When turning in homework 4, submit three files, zipped together:

1. All written portions of this as a .pdf, .doc, or .docx file (.pages is also fine for me).
2. Rename the program you used for questions 1-8 in Section 3 to <your andrew id>_link.<extension>.
3. Rename the program you used for questions 9-12 in Section 3 to <your andrew id>_theorem.<extension>.

If you did a combined program for all problems in Section 3, name it <your andrew id>_hw4_program.<extension>.

Section 1: l-diversity and t-closeness (25 points)

We discussed in class how k-anonymity can fail when *all* members of an anonymity group have *the same protected information*. Then, knowing that someone is in that group tells you their protected information, *even if you don't know which member of the group they are*.

L-diversity presents a solution to this problem, but it doesn't work perfectly either. Here's two papers that discuss l-diversity and deanonymization:

L-diversity and intersection attacks: http://www.cse.psu.edu/~kasivisw/kdd.pdf
T-closeness, skewness attacks, and similarity attacks: https://www.cs.purdue.edu/homes/ninghui/papers/t_closeness_icde07.pdf

1. (3 points) What is the *mathematical definition* of l-diversity?

2. (5 points) Give an *intuitive explanation* of l-diversity.
An example for k-anonymity might be like this:

> "K-anonymity is a property of a database in which, for each configuration of non-sensitive data present in the database, there exist at least k rows with that configuration. It's usually achieved by partially redacting non-sensitive columns, such as by trimming off some digits of a zip code, or referring to people by their first initial. It makes it so that an adversary who has all the non-sensitive information about someone in the database can't identify them to better than one of k rows."

3. (7 points) l-diverse datasets are still *vulnerable to intersection attacks*. Create a pair of *l-diverse* data sets that show this, and explain how the attack works on your tables. (HINT: Table 1 in the first paper shows an intersection attack, *but its constituent tables aren't l-diverse*.)

4. (5 points) Explain how a *skewness* attack works, and describe an example related to the topic of your databases from the previous part.

5. (5 points) Explain how a *similarity* attack works, and describe an example related to the topic of your databases from the previous part.

Section 2: Laplace Mechanism (25 points)

Consider the following table of information about the councilmembers of Cylabrador:

name	yearly_salary	righthanded
Helen	100,000	TRUE
Igor	2,000	FALSE
Jackie	40,000	TRUE

This database is protected so that *queries will return differentially private results* using the *Laplace mechanism*.

Formally, this mechanism, denoted κ_f for query function f, computes f(X) and adds noise with a Laplace distribution with mean 0 and variance 2*(Δf/ε)^2, where Δf is the global sensitivity of f and ε is the privacy parameter. For more details, check out these Wikipedia pages:

Differential privacy: https://en.wikipedia.org/wiki/Differential_Privacy#The_Laplace_mechanism
Laplace distribution: https://en.wikipedia.org/wiki/Laplace_distribution

1. (2 points) What is the global sensitivity of *count(righthanded)*?

2. (2 points) What is the global sensitivity of *max(yearly_salary)*?

3. (4 points) What would be the variance of the Laplacian distribution of the results of the query *count(righthanded)* for each of ε = 0.00001 and ε = 0.001?

4. (4 points) What would be the variance of the Laplacian distribution of the results of the query *max(yearly_salary)* for each of ε = 0.00001 and ε = 0.001?

5. (2 points) For a given level of privacy, which query will require more noise to be differentially private? Why?

6. (2 points) For a given query, what happens to the noise as the level of privacy increases? Why? Specify what it means for the level of privacy to increase.

After elections, a vacant seat is filled with a new councilmember having the following data:

name	yearly_salary	righthanded
Kevin	300,000,000	TRUE

7. (1 point) What would *count(righthanded)* return before Kevin joined the council? How about after? (Yes, this is as simple as it looks.)

8. (8 points) For ε = 0.001 and query *f=count(righthanded)*, is the differential privacy condition satisfied over the interval (2.3, 2.7] for datasets D1 and D2 representing the councilmember data before and after adding Kevin? Show your work.
(HINT: This is the same as asking whether the ratio of Pr(κ_f(D1)∈ (2.3,2.7]) and Pr(κ_f(D2)∈ (2.3,2.7]) is less than e^ε.)

Section 3: Netflix Deanonymization (35 points + 5 bonus)

The Netflix deanonymization attack has been a nice example throughout the semester, but it's also a thing you can do yourself with the techniques from this class! In this problem, we'll walk you through an example attack on a smaller set of movies.

To start, download and unzip the files here: http://www.archive.ece.cmu.edu/~ece734/homeworks/hw4-files.zip

The data:
In the folder movies, you'll see 15 .csv files, each with a five-digit number for a name. Each file represents the rating information for a certain movie, and the name of the file is an identifier number for that movie. Inside, you'll see a few thousand lines, each looking something like:

771345,2003-11-40,3

These lines consist of three entries, separated by a comma:

1. User ID (771345 on this line).
2. Date of rating (2003-11-40 on this line). We won't use this.
3. Rating (3 on this line).

Starter code:
We provide you with some starter code written in Python. This code reads in the files from the movie folder and populates the Python dictionary db. Each item of db has the user ID as a key and a value called movie-dict, which is itself a dictionary consisting of key-value pairs of movie IDs and ratings. For more information on how Python dictionaries work, consult this tutorial: http://www.tutorialspoint.com/python/python_dictionary.htm

You don't have to use our starter code, though if you want to do your homework in a language other than one of Python, C++, or Java, please check with us first. The more exotic your code, the more important good comments are.

The questions in this section are based on the paper "Provable Deanonymization of Large Datasets with Sparse Dimensions", available here: http://www.andrew.cmu.edu/user/divyasha/dss-post12.pdf

First, we will attempt to find a user in the Netflix database given auxiliary data about their movie ratings:

movie	rating	movie	rating	movie	rating	movie	rating
14199	4.5	17113	4.2	06315	4.0	01292	3.3
11977	4.2	15267	4.2	08191	3.8	16944	4.2
07242	3.9	06004	3.9	03768	3.5	03124	3.5

This auxiliary data is ratings of 12 of the 15 movies on another service *with a different scale*, so they're *perturbed from how they're rated on Netflix*.

For problems 1-8, starter code is available in link.py, and the auxiliary data is already entered as the variable aux.

1. (1 point) How many *users* are present in the database?

2. (4 points) Complete the function compute_weights() and use it to compute the weights of each movie. For each movie ID, what is its corresponding weight?

3. (7 points) Complete the function score() and compute the scores of the auxiliary information with respect to *every user's ratings* in the database. What is the *highest* score? What is the *second-highest* score?

4. (1 point) What is the user ID of the user with the highest score?

5. (2 points) Write the ratings of the user with the highest similarity score. Are these similar to the ratings in the auxiliary data?

Next, we'll look at the "Provable" part of the deanonymization paper's title. In question 4, we found the user from the Netflix data most likely to be the source of our auxiliary data. But can we say with confidence that they really are?

Theorem 2 (starts on page 8) of the paper provides us our answer. It says that given (m, γ)-perturbed auxiliary information, if the eccentricity is greater than γM, where M is the mean of the weights of each member of the auxiliary data, then the record with the maximum similarity score is the record that produced the auxiliary data.

Let's break this down. Eccentricity defined as the *difference* between the *highest* similarity score and the *next highest* similarity score.

6. (1 point) What is the eccentricity of this auxiliary data?

Our auxiliary dataset is generated using m=12 and γ=0.1.

7. (2 points) What is γM for this auxiliary data? Remember that M is the average of the weights *only of those movies that appear in the auxiliary data*!

8. (4 points) Compare the results of problems 6 and 7 and interpret this. Are we able to *confidently* say that we've linked the auxiliary information to the correct anonymized user?

But how useful is this technique in general? To see how frequently the eccentricity condition holds, and how the parameters m and γ affect this rate, we'll generate auxiliary data for each user with enough movie reviews. We want to check the eccentricity condition for each of $\gamma \in \{0.1, 0.2\}$ and $m \in \{8, 10\}$.

Problems 9-12 use the starter code available in theorem.py. You can reuse your code for score and compute_weights from link.py, so you'll mostly be working on filling out main here. If you are using the starter code, *do not change the random seed* — having that be fixed makes grading a lot easier.

The starter code has a method create_aux(r, m, gamma), which creates an (m, γ)-perturbed set of auxiliary data for a given movie record of a user.

