Homework 5: For Loops and Functions in Python

Welcome to the fifth homework!

In this homework you practice using for loops and writing functions.

Please complete this notebook by filling in the cells provided. For all problems that you must write explanations and sentences for, please provide your answer in the designated space.

Recommended Reading from Data Science for Everyone:

- Chapter 7.2: Loops
- Chapter 7.1: Defining functions
- Chapter 7.3: Conditional statements

Deadline:

This assignment is due **Friday July 18 at 10 PM in Gradescope.**

Directly sharing answers is not okay, but discussing problems with the course staff or with other students is encouraged. Refer to the policies page to learn more about how to learn cooperatively.

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 [3]:
```
# if you are running this notebook in colabs, please uncomment and
# !pip install https://github.com/emeyers/YData_package/tarball/mas
```

In [4]:
```
# 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("dennys.csv")
YData.download.download_data("laquinta.csv")
YData.download.download_image("mitch.png")
YData.download.download_image("sign.jpg")
```

```
The file `ACS_2017_sample_01.csv` already exists.
If you would like to download a new copy of the file, please rename
the existing copy of the file.
The file `dennys.csv` already exists.
If you would like to download a new copy of the file, please rename
the existing copy of the file.
The file `laquinta.csv` already exists.
If you would like to download a new copy of the file, please rename
the existing copy of the file.
The file `mitch.png` already exists.
If you would like to download a new copy of the file, please rename
the existing copy of the file.
The file `sign.jpg` already exists.
If you would like to download a new copy of the file, please rename
the existing copy of the file.
```

0. Quote and Reaction

We will skip the quote and reaction so that you have a little more time to find a good data visualization to share with the class (see question 4 below). There will be a few additional quote and reaction papers to read later in the semester!

In [6]:
```python
# 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
import statistics

%matplotlib inline
```

1. For loops

For our first set of exercises, let's practice using for loops!

In order to explore for loops, let's look again at the American Community Survey (ACS) data. The code below loads a sample of the ACS data as a pandas DataFrame and then extracts three *lists* from DataFrame using the `to_list()` method. The three lists that are extracted are:

- `earned_incomes` : contains the amount of money made from working for each of the 10,000 randomly selected participants
- `ages` : contains the ages for each of the 10,000 randomly selected participants
- `sex` : contains the sex for each of the 10,000 randomly selected participants

These Python lists are in order, so that, for example, the first item of the `earned_incomes` list comes from the same person as the first item in the `sex` list.

Note: For part 1 of this homework (i.e., the questions about for loops), you are not allowed to use any functions in the NumPy package. Also, throughout the homework, you are not allowed to use any pandas functions (apart from the code that loads the data that is written for you). Hopefully, this will make you appreciate the usefulness of the NumPy and pandas packages and also give you some insight into how the functions in these packages were written!

In [8]:
```
acs_data1 = pd.read_csv("ACS_2017_sample_01.csv")
acs_data1.head()

# This code converts pandas DataFrame data into Python lists so tha
earned_incomes = acs_data1["INCEARN"].to_list()
ages = acs_data1["AGE"].to_list()
sex = acs_data1["SEX"].to_list()
```

Question 1.1 (8 points): Let's start by calculating the mean **`age`** of the individuals in the the data **without using the `mean()`, the `sum()`, or the `len()` functions**.

To do this, you will use a for loop to calculate the sum of all the ages in the `ages` list, which you will store in a name called `age_sum`, and you will calculate the total number of elements in the `ages` list which you will store in a name called `n`. You will then calculate the mean age by calculating `age_sum/n`. Please use the following steps to do this:

1. Create a name called `age_sum` that is initially set to the value of 0. You will use `age_sum` to store the sum of all the ages in the `ages` list.
2. Create a name called `n` that is also initially set to a value of 0. You will use `n` to store the total number of ages in the `ages` list (i.e., it will ultimately contain the value `len(ages)`.
3. Create a for loop that loops over each value in the `ages` list. Inside the for loop, add the current age to `ages_sum`, and add 1 to value stored in `n`.
4. After the for loop code, write code that calculates the mean by dividing `ages_sum` by `n`. Print out the result. You can check your answer using the `statistics.mean()` function.

