CS 521

Aaron Taylor

Prof. Orsini

12/13/23

Project Overview

Spotify is one of the leaders in the realm of music streaming apps through innovative features like *Blends a* feature that allows two users to generate a playlist through common songs, artists, and genres. This Python project is a layman version of a *Blend* allowing users to create a playlist of 10 songs based on an input of three music genres.

I decided to encapsulate two private members in the initialization of the program *data & prompt*. A CSV file is inputted and used as a private attribute called *data* that is comprised of three columns: artist, song, genre. The songs and artist came from a web scrape of the Billboard Top 100 chart. The Billboard website only includes data on the artist name and song name, so the genre was manually created in a separate file.

Prompt is the other private attribute that prompts the user to input the three genres they want the playlist to consist of. If the user enters a genre that is not in the CSV, then the program will enter a *while loop* having the user select from a group of genres until they correctly enter three valid genres.

The data is from a CSV file with rows and columns interpreted as arrays that get added to a list based on a value. For this project, I used the genre variable since that is where the data has commonality. Placing the songs in lists and storing the lists in a dictionary allows the program to enter and retrieve songs based on genre. This was done in the private method "__song_genres."

The public method "Playlist" is where the playlist is generated based on the dictionary created in "__song_genres." Genre validity was important, so I decided to use try and except blocks that return "item [i] not in dictionary" while testing the user input for prompt. Iterating through the user's prompt is next in creating the playlist. The *Enumerate* method creates a count of the order in which the user entered the genres. *Random* method is used to randomly select a song from the genre dictionary, and to ensure that songs are not duplicated they are wrapped in a *tuple* and checked against "included_songs" which is a *set* to make sure it has not already been added to the playlist. The variable "song_counts" is a list of integers that dictates the number of songs per inputted genre that will be in the playlist.

Printing the playlist required a *for loop* iterating through the list/array "user_playlist" producing the song and artist (elements [0] and [1] respectively).

Conducting proper assert tests to check the length of the playlist and that each song is unique expresses the robustness of the program. The first thing was to check if the length of the

playlist is equal to the “expected_song” which contained the same integers (5, 3, 2) as the playlist method. If the playlist contains more than 10 songs an *Assertion Error* will be shown. To test duplicates in the playlist I add a tuple of each song in the playlist to a set variable “unique_songs” since set methods cannot carry duplicates.

This was a fun project to work on. I know the idea of a playlist is not original the thought process was. Being able to tinker with list data structures inside Python is a valuable skill for industry work of data scientists or data analysts. In the future, I would like to use artificial intelligence or machine learning to create programs that can recommend songs or artists that a user might like based on their inputs.