Homework 4: Tables and data manipulation

Welcome to the fourth homework!

In this homework you will practice manipulating data in Tables. Please complete this notebook by filling in the cells provided.

Deadline:

This assignment is due **Wednesday July 16 at 10pm in Gradescope.**

Directly sharing answers is not okay, but discussing problems with the course staff or with other students is encouraged. *You are not allowed to use chatGPT to answer any questions on the homework*. If your answers include code that we have not discussed in class that is copied from chatGPT, this will be considered a violation of academic dishonesty and will face disciplinary action.

You should start early so that you have time to get help if you're stuck.

Getting started

In order to complete the homework, it is necessary to download a few files. Please run the code below **only once** to download data needed to complete the homework. To run the code, click in the cell below and press the play button (or press shift-enter).

In [75]:
```
# if you are running this notebook in colabs, please uncomment and
# !pip install https://github.com/emeyers/YData_package/tarball/mas
```

In [2]:
```
# Please run this code once to download the files you will need to

import YData

YData.download.download_data("ACS_2017_sample_01.csv")
YData.download.download_data("ACS_major_codes.tsv")
YData.download.download_data("2018_Central_Park_Squirrel_Census.csv
```

0. Quote and reaction

This week's reading is from the website FiveThirtyEight and discusses which college majors make the most money. We will do a similar analysis, with

slightly different data, on this homework. Please read the blog post, and write down a quote that you find interesting. In the space below, write down the quote as well as a one paragraph description for why you thought the quote was interesting.

Question 0.1 (5 points) Please write down your "quote and reaction" here.

Quote: The link between education and earnings is notoriously fraught, with cause and effect often difficult to disentangle. But a look at detailed data on college graduates by major reveals some clear messages: Don't be pre-med if you aren't planning to go to medical school; don't assume that all "STEM" — science, technology, engineering and math — majors are the same; and if you study drama, be prepared to wait tables.

Reaction: This quote describe the fact of the society and give a abstract of what the passage is going to discuss. It is really amazing that the "discriminate" over major are so common around the world and the huge difference a degree could make.

In [2]:
```
# This cell imports functions from packages we will use below.
# Please run it each time you load the Jupyter notebook

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

%matplotlib inline
```

1. Squirrels of New York

Have you ever wondered how many squirrels are in New York City parks, where they are located, and what they are up to? Well some people have, and those people created The Squirrel Census which contains information about squirrels sightings in New York City (NYC).

Let's look at data on squirrels in Central Park that was collected in 2018.

People living in NYC are often very busy so they can sometimes seem indifferent to their surroundings. But is this true of squirrels living in NYC too? Let's examine this important question and also see if squirrel indifference varies depending on the color of their fur.

In [5]:
```
squirrel_data = pd.read_csv("2018_Central_Park_Squirrel_Census.csv"
```

Question 1.1 (3 points) The code in the cell above loads the data as a panda DataFrame. As a first step in analyzing data, it is always a good idea to take a quick visual inspection of the data.

Please do the following as an initial exploration of the data:

- Print out the shape of the data
- Print out the first 3 rows of the DataFrame.

Hint: the `.head()` method will be useful for the last problem.

In [8]:
```
print(squirrel_data.shape)
print(squirrel_data.head(3))
```

```
(3023, 31)
           X          Y Unique Squirrel ID Hectare Shift      Date
\
0 -73.956134  40.794082     37F-PM-1014-03     37F    PM  10142018
1 -73.968857  40.783783     21B-AM-1019-04     21B    AM  10192018
2 -73.974281  40.775534     11B-PM-1014-08     11B    PM  10142018

   Hectare Squirrel Number  Age Primary Fur Color Highlight Fur Colo
r  ...  \
0                        3  NaN              NaN                 Na
N  ...
1                        4  NaN              NaN                 Na
N  ...
2                        8  NaN             Gray                 Na
N  ...

    Kuks  Quaas  Moans Tail flags Tail twitches  Approaches  Indiffe
rent  \
0  False  False  False      False         False       False        F
alse
1  False  False  False      False         False       False        F
alse
2  False  False  False      False         False       False        F
alse

   Runs from  Other Interactions
Lat/Long
0      False                NaN  POINT (-73.9561344937861 40.794082
3884086)
1      False                NaN  POINT (-73.9688574691102 40.783782
5208444)
2      False                NaN  POINT (-73.97428114848522 40.77553
3619083)

[3 rows x 31 columns]
```

Question 1.2 (4 points) For a first set of analyses, let's look at the Indifferent column, which describes whether a squirrel was indifferent to the presence of the human observer.