In [10]:
```
# create age_sum and n, and set their values to 0
age_sum = 0
n = 0
```

```
# create a for loop that loops over the ages
for i in ages:
    n = n+1
    age_sum = age_sum + i


# calculate and print the mean
print(age_sum/n)


# you can check your work using the statistics.mean() function
print(statistics.mean(ages))
```

```
41.2415
41.2415
```

Question 1.2 (7 points) Now let's explore loop and conditional statements by calculating how many people in our sample listed their sex as male, and how many listed their sex as female.

To do this, start by creating names `male_count` and `female_count`, and set both of these values to 0. Then use a for loop to go through each value in the `sex` list, increasing `male_count` by 1 if the current person's sex is male and increasing `female_count` by 1 if the current person's sex is female.

Once you have calculated these values, please print out the `male_count` and `female_count` values to "show your work".

In [12]:
```
male_count = 0
female_count = 0
for i in sex:
    if i == 1:
        male_count += 1
    elif i == 2:
        female_count += 1
print(male_count,female_count)
```

```
4900 5100
```

Question 1.3 (5 points) Let's now calculate the average "earned income" separately for males and females. To do this, please complete the following steps:

1. Create a name called `male_income` which initially has a value of 0, and likewise create a name called `female_income` which also initially has a value of 0.

2. Use a for loop that loops over each person and adds the person's income to the `male_income` name if the current person is male, or adds the person's income to the `female_income` name if the current person is female.

3. Calculate the average income for males by dividing the sum total male income by the number of males in the data set (using the `male_count` values you created in question 1.2 could be useful here). Do the same for females.

Be sure to print out the average income for males and for females below to "show your work". Then in the answer section below, report what these mean values are and describe if these are about the values you were expecting.

Hints: There are a few different ways to solve this problem. One way is to loop over successive numbers `i` using the `range()` function. Then inside the loop you can extract the sex for the i^{th} individual and also the earned income for the i^{th} individual. You can then check the sex of the i^{th}, and add the current income to the appropriate total.

Note: remember that the values in the `sex` list and the `earned_income` are in the same order; i.e., `sex[0]` and `earned_incomes[0]` are the sex and earned income values for the first person in the dataset, etc. Also, there are no missing values in the data.

In [14]:
```
male_income = 0
female_income = 0
for i in range (0,10000):
    if sex[i] == 1:
        male_income = male_income + earned_incomes[i]
    elif sex[i] == 2:
        female_income = female_income + earned_incomes[i]

print(male_income/male_count,female_income/female_count)
```

```
32916.77020408163 18759.870588235295
```

ANSWER: The average income for male is 32917 dollar and 18760 dollar for female. It is reasonable that male makes more money than female but it shocks me that there is such a big lag.

2. Warm up writing functions: Distances

The "Euclidean distance" between two points (x_1, y_1) and (x_2, y_2) is defined as: $dist(x_1, y_1, x_2, y_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

Question 2.1 (6 points) Please write a function called `euclid_dist(x1, y1, x2, y2)` that takes in the x and y coordinates from two points, and calculates the distance between these points. Please make sure all functions you use will work on numpy arrays (e.g., don't use any functions from the `math` module).

Hint: The body of our function was one line long.

In [17]:
```python
def euclid_dist(x1, y1, x2, y2):
    x1 = np.array(x1)
    y1 = np.array(y1)
    x2 = np.array(x2)
    y2 = np.array(y2)
    z = ((x1-x2)**2+(y1-y2)**2)**0.5
    return z




# Testing your function on a 3, 4, 5 triangle (do not change this c
euclid_dist(0, 0, 3, 4)
```

Out[17]:
```
5.0
```

Question 2.2 (4 points) Does the `euclid_dist()` function you wrote above work on ndarrays of points? I.e., suppose you gave it ndarrays of x_1, y_1, x_2, y_2, where each ndarray had k elements. Does your function naturally calculate k distances for each entry in these ndarrays?

Using the `array_x1`, `array_y1`, `array_x2`, `array_y2` arrays defined below, test out your function. In the answer section, report whether it works.

