! Is there anything else I can help you with?

Nope. I'll do a start over and use 510.

Okay, please reach out to an advisor if you have any more questions. Best of luck and happy coding!

One more thing. Do I calculate questions 4 and 5 (the how many weeks questions) on 890 or 510?

you'll need to calculate them on 510, but also calculate them on 890 and I'll let you know if you're right

For Yellowstone 510 I get 1.005917

For Yellowstone 890 I get 1.75542.

For Bryce 510, I get 2.04.

For Bryce 890, I get 3.56.

I believe with these you just need to round the number up if it's a decimal

you should be getting 1 week for yellowstone and 2 weeks for bryce, at least those are the accepted answers for the 510 value

They are just a tiny fraction above 1 (1.005) and 2 (2.04), so rounding down in this case gets us close enough. Moving on now to the last page. Thanks.

No problem Alan! Let us know if you have any further questions. Best of luck and happy coding!

BIODIVERISTY CAPSTONE PROJECT - INVESTIGATING PROTECTED SPECIES

Foot and Mouth Reduction Effort - Final Thoughts

What do the results of the last exercise tell us?

Given a baseline of 15% occurrence of foot and mouth disease in sheep at Bryce National Park, you found that if the scientists wanted to be sure that a >5% drop in observed cases of foot and mouth disease in the sheep at Yellowstone was significant they would have to observe at least 510 sheep.

Then, using the observation data you analyzed earlier, you found that this would take approximately one week of observing in Yellowstone to see that many sheep, or approximately two weeks in Bryce to see that many sheep.

Congratulations! You've completed Part 2 of the Biodiversity at National Parks Data Analysis Project!

Consider this, starting from only two DataFrames containing species information and species sightings, you were able to create several informative visualizations, perform chi-squared tests to answer the question: "Are some species more likely to be endangered than others?", determine the best place to observe sheep, and calculate the sample size necessary for confident measurements in a disease reduction study.

And that is only scratching the surface of what those two DataFrames can offer. Once you can master the tools and techniques of Data Analysis, every set of data becomes a hundred new stories that you can discover and share. Keep Exploring.

TASK 2: CREATE A SLIDE DECK

Once you've performed your analysis, either on your computer or on Codecademy, you're ready to create your slide deck.

Create a slide deck using Google Drive, Microsoft Powerpoint, or some other presentation software. Your presentation should include the following:

- A title slide
- A section describing the data in **species_info.csv**. Be sure to include some (or all) of what you noticed while working through the notebook.
- A section describing the significance calculations that you did for endangered status between different categories of species.
- A recommendation for conservationists concerned about endangered species, based on your significance calculations
- A section describing the sample size determination that you did for the foot and mouth disease study
- All of the graphs that you created in the notebook

If you like, you can also record a video of yourself giving the presentation and upload it to YouTube.