Please extract the `Indifferent` column as a Series and save it to the name `squirrel_indifference`. Also print out the dtype to show the type of data that is stored in the Series.

In [11]:
```
squirrel_indifference = squirrel_data.Indifferent
print(squirrel_indifference.dtype)
```

```
bool
```

Question 1.3 (4 points): Let's now look at the following questions:

1. How many squirrels are there in total?
2. How many squirrels are indifferent?
3. What proportion of squirrels are indifferent?

Please print out numbers that show the answers to these questions. For the proportion, *round your answer to 3 decimal places*. In the answer section below, answer the above questions in complete sentences using the numbers.

Hint: As you recall, series are like ndarrays so we can use many of the same ndarray functions we have previously used.

In [18]:
```
print(f"There are {len(squirrel_indifference)} squirrels in total")
sqr_list = squirrel_indifference.tolist()
print(f"There are {sqr_list.count(True)} squirrels that are indiffe
print(f"There are {round(sqr_list.count(True)/len(squirrel_indiffer
```

```
There are 3023 squirrels in total
There are 1454 squirrels that are indifference
There are 0.481 of squirrels that are indifferent
```

In [24]:
```
#method 2
squirrel_indifference[squirrel_indifference == True].count()
```

Out[24]: 1454

Answers

1. There are 3023 squirrels in total
2. There are 1454 squirrels that are indifference
3. There are 0.481 of squirrels that are indifferent

Question 1.4 (4 points): For the next set of questions, let's look at whether the proportion of squirrels that are indifferent differs for squirrels with different fur colors.

To start, please extract a DataFrame that has just two columns:

1. `The Primary Fur Color`: The main fur color of each squirrel
2. `Indifferent`: Whether the squirrel was indifferent

Save this DataFrame to the name `squirrel_fur` and display the first 2 rows of this DataFrame below.

In [27]:
```
squirrel_fur =  squirrel_data[["Primary Fur Color","Indifferent"]]
squirrel_fur.head(2)
```

Out[27]:

	Primary Fur Color	Indifferent
0	NaN	False
1	NaN	False

Question 1.5 (4 points): Let's now count how many squirrels there are of each color.

Please create a DataFrame called `color_count_mislabeled` which has the counts of how many squirrels are of each color. Once you have calculated this, display the `color_count_mislabeled` DataFrame below.

Hint: The `.groupby()` and `.count()` methods should be useful here.

In [32]:
```
color_count_mislabeled = squirrel_fur.groupby("Primary Fur Color").
color_count_mislabeled
```

Out[32]:

Primary Fur Color	Indifferent
Black	103
Cinnamon	392
Gray	2473

In [37]:
```
# method 2
color_count_mislabeled = squirrel_fur.groupby("Primary Fur Color").
color_count_mislabeled
```

Out[37]:

Primary Fur Color	Indifferent
Black	103
Cinnamon	392
Gray	2473

Question 1.6 (4 points): If you look at the `color_count_mislabeled` DataFrame you created in the previous question, you will notice that the column that contains the counts of each squirrel is titled `Indifferent`. This is a poor name choice since it really contains the counts of squirrels of each color and has nothing to do with whether the squirrels were indifferent. Let's rename it so that the column is called as `Count`.

Recall that to relabel a column, we can create a dictionary where the keys are the old column names, and the values are the new column names. For example, if we wanted to replace the column name `old_column` with the name `new_column` we would create a dictionary that had the values `{"old_column": "new_column"}`. Once we create a dictionary that can relabel a column name, we can update the column name using the method: `my_df.rename(columns = new_name_dictionary)`.

Please go ahead and rename the column `Indifferent` column to be called `Count` in the `color_count_mislabeled` DataFrame. Save the new DataFrame in the name `color_count` and display this DataFrame to show you have the correct answer.

In [40]:
```
tool_dict = {"Indifferent":"Count"}
color_count = color_count_mislabeled.rename(columns = tool_dict)
color_count
```

Out[40]:

Primary Fur Color	Count
Black	103
Cinnamon	392
Gray	2473

Question 1.7 (4 points): Now let's get the total number of squirrels that are reported in the `color_count` DataFrame by summing the values in the (newly renamed) `Count` column.

Please print this number below. In the answer section, in 1-3 sentences, report why this number is different from the number of rows in the original `squirrel_data` DataFrame that you reported in question 1.1.