In [19]:
```python
array_x1 = np.arange(1, 6)
array_y1 = np.arange(1, 10, 2)
array_x2 = np.arange(1, 6)
array_y2 = np.arange(3, 16, 3)


# run your code here...
euclid_dist(array_x1,array_y1,array_x2,array_y2)
```

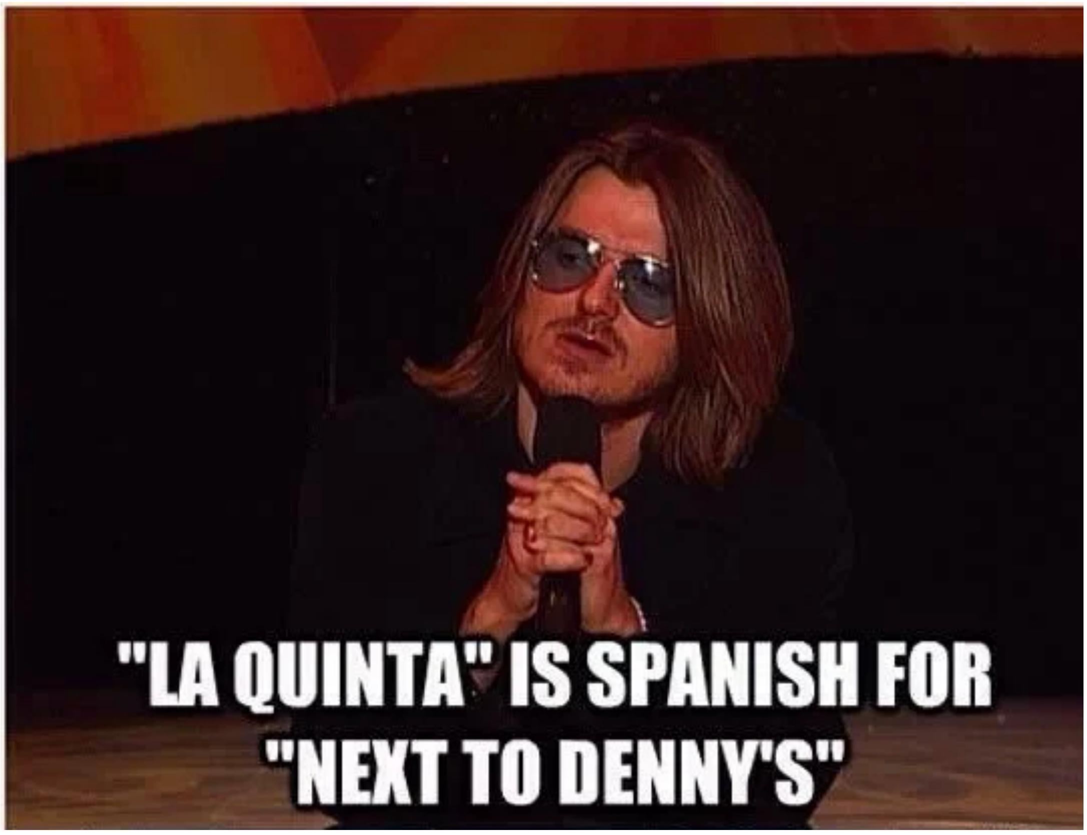
Out[19]:
```
array([2., 3., 4., 5., 6.])
```

ANSWER: It works and correctly print a k-dimension array.

3. More practice writing functions: La Quinta and Denny's

La Quinta is a hotel chain that has locations across the United States. As one can tell, the name of the hotel chain is in Spanish, and a common question that arises is what does "La Quinta" translate to in English?

The comedian Mitch Hedberg claimed it translated to "Next to Denny's". But is that accurate?



The code below loads the location of every La Quinta hotel and every Denny's restaurant in the United States. In particular, it creates two ndarrays called `dennys_longs` and `dennys_lats` that had the longitude and latitude locations of every Denny's restaurant in the United States, and also creates two more ndarrays called `laquinta_longs` and `laquinta_lats` that had the longitude and latitude locations of every La Quinta hotel in the United States

Let's use this data to examine whether "La Quinta is Spanish for next to Denny's" is an accurate statement by calculating the distance from each La Quinta location to the closest Denny's restuarant.

