John Botonakis
CS-410
03/29/25

Lab 2: Bell Choir

Explanation of the Program:

When running the ANT build, you are executing the main method in the Tone class, which serves as the entry point for everything. First, the Tone class initializes the audio output system (SourceDataLine) and loads music from a file, which can be specified via a command-line argument, and validates the file for any errors. If it has errors, it will not play, and will display a specific message as to why it won't. After parsing the notes into the "loadedSong" list, it creates and starts Members on separate threads, each unique in the note they play. After creation, the Members are immediately told to wait, while the Constructor is made on his own thread. The Conductor is responsible for the performance; It identifies each note in the song while also identifying the Member who is assigned to play it. It signals using triggerPlay, which also uses notify(), to tell the Member to ring his bell to the shared SourceDataLine, then goes back to waiting until it's called on again. The Conductor sleeps the thread between notes, to maintain the correct tempo and rhythm of the song. After every note has been processed and played, the main thread signals every Member and the Conductor to stop, joins their threads together, then exits.

Challenges

Starting out, I actually thought I had this in the bag within a week, because I was able to create the Conductor and the Member classes relatively quickly! However, as I continued on, I separated BellNote from Tone, I started making Test classes that ended up not effectively testing what I had, and thought WAY too much on how exactly it was supposed to work. Ultimately I had to reset to a point where I was comfortable, and had to take it slow, building each functionality piece by piece. I knew what the end product should've been, but the finer details, as usual, escaped me. All in all, I am pretty proud of my here and it was pretty similar to the Juice Bottling lab.

Meeting the Requirements

I believe that I meet all the requirements for this program. With the loadSong method in Tone reading the song from a file, I am able to not only read a file containing a list of Bell Notes, but also validate that they are not broken. When calling startMembers, there is exactly *one* member per unique note. Continuing on, Member extends Thread, allowing distinct thread instances to be created and started, with the logic being *inside* the run() method. The Conductor thread's playSong conducts the performance based on the loadedSong list, and processes it sequentially, triggering only one Member per note, then pausing them using Thread.sleep(). Only after the Member is sleeping, does it move onto the next note in the list. Each Member pulls the noteLength and plays for that specific amount of time.