In [111…
```
color_count.sum()
```

Out[111…
```
Count    2968
dtype: int64
```

Answer: There is a difference because there are some NaN value in the Primary Fur Color column.

Question 1.8 (4 points): Now let's count how many squirrels of each color are indifferent. To do this, use the `.groupby()` and `.sum()` methods. Save the results in the name `indifferent_color_count`, and print out this DataFrame to show your work.

In the answer section, in 1-3 sentences describe why the `sum()` method

gives the total number of squirrels that are *indifferent* (as opposed to, for example, using the `count()` method which returns the total number of squirrels regardless of whether they are indifferent).

Hint: think about what type of values the `sum()` function is summing over.

In [46]:
```
indifferent_color_count =  squirrel_fur.groupby("Primary Fur Color"
print(indifferent_color_count)
```

```
                   Indifferent
Primary Fur Color
Black                       44
Cinnamon                   181
Gray                      1219
```

Answer: The sum function sums all bollean values and since there are some True(=1) and some False(=0), the .sum must be less than .count since the later one gives the number of objects that has a value, regardless what their value is.

Question 1.9 (4 points): Let's now join the `color_count` and `indifferent_color_count` DataFrames together to get one table that has information about both the total number of squirrels of each color and number of indifferent squirrels of each color.

Since the `color_count` and `indifferent_color_count` DataFrames both have the same Index values (due the fact that the same column was passed to the `.groupby` method), we can simply use the `.join()` method to join these tables together.

Please join these DataFrames together and save the result to a DataFrame called `counts_joined`. Print this DataFrame so we can see the answer.

In [52]:
```
counts_joined = color_count.join(indifferent_color_count,how = "lef
counts_joined
```

Out[52]:

Primary Fur Color	Count	Indifferent
Black	103	44
Cinnamon	392	181
Gray	2473	1219

Question 1.10 (4 points): Now let's use the `counts_joined` DataFrame to calculate the proportion of squirrels of each color that are indifferent.

Save the results to a Series called `prop_indifferent` and print out the Series to show you have the correct answer.

In [56]:
```
prop_indifferent = counts_joined["Indifferent"]/counts_joined["Coun
prop_indifferent
```

Out[56]:
```
Primary Fur Color
Black       0.427184
Cinnamon    0.461735
Gray        0.492924
dtype: float64
```

Question 1.11 (4 points): Let's now add these proportions values as a new column to our `counts_joined` DataFrame. Please do this and print out the `counts_joined` DataFrame to show it has the correct 3 columns.

In [63]:
```
counts_joined["proportion"] = prop_indifferent
counts_joined
```

Out[63]:

Primary Fur Color	Count	Indifferent	proportion
Black	103	44	0.427184
Cinnamon	392	181	0.461735
Gray	2473	1219	0.492924

Question 1.12 (4 points): From looking at the `counts_joined` DataFrame you printed out in the previous question, you will notice that the proportion of squirrels that are indifferent differs for the three colors of squirrels. However, you will also notice that these proportions are only slightly different.

In the answer section, in 1-3 sentences, state whether you think there is a real difference between the proportion of squirrels that are indifferent depending on the color of their fur, or whether these results are due to the random sample of squirrels that were observed; e.g., if NYC residents went out and observed another set of ~3,000 squirrels, do you think the highest proportion of squirrels that were indifferent would again be the gray squirrels?

Note: This type of question is what statistical analyses are useful for. We will explore this type of analysis in the second half of the semester.

Answer I consider there is a significant real difference of the proportion. This is because the lag between diffferent color of fur is small in absolute amount but relatively large in proportion, for instance it is $\frac{1}{6}$ higher for the proportion of indifference in gray squirrel than the black one, which is a high proportion.

Recreating the FiveThirtyEight college major's analysis

Let's now recreate the analysis that was done in the FiveThirtyEight article that you read on part 0 of this homework. In particular, we will use our sample of the ACS data to calculate average income of different college majors.

Note: the ACS data has different categories of majors compared to the FiveThirtyEight data, so our results will be a bit different from the article (also our data is more recent).

Question 2.1 (2 points): Let's start in the usual way. The code below loads the ACS data and stores the results in the name `acs_data`. Please display the first 3 rows of this data in the cell below.