Credit: Several authors have contributed to the framing of this data science problem including material analyses by John Reiser and Mine Cetinkaya-Rundel

In [22]:
```
dennys = pd.read_csv("dennys.csv", index_col=0)
print(dennys.shape)
dennys_longs = np.array(dennys["longitude"])
dennys_lats = np.array(dennys["latitude"])
dennys.head()
```

```
(1641, 6)
```

Out[22]:

	address	city	state	zip	longitude	latitude
1	2900 Denali	Anchorage	AK	99503	-149.8767	61.1953
2	3850 Debarr Road	Anchorage	AK	99508	-149.8090	61.2097
3	1929 Airport Way	Fairbanks	AK	99701	-147.7600	64.8366
4	230 Connector Dr	Auburn	AL	36849	-85.4681	32.6033
5	224 Daniel Payne Drive N	Birmingham	AL	35207	-86.8317	33.5615

In [23]:
```
laquinta = pd.read_csv("laquinta.csv", index_col=0)
print(laquinta.shape)
laquinta_longs = np.array(laquinta["longitude"])
laquinta_lats = np.array(laquinta["latitude"])
laquinta.head()
```

```
(895, 6)
```

Out[23]:

	address	city	state	zip	longitude	latitude
1	793 W. Bel Air Avenue	Aberdeen	MD	21001	-76.188459	39.523216
2	3018 CatClaw Dr	\r\nAbilene	TX	79606	-99.778765	32.413489
3	3501 West Lake Rd	\r\nAbilene	TX	79601	-99.722688	32.491357
4	184 North Point Way	\r\nAcworth	GA	30102	-84.656087	34.082039
5	2828 East Arlington Street	\r\nAda	OK	74820	-96.636515	34.781805

Question 3.1 (8 points). In order to get the distance between two geographic locations (such as a La Quinta hotel and Denny's restaurant), we need a function that can compute spatial distances. While the Euclidean distance function we created above is good for calculating distances on a flat plane, the Earth is round and thus the Euclidean distance is not an appropriate measure to use (although some people would disagree and say the earth is flat).

In order to calculate the distance between longitude and latitude coordinates on a sphere, we can use the Haversine formula. Let's write a function called `haversine_dist(long1, lat1, long2, lat2, precision = 3)` which takes in two geographic coordinates `(long1, lat1)` and `(long2, lat2)` and calculates the geographical distance between the points using the Haversine formula. Your function should do the following:

1. Convert all the longitude and latitude coordinates into radians by multiplying the original coordinates by $\pi/180$. In your function, you can do this multiplication and then save the results back into the original argument names of `long1`, `lat1`, `long2`, `lat2`. *Hint:* `math.pi`

could be useful here.

2. Calculate a value `a` using the formula: $a \ = \ sin^2((lat_2 - lat_1)/2) + cos(lat_1) \cdot cos(lat_2) \cdot sin^2((long_2 - long_1)$. *Note: that sin^2 is the value you get by taking the sine of an number and then squaring the results. The functions `np.sin()`, `np.cos()` and `np.arcsin()` will be useful for this problem (be sure to use these NumPy rather than functions from other modules).*

3. Calculate the distance in kilometers using the formula: $d = R \cdot 2 \cdot arcsin(\sqrt{a})$, where R is the radius of the Earth in kilometers and is equal to a value of 6371.

4. Convert the distance from kilometers to miles by multiplying the distance in kilometers by the constant 0.621371.

5. Return the results rounded to the number of decimal places specified by the `precision` argument (which should have a default value of 3, as already specified in the function arguments).

Once you have written the function use the test code below to print the distance between the first La Quinta and the first Denny's (by first La Quinta, we mean the La Quinta at index 0). The code also prints out the addresses of these locations which you will use in the next question.

