# Fall 2026 COP 3330 Program #6: Billboard Top 100,000

Music is filled with lists of the Top 10 or Top 40 or Top 100, or more. For this program, you'll read in information about songs, and then sort those songs based on several criteria. (Our chart will have to be the best, so we'll have to come up with unique comparison criteria to compare songs.)

## Song Information

For each song you'll have the following information:

Title
Artist
Year
Sales
Streams

You will create a Song class that implements Comparable<Song>. The criteria for comparing songs is as follows:

1. First criteria is based on sales. A song that has more sales than another song comes before it in sorted order.

2. For any songs with the same number of sales, break ties by streams, with more streams coming first.

3. Finally, if those are both tied, break ties based on lexicographical order of the title. (So the title "Choosin' Texas" would come before "Patient Zero")

## Input Warning

You will read your input from standard input, but not need to prompt the user to enter it. Since song titles can have spaces in them, you must only call the **nextLine()** method from the Scanner class and no other method. As necessary, you'll have to create a StringTokenizer object, call the **nextToken()** method on it and use methods such as **Integer.parseInt().**

## Input Specification

Your program will get tested on separate input files (which you can test by piping the input file to standard input as shown in class). Each input file will use the following specification:

The first line of the input file will have a single positive integer, $\boldsymbol{n}$ ($1 \leq \boldsymbol{n} \leq 10^5$) representing the number of songs in the input.

The information about the songs follows. The information for each song will be provided in three lines. The first line will have the Title of the song only. The second line will have the artist of the song only. The third line will have three integers separated by spaces: $\boldsymbol{y}$ ($1800 \leq \boldsymbol{n} \leq 2026$)**,** the year the song was released, $\boldsymbol{x}$ ($1 \leq \boldsymbol{x} \leq 10^9$) the number of sales of the song, and $\mathbf{y}$ ($1 \leq \boldsymbol{y} \leq 10^{12}$)

the number of times the song was streamed. **The number of sales will be LTE to $10^9$, but the number of streams may be as high as $10^{12}$, meaning that the streams have to be stored in a variable of type long.** (Note: This is intentional as I suspect songs have indeed been streamed more than 3 billion times and that number can't fit into an int variable.) You'll also discover something rather interesting when trying to work on the compareTo method due to this. I'll let you figure it out on your own...

**Output Specification**
You will create multiple sorted lists:

1. You will read in the data and sort all of it based on the criteria described.
2. You will separate out the data by year and create a separate sorted list for every year that appears in the file.

You will first output the sorted global list, outputting a single line for each song in the following format:

```
#. Title, Artist Year Sales Streams
```

# represents the 1-based rank of the song. The rest of the information is exactly as provided in the input.

This will be followed by each non-empty list by year. For each of year where there was one song or more, output the year on a line by itself. Follow this with the sorted list for that year, using the same format described above for each listing.

**Sample Input/Output**
Two sample runs are included in the files:

```
billboard01.in billboard01.out
billboard02.in billboard02.out
```

**Deliverables**
**Please submit two files, Song.java and Billboard.java** for your solution to this problem via WebCourses by the designated due date.

Please make sure to include a header comment and internal comments in your code and indent when appropriate.