In [67]:
```
acs_data = pd.read_csv("ACS_2017_sample_01.csv")
```

In [69]:
```
acs_data.head(3)
```

Out[69]:

	YEAR	SAMPLE	SERIAL	CBSERIAL	NUMPREC	HHWT	HHTYPE	
0	2017	201701	250395	2017000834943	2	79	2	2
1	2017	201701	1062813	2017000532479	2	27	1	1
2	2017	201701	1235164	2017001001296	1	48	6	1

3 rows × 61 columns

Question 2.2 (4 points): As you are aware, there are many columns in our ACS data. To make the data easier to manage, let's reduce the data to only the two columns we care about which are:

- `DEGFIELD`: A code indicating which field of study an individual majored in
- `INCEARN`: The earned income an individual made

Create a DataFrame called `acs_data2` that only has data from these columns, and print the first 3 rows to show your code worked.

In [72]:
```
acs_data2 = acs_data[["DEGFIELD","INCEARN"]]
acs_data2.head(3)
```

Out[72]:

	DEGFIELD	INCEARN
0	23	0
1	0	0
2	0	150

Question 2.3 (4 points): Let's now look at how many unique majors there are in our data. Please print out an ndarray that contains all the unique major

numbers. Also print out the number of unique majors there are.

Hint: `np.unique()` could be useful here.

In [76]:
```
print(np.unique(acs_data2["DEGFIELD"]))
print(len(np.unique(acs_data2["DEGFIELD"])))
```

```
[ 0 11 13 14 15 19 20 21 23 24 25 26 29 32 33 34 35 36 37 40 41 48 4
9 50
 52 53 54 55 56 59 60 61 62 64]
34
```

Question 2.4 (2 points): As you will have noticed, identifying college majors by a numeric code makes the data hard to interpret. A large part of Data Science consists of converting this type of more raw data, into more meaningful names using codebooks. A codebook that explains what these numeric codes mean can be found here.

Fortunately, someone has made this process converting codes to more meaningful names easier for you by creating a file with this codebook information!

The codebook is loaded as a DataFrame named `major_codes` in the cell below. Please print out the first 5 rows of this codebook DataFrame.

In [83]:
```
major_codes = pd.read_csv("ACS_major_codes.tsv", sep = "\t")
major_codes.head()
```

Out[83]:

	Code	Label
0	0	NaN
1	11	Agriculture
2	13	Environment and Natural Resources
3	14	Architecture
4	15	Area, Ethnic, and Civilization Studies

Question 2.5 (5 points): Let's now join the names in the `major_codes` DataFrame onto the `acs_data2` DataFrame, so that each row of the `acs_data2` has a more meaningful major name as well as just a numeric code.

Since we are joining our DataFrames based on values in a *column* (rather than based on Index values) we will use the `.merge()` method. Some arguments that will be useful when using the `.merge()` method are:

- `left_on`: The name of the column in the left DataFrame
- `right_on`: The name of the column on the right DataFrame

- `how`: whether it should be a "left", "right", "inner", "outer", etc.

Save the results of the joined DataFrame in the name `acs_data3` and print out the first three rows of the DataFrame to show your work.

Hint: Think carefully about what type of join you would like to do, and then use `how` and the other column names appropriately.

In [86]:
```
acs_data3 = acs_data2.merge(major_codes,how="left",left_on="DEGFIELI
acs_data3.head(3)
```

Out[86]:

	DEGFIELD	INCEARN	Code	Label
0	23	0	23	Education Administration and Teaching
1	0	0	0	NaN
2	0	150	0	NaN

Question 2.6 (4 points): Let's rename some of the columns in our DataFrame to have more meaningful names. In particular, please rename:

- "INCEARN" to be "Income"
- "Label" to be "Major"

Save the results to the name `acs_data4` and as always, print the first 3 rows.

Hint: Recall from your analysis of the squirrel data that you can rename columns using a dictionary and the `.rename(columns = rename_dictionary)` method.

In [91]:
```
acs_data4 = acs_data3.rename(columns={"INCEARN":"Income","Label":"M
acs_data4.head(3)
```

Out[91]:

	DEGFIELD	Income	Code	Major
0	23	0	23	Education Administration and Teaching
1	0	0	0	NaN
2	0	150	0	NaN

Question 2.7 (4 points): Now that we have meaningful names in our DataFrame, let's get rid of the college major codes which we will no longer need.

Please create a DataFrame called `acs_data5` that has only the "Income" and "Major" columns, and print the first 3 rows.

