Complete the following exercises. Each question is a single codeblock with the question at the top as a comment.

- (1) Import numpy as np, matplotlib.pyplot as plt, and pandas as pd. Set matplotlib to inline display.
- (2) Read the Fisher Iris Data in as a variable called df.
- (3) Display the first five lines of df using head.
- (4-7) Plot a histogram of each of the four variables (leaf/sepal length/width). Adjust the number of bins to produce a resolution you find instructive/visually satisfying. Use xlabel and plabel to label the axes.
- (8) Save this figure use savefig.
- (9) Using groupby, group the data by species
- (10) Using the groups, create a bar graph of the mean sepal length by species. Label the axes.
- (11) Use subplots to make a 2x2 pane figure, with each pane a scatterplot of your choosing. Make sure axes are labeled!
- (12) Use tail to print out the last 10 rows of df.
- (13) Use a combination of np.sum() and a selector (e.g., `df['x'] > 3`), count how many rows have a petal length greater than 40.
- (14) How many have a sepal width below the median sepal width?
- (15) Use groups to report the mean values for each variable.
- (16) Calculate a new variable based on existing variables, and create a histogram of its distribution.
- (17) Export the data with the new calculated variable as an Excel file. Open it in Excel, and comment on its appearance. Look at the documentation for to_excel and figure out how to make it so the index values (row numbers) don't get exported.
- (18-23) Download the prepared election data (https://github.com/thomasgmcgrath/python/blob/master/election/state_election_data_1976-2016.csv), load it as a new dataframe, and use some of the commands above to explore the dataset. *Each question # should correspond to a single codeblock that examines and sheds light on the dataset.* Focus not only on executing working code, but usefully exploring the data.

In [1]:
```
# Question 1
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

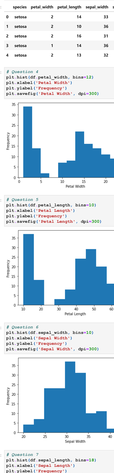
In [2]:
```
# Question 2
df = pd.read_csv('fisher_iris_data.csv')
```

In [3]:
```
# Question 3
df.head()
```

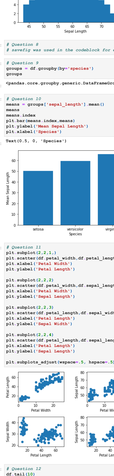
Out[3]:

	species	petal_width	petal_length	sepal_width	sepal_length
0	setosa	2	14	33	50
1	setosa	2	10	36	46
2	setosa	2	16	31	48
3	setosa	1	14	36	49
4	setosa	2	13	32	44

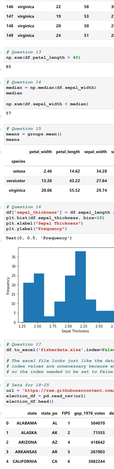
In [39]:
```
# Question 4
plt.hist(df.petal_width, bins=12)
plt.xlabel('Petal Width')
plt.ylabel('Frequency')
plt.savefig('Petal Width', dpi=300)
```



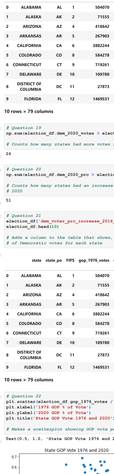
In [38]:
```
# Question 5
plt.hist(df.petal_length, bins=15)
plt.xlabel('Petal Length')
plt.ylabel('Frequency')
plt.savefig('Petal Length', dpi=300)
```



In [37]:
```
# Question 6
plt.hist(df.sepal_width, bins=15)
plt.xlabel('Sepal Width')
plt.ylabel('Frequency')
plt.savefig('Sepal Width', dpi=300)
```



In [36]:
```
# Question 7
plt.hist(df.sepal_length, bins=18)
plt.xlabel('Sepal Length')
plt.ylabel('Frequency')
plt.savefig('Sepal Length', dpi=300)
```



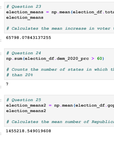
In [35]:
```
# Question 8
# normally was used in the codeblock for each individual histogram
```

In [3]:
```
# Question 9
groups = df.groupby('Species')
groups
```

Out[3]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000002CF1A6A1A90>
```