In [25]:
```
import math

def haversine_dist(long1, lat1, long2, lat2, precision = 3):
    long1 = long1 * math.pi/180
    long2 = long2 * math.pi/180
    lat1 = lat1 * math.pi/180
    lat2 = lat2 * math.pi/180
    a = np.sin((lat2-lat1)/2)**2+np.cos(lat1)*np.cos(lat2)*np.sin((
    d = 6371*2*np.arcsin(a**0.5)
    d_in_miles = d * 0.621371
    return np.round(d_in_miles,precision)




# Testing that the code gives the correct answers (do not change th


print("\nFirst La Quinta address:  " + laquinta.iloc[0]["address"] 
      + ", " + laquinta.iloc[0]["state"] + ", " + str(laquinta.iloc

print("First Denny's address:  " + dennys.iloc[0]["address"] + ", "
      + ", " + dennys.iloc[0]["state"] + ", " + str(dennys.iloc[0][

print("Calculated distance:    "  +
      str(haversine_dist(laquinta_longs[0], laquinta_lats[0], denny
First La Quinta address:  793 W. Bel Air Avenue, Aberdeen, MD, 21001

First Denny's address:  2900 Denali, Anchorage, AK, 99503

Calculated distance:    3354.115 miles
```

Question 3.2 (4 points) Almost any time you write a function, you should immediately check that it is working correctly. When developing complex software, one usually writes "unit tests" which consists of a separate set of functions that test that your code/function is working correctly. Here, we will not be so formal as to write unit tests. However, let's check that our function is giving reasonable answers by comparing the results that it returns to the results one gets from using Google maps.

The code to test your `haversine_dist()` function in the previous question calculates the distance between the first La Quinta location in our data set, and the first Denny's location in our data set, and it also prints the addresses of the first La Quinta location and the first Denny's location. Using this address information, please do the following to get the distance between the first La Quinta and the first Denny's from Google maps:

1. Go to Google maps and get the driving directions between the first La Quinta and the first Denny's using the address information shown in the previous problem.
2. Right click on the La Quinta location in MD, and select "Measure Distance" from the bottom of the dropdown menu.
3. Click on the location of the Denny's.
4. The distance between the locations should be shown near the bottom of the map.
5. In the the answer section below, please report what Google maps says the distance between these locations is, and comment on how well the `haversine_dist()` function you wrote above is working.

Hint: your `haversine_dist()` function and the Google maps results should differ by less than 10 miles. If it is off by more than this, please fix your `haversine_dist()` function before going on to the next question.

In [27]:
```python
# Calculate the percent that our function differs from the distance
(3354.115-3351.41)/3351.41
```

Out[27]:
```
0.0008071229721221597
```

Answer The google map shows the distance is 3351.41 miles and that means my function works very well.

Question 3.3 (5 points) Now use your `haversine_dist()` to calculate the distance between the first La Quinta and *all* the Denny's in the data set, and save the results to the name `first_la_quinta_distances`. If you have implemented your `haversine_dist()` correctly using NumPy, you should be able to call the `haversine_dist()` with the longitude and latitude of the first La Quinta, and the ndarrays of all the denny's longitudes and latitudes (i.e., your code to get all the distances should be one line long and you do not need to use a for loop). Also print out the first 10 distances to show that your code is working correctly.

In [30]:
```
# get the distances between the first La Quinta and all the Denny's
first_la_quinta_distances = haversine_dist(laquinta_longs[0], laqui


# print out the first 10 distances to show your code is working cor
first_la_quinta_distances[0:10]
```

Out[30]:
```
array([3354.115, 3351.734, 3259.918,  704.398,  719.405,  720.731,
        694.748,  774.312,  763.685,  664.009])
```

Question 3.4 (6 points) Let's create a function callled `dist_closest_dennys(long, lat)` that takes a longitude and a latitude location and returns the distance to the closest Denny's (of all the Denny's in the United States).

Once you have written this code, show that your code is working by printing:

1. The distance between the first La Quinta and the closest Denny's.
2. The distance to the closest Denny's from our classroom at Yale.

Also take a moment to appreciate what you have done. No matter where you are in the USA, you can now run this code to know how far away you are from a Denny's.

Hints: Functions can access data that is outside their function body (e.g., your function can access the data in the `dennys_longs` and `dennys_lats` ndarrays without needed to pass them as arguments). Our function body consisted of two lines of code.