In [95]:
```
acs_data5 = acs_data4[["Income","Major"]]
```

```
acs_data5.head(3)
```

Out[95]:

	Income	Major
0	0	Education Administration and Teaching
1	0	NaN
2	150	NaN

Question 2.8 (5 points): Now let's compute the mean and median salaries separately for each major. Let's also get the counts of how many people we have in our data for each major.

To do this, you can group the input DataFrame ('acs_data5') by major. Then use the `.agg()` method where you specify tuples indicating the column you want to apply a function to, and the function you want to use. For example, we could use code in the form:

```
my_aggregated_results =
input_df.groupby("grouping_col_name").agg(
    output_col_name1 =('column_to_summarize1',
'statistic_name1'),
    output_col_name2 =('column_to_summarize2',
'statistic_name2'),
    output_col_name3 =('column_to_summarize3',
'statistic_name3'),

)
```

Where:

- `input_df`: is the DataFrame we want to get summary statistics from
- `"grouping_col_name"`: is the column we want to use to group our input_df DataFrame
- `output_col_name1`: is the name of the column where we want to save the statistic we calculate
- `'column_to_summarize1'` is the name of column we want to use to get our statistic from
- `'statistic_name1'` is the name of the statistic function we want to apply
- same for `output_col_name2`, `column_to_summarize2`, etc.

Please create a DataFrame called `major_stats` that has the count, mean, and median for each college major area; i.e., `major_stats` should have an index which is the college major area, and there should be three columns called `count`, `mean` and `median` which corresponds to statistical summaries of the `acs_data5` DataFrame. As always, show the first 3 rows of

the results to show your work.

In [111…
```
major_stats = acs_data5.groupby("Major").agg(
    count = ("Income","count"),
    mean = ("Income","mean"),
    median = ("Income","median"),
)
major_stats.head(3)
```

Out[111…

Major	count	mean	median
Agriculture	18	84783.333333	36000.0
Architecture	18	46244.444444	27500.0
Area, Ethnic, and Civilization Studies	5	24600.000000	0.0

Question 2.9 (4 points): You will notice from looking at the `major_stats` DataFrame that the decimal places in the data make the values in the DataFrame hard to read. To deal with this we could round the data, but an even nicer way to make the data legible is to convert all values into integers.

Please convert all the values in the data frame to integers and save the results to a DataFrame called `major_stats2`. As always, "show your work".

Hint: the `.astype()` method could be useful here.

In [113…
```
major_stats2 = major_stats.astype("int")
major_stats2.head(3)
```

Out[113…

Major	count	mean	median
Agriculture	18	84783	36000
Architecture	18	46244	27500
Area, Ethnic, and Civilization Studies	5	24600	0

Question 2.10 (5 points): You will also notice from looking at the `count` column that there are some majors where there were only a few people in our dataset that had that major. Because of these small numbers, our statistics for these majors will be unreliable.

While we could deal with this issue here by looking at a larger dataset (rather than just a smaller random sample), for the sake of simplicity, let's just remove all majors that have counts less than 30.

Please create a DataFrame called `major_stats3` that only has rows from

`major_stats2` where there were more than 30 or more majors in our dataset. Print the results to show your work.

Hint: Boolean masking could be useful here!

In [120…
```
major_stats3 = major_stats2[major_stats2["count"]>30]
major_stats3
```

Out[120…

Major	count	mean	median
Biology and Life Sciences	107	76960	55000
Business	485	64302	42500
Communications	83	70038	50000
Computer and Information Sciences	60	92122	74500
Criminal Justice and Fire Protection	49	55866	55000
Education Administration and Teaching	329	30110	12000
Engineering	200	68090	54500
English Language, Literature, and Composition	72	47518	39500
Fine Arts	117	47930	32000
History	44	58438	27500
Liberal Arts and Humanities	41	53819	50000
Mathematics and Statistics	33	46963	20000
Medical and Health Sciences and Services	179	52230	54000
Physical Fitness, Parks, Recreation, and Leisure	33	55524	40000
Physical Sciences	77	79300	45800
Psychology	114	56819	31000
Social Sciences	176	57148	36250

Question 2.11 (5 points): Finally, let's sort our data to see which majors make the most money!

Please sort the data based on the mean income level. In particular, sort the data so that majors that make the most money appear at the top of the DataFrame. Save it to the name `major_stats4`, and then print the full DataFrame to see which majors make the most money.

In the answer section, report if there might be some upsides to suffering through these homework problems in order to learn more Data Science.