In [27]:
```
# Question 10
means = groups['sepal_length'].mean()
means
means.index
plt.bar(means.index,means)
plt.xlabel('Mean Sepal Length')
plt.xlabel('Species')
```

Out[27]:
```
Text(0.5, 0, 'Species')
```



In [33]:
```
# Question 11
fig subplot(2,2,1)
plt.scatter(df.petal_width,df.petal_length)
plt.xlabel('Petal Width')
plt.ylabel('Petal Length')

plt.subplot(2,2,2)
plt.scatter(df.petal_width,df.sepal_length)
plt.xlabel('Petal Width')
plt.ylabel('Sepal Length')

plt.subplot(2,2,3)
plt.scatter(df.petal_length,df.sepal_width)
plt.xlabel('Petal Length')
plt.ylabel('Sepal Width')

plt.subplot(2,2,4)
plt.scatter(df.petal_length,df.sepal_length)
plt.xlabel('Petal Length')
plt.ylabel('Sepal Length')

plt.subplots_adjust(wspace=0.3, hspace=.5)
```


In [34]:
```
# Question 12
df.tail(10)
```

Out[34]:

	species	petal_width	petal_length	sepal_width	sepal_length
140	virginica	16	58	30	72
141	virginica	21	59	30	71
142	virginica	18	56	29	63
143	virginica	23	69	26	77
144	virginica	19	61	28	74
145	virginica	18	63	29	73
146	virginica	22	58	30	65
147	virginica	19	53	27	64
148	virginica	20	59	25	67
149	virginica	24	51	28	58

In [30]:
```
# Question 13
np.sum(df.petal_length > 40)
```

Out[30]:
```
85
```

In [31]:
```
# Question 14
median = np.median(df.sepal_width)
median

np.sum(df.sepal_width < median)
```

Out[31]:
```
57
```

In [5]:
```
# Question 15
means = groups.mean()
means
```

Out[5]:

species	petal_width	petal_length	sepal_width	sepal_length
setosa	2.46	14.62	34.28	50.10
versicolor	13.26	43.22	27.64	59.36
virginica	20.56	55.52	29.74	65.88

In [29]:
```
# Question 16
df['sepal_thickness'] = df.sepal_length / df.sepal_width
plt.hist(df.sepal_thickness, bins=15)
plt.xlabel('Sepal Thickness')
plt.ylabel('Frequency')

Text(0, 0.5, 'Frequency')
```


In [28]:
```
# Question 17
df.to_excel('fisherdata.xlsx',index=False)

# The excel file looks just like the dataframe after sepal_thickness was added. The
# index values are unnecessary because excel sheets include them by default,
# so the index needed to be set to False for them to not show up.
```

In [6]:
```
# Data for 18-23
url = 'https://raw.githubusercontent.com/thomasgmcgrath/pandata/master/election/state_election_data_1976-2020.csv'
election_df = pd.read_csv(url)
election_df.head()
```

Out[6]:

	state	state_po	FIPS	gop_1976_votes	dem_1976_votes	totalvotes_1976	gop_1976_pct	dem_1976_pct	gop_minus_dem_pct_1976
0	ALABAMA	AL	1	504070	659170	1162850	42.61	55.73	-13.12
1	ALASKA	AK	2	71555	44058	123574	57.90	35.65	22.25
2	ARIZONA	AZ	4	418642	295602	742719	56.37	39.80	16.57
3	ARKANSAS	AR	5	267903	498604	767535	34.90	64.96	-30.06
4	CALIFORNIA	CA	6	3582244	3742284	7893770	49.75	47.95	1.80

5 rows × 75 columns

In [7]:
```
# Question 19
election_df['voter_turnout_change'] = election_df.totalvotes_2020 - election_df.totalvotes_1976
election_df.head(10)

# Adds a column to the table showing increase in voter turnout for each state between
# 1976 and 2020
```

