**Description:**

This data set simulates the music history of users, with each record being their name followed by their streaming platform, main device, their most listened song, artist, and album, and the amount of minutes they listened to music in the current year. Many companies like Spotify and Apple music use them to track their listeners. However, my database transcends this by not limiting the tracking to one streaming service, and broadening the data to track the listening history of people from all streaming services.

By the way, the screenshots show the data printed instead of in a file because I coded it to print the first and last 15 elements, but I still stored all 100 million data records in a file

**Proposal:**

Companies like Spotify use this data to recommend other artists and music to their users and also to release their popular once-yearly Spotify wrapped, which shows users their listening history and amounts of the year, while showing them what percentage of listeners they are (compares their listening minutes to others).

A potential algorithm I could implement is findPopularService(). This algorithm would have a dictionary where each Streaming service (key) would have an integer count of users. The algorithm would traverse through the data and for each user, get their streaming service and then add one to that service count in the dictionary. Finally it will return the highest of the counts.

Another algorithm I could implement is a simple getUser(name) which would go through the data and return their listening history. Depending on whether I put my data into a dictionary, list, linked list, or any other data structure, the algorithm would have different times. Putting it into a dictionary would be ideal as the key would just be the user name and the value could be a list of the data.

A third algorithm I could implement is to topArtistPerService(service). Taking in a streaming service as an input. This algorithm, using a hashmap or dictionary would tally up the artists and then return the artist with the most counts.

A final algorithm I could implement is to do what spotify has done, and get the percentile of listeners a person is in with getListenerPercentile(artist, minutes). This algorithm would get every listener of a certain artist and for each person would add a count to the total and also the lower variable if their minutes is lower than the input parameters minutes. Then it would return the lower/total \*100% and return it.