Zhengyang Zhou
COSI105b
Pa-ims

Brief introduction
A simple command-respond tool for DJ to use

Getting Started
Run “riby ims.rb” on terminal
As for testing file, please gem install o_stream_catcher first, which I have already included in gem file.

Design Solution
There are three parts: the ims (driver class containing the loop), the manager to handle the client request and the warehouse to store the relevant information. In particular, the warehouse is the attribute of manager, and the manager is the attribute of ims.

Class IMS : get input from the user repeatedly until user type “exit”
Class Manager : Handle the client request. If I put all the functions in Manager class, there will be more than 100 lines. So I separate add, info, and search into another three sub manager classes.
Class Warehouse: Responsible for YAML file reading and saving, so that it can feed the manager the needed information.
Class Track: track object with track name and singer id.
Class Artist: artist object with attribute name, id, and song list

New Idea:

1. Create reverse mapping/hashing for searching.
   When it comes to mapping between artist name and id, I create two maps. One is name-id while the other is id-name. This can enable it to only take O(1) time to finish both direction of searching. However, it will come with a price with extra O(n) space.
2. Use YAML instead of PStore
   Though PStore is much quicker, but it is not readable. So I pick YAML instead.
3. Add a search function to make the system more user friendly
   According to the problem prompt, the info function is to retrieve the information of a song by track number and an artist by id. However, the user does not necessarily know the corresponding track number or id. So I add a function called search to enable user to search the track or artist by name.

Command introduction:
Help - display a simple help screen. This is a text message, multi line, that explains the available commands. Sort of like this list.

Exit - save state and exit. The effect of this is that when the app is run again, it is back to exactly where it was when you exited. What this amounts to is basically to make sure the tracks and artists and their info have all been saved.

Info - display a high level summary of the state. At minimum, the last 3 tracks played, and a count of the total number of tracks and the total number of artists. You can elaborate if you want.

Info track - Display info about a certain track by number. e.g. info track 13

Info artist - Display info about a certain artist, by id. e.g. info artist jo

Add Artist - Adds a new artist to storage and assign an artist id. Default artist id is the initials of the artist. If this is ambiguous then another id is automatically assigned and displayed. e.g. add artist john osborne

Add Track - Add a new track to storage. add track watching the sky turn green / jo

Play Track - Record that an existing track was played at the current time, e.g. play track 13.

Count tracks - Display how many tracks are known by a certain artist, e.g. count tracks by jo

List tracks - Display the tracks played by a certain artist, e.g. list tracks by jo

Search track - Display the track number by the track name, e.g search track my heart will go on

Search artist - Display the artist id by the artist name, e.g search artist celine dion

Testing:

ims_test.rb is the testing file