Out[7]:

	state	state_po	FIPS	gop_1976_votes	dem_1976_votes	totalvotes_1976	gop_1976_pct	dem_1976_pct	gop_minus_dem_pct_1976	g
0	ALABAMA	AL	1	504070	659170	1162850	42.61	55.73	-13.12	
1	ALASKA	AK	2	71555	44058	123574	57.90	35.65	22.25	
2	ARIZONA	AZ	4	418642	295602	742719	56.37	39.80	16.57	
3	ARKANSAS	AR	5	267903	498604	767535	34.90	64.96	-30.06	
4	CALIFORNIA	CA	6	3582244	3742284	7893770	49.75	47.95	1.80	
5	COLORADO	CO	8	584278	460801	1091440	54.03	42.61	11.42	
6	CONNECTICUT	CT	9	719261	647895	1386355	51.88	46.73	5.15	
7	DELAWARE	DE	10	109780	122461	235642	46.59	51.97	-5.38	
8	DISTRICT OF COLUMBIA	DC	11	27873	137818	168830	16.51	81.63	-65.12	
9	FLORIDA	FL	12	1409531	1636000	3150631	46.64	51.93	-5.29	

10 rows × 79 columns

In [8]:
```
# Question 20
np.sum(election_df.dem_2020_pre > election_df.dem_2016_pre)

# Counts how many states had an increase in democratic vote percentage between 2016 and
# 2020
```

Out[8]:
```
51
```

In [9]:
```
# Question 21
election_df['dem_voter_pct_increase_2016_to_2020'] = election_df.dem_2020_pct - election_df.dem_2016_pct
election_df.head(10)

# Adds a column to the table that shows, from 2016 to 2020, the increase in percentage
# of Democratic votes for each state
```

Out[9]:

	state	state_po	FIPS	gop_1976_votes	dem_1976_votes	totalvotes_1976	gop_1976_pct	dem_1976_pct	gop_minus_dem_pct_1976	g
0	ALABAMA	AL	1	504070	659170	1162850	42.61	55.73	-13.12	
1	ALASKA	AK	2	71555	44058	123574	57.90	35.65	22.25	
2	ARIZONA	AZ	4	418642	295602	742719	56.37	39.80	16.57	
3	ARKANSAS	AR	5	267903	498604	767535	34.90	64.96	-30.06	
4	CALIFORNIA	CA	6	3582244	3742284	7893770	49.75	47.95	1.80	
5	COLORADO	CO	8	584278	460801	1091440	54.03	42.61	11.42	
6	CONNECTICUT	CT	9	719261	647895	1386355	51.88	46.73	5.15	
7	DELAWARE	DE	10	109780	122461	235642	46.59	51.97	-5.38	
8	DISTRICT OF COLUMBIA	DC	11	27873	137818	168830	16.51	81.63	-65.12	
9	FLORIDA	FL	12	1409531	1636000	3150631	46.64	51.93	-5.29	

10 rows × 79 columns

In [10]:
```
# Question 22
election_df.gop_1976_votes / election_df.totalvotes_1976, election_df.gop_2020_votes / election_df.totalvotes_2020
plt.hist(election_df.gop_1976_votes / election_df.totalvotes_1976, label='2020 GOP % of Vote')
plt.ylabel('2020 GOP % of Vote')
plt.title('State GOP Vote 1976 and 2020')

# Makes a scatterplot showing GOP vote percentage in 1976 and 2020

Text(0.5, 1.0, 'State GOP Vote 1976 and 2020')
```


In [11]:
```
# Question 23
election_df['dem_increase'] = np.mean(election_df.totalvotes_2020 - election_df.totalvotes_2016)
election_df.head()

# Calculates the mean increase in voter turnout per state between 2016 to 2020
```

Out[11]:
```
65788.07843137255
```

In [12]:
```
# Question 24
np.sum(election_df.dem_2020_pre > 60)

# Counts the number of states in which the 2020 Democratic margin of victory was greater
# than 20%
```

Out[12]:
```
9
```

In [13]:
```
# Question 25
election_mean2 = np.mean(election_df.gop_2020_votes)
election_mean2

# Calculates the mean number of Republican voters per state in 2020

1435218.549019608
```