Note: Unfortunately, the closest Denny's in the data set to our classroom has closed :(. Perhaps later in the semester we can learn out to scrape the data from the internet to get a more updated Denny's list, and/or we can work together to try to reopen that Denny's.

In [32]:
```
laquinta_lats[0]
```

Out[32]:
```
39.523216
```

In [33]:
```
def dist_closest_dennys(long, lat):
    minimum = np.min(haversine_dist(long,lat,dennys_longs,dennys_la
    return minimum



# Print the distance from the first La Quinta to the closest Denny'
print(dist_closest_dennys(laquinta_longs[0], laquinta_lats[0]))



# Print the from our classroom at Yale to the closest Denny's
print(dist_closest_dennys(-72.92101422953596,41.31686186374537))
```
```
7.891
3.121
```

Question 3.5 (6 points): We are now getting significantly closer to ruining Mitch Hedberg's joke! (if we haven't already ruined it). Let's continue with our analysis by using a for loop to calculate distance between each La Quinta and its closest Denny's. Store the result in a list named `distances_to_dennys`. To show your code is working correctly, print the first 10 elements in this list which should correspond to the distanece to the first 10 La Quinta's in our ndarrays of La Quinta longitudes and latitudes.

Hint: Our code below consisted of 4 lines including the line to print the first 10 distances.

In [68]:
```
# Calculate the distance between all La Quintas and their closest De
distances_to_dennys = []
for i,j in zip(laquinta_longs,laquinta_lats):
    distances_to_dennys.append(dist_closest_dennys(i,j))

# print the distances to the closest Denny's for the first 10 La Qu
print(distances_to_dennys[0:10])
```
```
[7.891, 0.401, 5.848, 14.322, 44.486, 3.177, 6.545, 0.462, 0.141, 1.
961]
```

Question 3.6 (5 points): Let's now create some summary statistics and visualization of these distances.

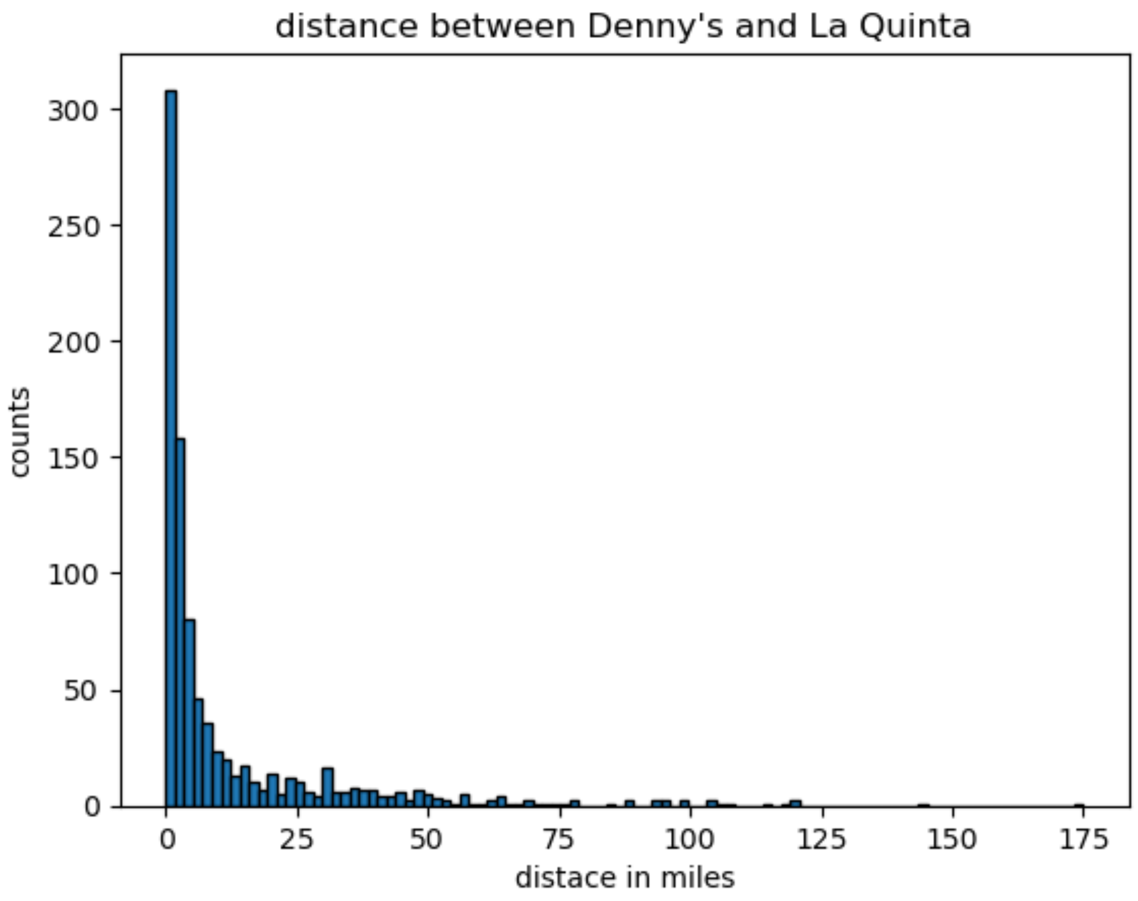
Please start by plotting a histogram of distances between all La Quinta's and their closest Denny's. Also, in the answer section, please answer these questions (and in the code section below print out statistics that support your answers):