In [130…
```
major_stats4 = major_stats3.sort_values("mean",ascending=False)
major_stats4
```

Out[130…

Major	count	mean	median
Computer and Information Sciences	60	92122	74500
Physical Sciences	77	79300	45800
Biology and Life Sciences	107	76960	55000
Communications	83	70038	50000
Engineering	200	68090	54500
Business	485	64302	42500
History	44	58438	27500
Social Sciences	176	57148	36250
Psychology	114	56819	31000
Criminal Justice and Fire Protection	49	55866	55000
Physical Fitness, Parks, Recreation, and Leisure	33	55524	40000
Liberal Arts and Humanities	41	53819	50000
Medical and Health Sciences and Services	179	52230	54000
Fine Arts	117	47930	32000
English Language, Literature, and Composition	72	47518	39500
Mathematics and Statistics	33	46963	20000
Education Administration and Teaching	329	30110	12000

Answer It seems so since student major in computer and information sciences has the highest mean of Income.

3. Answer another question on the ACS data or another dataset (6 points)

Now that you have a new set of skills manipulating DataFrames, try again to answer another interesting question on the ACS data. This could be one of the questions you initially posed on the second homework, or a new question.

Alternatively, you are welcome to find another dataset on the Internet (e.g., in a .csv file), and then show an analysis on that dataset.

In [145…
```
acs_data1 = pd.read_csv("ACS_2017_sample_01.csv")
acs_data1.head()
```

Out [145…

	YEAR	SAMPLE	SERIAL	CBSERIAL	NUMPREC	HHWT	HHTYPE	
0	2017	201701	250395	2017000834943	2	79	2	2
1	2017	201701	1062813	2017000532479	2	27	1	
2	2017	201701	1235164	2017001001296	1	48	6	
3	2017	201701	142278	2017000769070	3	50	1	1
4	2017	201701	954089	2017000012970	5	15	1	2

5 rows × 61 columns

In [155…

```
acs_data2 = acs_data1[["NUMPREC","INCEARN"]]
```

In [175…

```
acs_data3 = acs_data2.groupby("NUMPREC").agg(
    mean = ("INCEARN","mean"),
    count = ("INCEARN","count"),
)
acs_data3
```

Out [175…

NUMPREC	mean	count
1	20961.019708	1573
2	29983.957661	2787
3	28909.584519	1757
4	27462.511823	1903
5	21898.157407	1080
6	19373.614458	498
7	10414.601770	226
8	12470.506329	79
9	8829.166667	48
10	10090.476190	21
11	6471.428571	14
12	0.000000	4
13	12000.000000	1
14	0.000000	3
15	0.000000	2
18	0.000000	1
19	0.000000	1
20	0.000000	2

In [181…

```
acs_data4 = acs_data3[acs_data3["count"]>20]
acs_data4
```

Out[181...

NUMPREC	mean	count
1	20961.019708	1573
2	29983.957661	2787
3	28909.584519	1757
4	27462.511823	1903
5	21898.157407	1080
6	19373.614458	498
7	10414.601770	226
8	12470.506329	79
9	8829.166667	48
10	10090.476190	21

Answer: From the analysis above we can find that as the number of people in a family goes up, there is a trend that their income falls down, which contradicts to the common believe since more people mean that they require more money to raise them.

4. Reflection (3 points)

Please reflect on how the homework went by going to Canvas, going to the Quizzes link, and clicking on reflection on homework 4.

5. Submission

Please submit your assignment as a .pdf on Gradescope. You can access Gradescope through Canvas on the left hand side of the class home page. The problems in each homework assignment are numbered. **NOTE:** When submitting on Gradescope, please select the correct pages of your pdf that correspond to each problem. **Failure to mark pages correctly will result in points being deducted from your homework score.**

If you are running Jupyter Notebooks through an Anaconda installation on your own computer, you can produce the .pdf by completing the following steps:

1. Go to "File" at the top-left of your Jupyter Notebook
2. Under "Download as", select "HTML (.html)"
3. After the .html has downloaded, open it and then select "File" and "Print" (note you will not actually be printing)

4. From the print window, select the option to save as a .pdf

If you are running the assignment in a Google Colabs, you can use the following instructions:

1. Go to "File" at the top-left of your Jupyter Notebook and select "File" and "Print" (note you will not actually be printing)
2. From the print window, select the option to save as a .pdf
3. Be sure to look over the pdf file to make sure all your code and written work is saved in a clear way.

In [ ]: