Task1

- **Address**: JARO DISTANCE
I used Jaro distance instead of Jaro-Wrinkler in this field because the later gives more preference to pairs with common prefixes. But in this case address can have common prefixes but can be completely dissimilar.
For example, "*Pinot Bistro 12969 Ventura Blvd. Los Angeles*" and "*Pinot Hollywood 1448 N. Gower St. Los Angeles*" have the same prefix "*Pinot*" but addresses are completely different.

- **Cuisine**: JARO-WRINKLER
I used Jaro-Wrinkler in this field because apart from the matches, some cuisines may have common prefixes and did not want to penalise them.
For example, "*American*" and "*American (New)*" have the same common prefix and refer to the same cuisine. So wanted to boost the probability of it matching.

- **Phone Number**: EXACT MATCH
After preprocessing the phone number by removing '/' and '-' and also 'or' and 'and', I did an exact match on the phone number. This is because restaurants having the same phone numbers are set to be similar. The possibility of 2 restaurants having the same phone number is quite low. The possible values can either be 1 or 0

Task2

I have calculated the weighted average of the three fields to come up with a final similarity score. After doing this I took the highest 112 similarities and wrote the IDs to the output file. The weights to each of the fields are summarised below:

- **Address**: 0.5
- **Cuisine**: 0.2
- **Phone Number**: 0.9

The rationale behind these weights is that, for *Address,* I want to give it a medium sized weight, which is not too high or low. This is because I want to avoid the False Positives in this case.

For *Cuisine,* I have used a very low weight because there can be too many False Positives in this case as cuisine for two separate restaurants can be same but they could be completely different. I do not want cuisine to have an ill-effect on the scoring function.

For *Phone Number,* I have used a fairly high weight because this field is what matters the most when it comes to data linking between these 2 files. The possibility of 2 restaurants having the same phone number is quite low.

Task3

I have implemented the approach of hashing top speed up the code and to get less

number of comparisons. I have hashed based on area code in the phone number. After hashing, I have only computed the pairwise similarities between inside each bucket. This helped reduce the number of comparisons from 176,423 to 34,229 (80% decrease!!!). The rationale behind this is that the restaurants have to be in the same city in order to be similar, hence hashing on area code was logical to speed up. The time complexity also halved!

It **does not miss** any record in the output file because the weight of phone number as an exact match is very high, which contributes heavily to the scoring function. So solely comparing only the records in the same buckets of area code is sufficient for this particular task.