1. In feet, what is the closest distance between a La Quinta and a Denny's?
2. In miles, what is the furthest distance between a La Quinta and a Denny's?
3. In miles, what is the mean and median distance between a La Quinta and a Denny's?

4. Based on the histogram of the data, does it make sense that the mean distance would be larger than the median distance? In 1-3 sentences, explain why.

In [82]:
```
# Create a histogram of the distances to Denny's
plt.hist(distances_to_dennys,bins = 100, edgecolor = "black");
plt.xlabel("distace in miles");
plt.ylabel("counts");
plt.title("distance between Denny's and La Quinta");

# print out statistics that support answers to the questions posed.

# 1. In feet, the distance between the La Quinta and Denny's that a
print(min(distances_to_dennys)*5280)


# 2. In miles, the furthest distance between a La Quinta and a Denn
print(max(distances_to_dennys))


# 3. In miles, the mean and median distance between a La Quinta and
print(np.mean(distances_to_dennys))
print(np.median(distances_to_dennys))
```

```
52.800000000000004
174.969
11.98822905027933
3.187
```



Answers

1. 52.8 feet
2. 174.969 miles
3. mean is 11.99 miles and median is 3.187 miles
4. Yes it does. Because the density graph is right skewed.

Question 3.7 (5 points): Finally, please answer the following questions:

1. Describe whether Mitch's joke has some truth to it; i.e., is it really the case that La Quintas are close to Denny's?
2. Briefly describe whether there are other analyses that would be good to do to better assess whether it is true that La Quinta's are close to Denny's.
3. Briefly describe other questions that would be interesting to know the answer given the data and/or given the results you have.

Answers

1. It does have some truth in it, most of La Quintas are no more than 25 miles to a Denny's. However, there are also some La Quintas that are far from Denny's.
2. I think the reason that this method is not good enough is that since La Quintas and Denny's are both famous restaurants in US, the reason that they seems to be close might lead by the high density of them. To rule out this possible explanation, it might be helpful to do the same analysis for La Quintas and McDonald's. If the result is significantly differ from the one we have, we can say that Mitch's joke might be true.
3. Does the same thing happen to McDonald's and KFC?

4. Finding an interesting visualization to share

Instead of doing a quote and reaction this week, homework activity is to find an interesting data visualization that you can share with the class.

Please search the Internet (or scan a picture from a book or magazine) to find an interesting data visualization. Once you have found the visualization, put a link to it below along with a brief description of why you found the visualization interesting.

Question 4 (4 points) Please write a link to your image and why you find it interesting.

Link to the image: https://www.data-to-viz.com/graph/ridgeline.html

Why you find it interesting: Ridgeline plot shows the density in a much more interesting and beautiful way than simply use histogram. It provides the reader a sense of flexible and etherealize, which reduce the boring and tired when reading papers.

5. 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 5.

6. 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.

LA QUINTA
INN
Denny's