9. (2 points) To make an (m, γ)-perturbed auxiliary, a record must have *at least m* entries. Therefore, we will only be able to create the auxiliaries for users with at least 8 or 10 movie reviews. Filter the users accordingly. How many users have *at least 8* movie reviews? How many have *at least 10*?

Our procedure is as follows:

1. For each combination of $\gamma \in \{0.1, 0.2\}$ and $m \in \{8, 10\}$, we compute the proportion of records for which the eccentricity condition is met as follows:
    1. For each record with enough reviews, check the eccentricity condition:
        1. Create auxiliary information for that record using the current γ and m.
        2. Compute the similarity scores of *all* users.
        3. Find the eccentricity.
        4. Compare it to Mγ, where M is the average weight of movies in the auxiliary.
    2. Find the proportion of records with enough reviews where the eccentricity condition holds.
2. To ameliorate the randomness of the process of auxiliary creation, run this process 5 times for each combination of γ and m.

10. (7 points) For *each combination* of γ and m, for what proportion of users *with sufficient reviews* does the eccentricity condition hold?

11. (1 point) Plot these proportions as a bar graph, similar to that on page 15 of the Provable Deanonymization paper.

12. (3 points) Why does changing γ and m have the effects they do on this proportion?

13. (5 BONUS points) We'll look at your code and give you up to 5 bonus points for good style, including documentation, efficiency, and overall clarity.

Section 4: Algorithmic Fairness (15 points + 15 bonus points)

The land of Cylabrador is not immune to social problems. For centuries, superstition has held that left-handed people are sinister and untrustworthy, and so there's considerable prejudice against hiring them for accounting roles for fear of embezzlement. Perhaps because of the pervasiveness of this stereotype, throughout the nation, left-handed Cylabradorans really are somewhat more likely to embezzle than right-handed ones, at a rate of 5% over their career as opposed to 3% for right-handed Cylabradorans.

Cylabrador Medicine Universal (CMU) is always looking hire accountants, and for the last twenty years, they've been using an algorithm that researchers fear may reflect this bias. The algorithm assigns each applicant a trust score between 0 and 100, where a higher score is more trustworthy, and CMU will hire anyone with a trust score over 75 (this is the last step of the interview process).

Detractors of the algorithm point out that among people with high enough trust scores to be hired, left-handed employees actually have proven to have a lower rate of embezzlement than right-handed employees, suggesting that the algorithm holds left-handed applicants to an unfairly higher standard than right-handed ones. The algorithm's developers counter that it doesn't use handedness to come to its conclusion, and anyhow, the algorithm is more accurate for left-handed people.

The slide deck here is a good overview, though independent investigation is encouraged: https://policylab.stanford.edu/projects/defining-and-designing-fair-algorithms.html

You don't need to write a huge amount to answer these questions; just 2-4 sentences is all. There's not one right answer for these.

1. (2 points) Give an advantage and a disadvantage of using an algorithm instead of a human to determine trustworthiness for hiring purposes.

2. (4 points) Suppose that the critics of the algorithm are *correct* about the relative rates of embezzling between people above the score cutoff. Why might this *not* support their claim that the algorithm is unfair?

3. (4 points) Suppose that the supporters of the algorithm are *correct* that the algorithm doesn't use applicants' handedness and that the algorithm really is more accurate for left-handed people. Why might this *not* support their claim that the algorithm is fair?

4. (3 points) Left-handed Cylabradorans are actually *investigated* for embezzling *significantly more than right-handed ones*. What effect might this have on the usefulness of a machine learning system to measure trustworthiness?

5. (2 points) Ambidextrous people are a small minority in Cylabrador, but they embezzle at a *very low rate* of 1% over their careers. What's a *possible* way they might get treated by a trust-scoring algorithm? (This is very open.)

6. (15 BONUS points) Write a short essay (250-500 words) discussing your thoughts on this situation. Your essay should at least address the following questions:
- How might CMU hire equitably?
- What does it mean for them to hire equitably?
- How can CMU balance equity with concerns about embezzlement?

Make your assumptions clear — there's quite a few ways to interpret the situation laid out in this section. It's okay to reuse content from your answers to previous questions. If you draw on external sources, including those linked earlier, be sure to cite them.

We're not grading you on the quality of your prose, but we do have to be able to understand your arguments.

Rubric:

4 points — Assumptions are clear and explicit
8 points — Insightful analysis of the situation
3 points — Proper citations