Randomized Playlist Generation in Python AI1110

Edward Nathan Varghese
CS22BTECH11020

I. Introduction

This report summarizes and analyses my implementation of a randomized playlist generator in Python using Numpy, Tkinter, Pygame and Playsound modules.

II. Code Explanation

- The beginning of the code has import statements for the required modules.
- `Construct()` is the randomizer and list populator. It uses the uniform distribution and returns a float from 0 to 20 which is passed to the `ceil()` function to return an integer between 1 and 20. It then populates a list `songnums` with a random sequence of numbers from 1 - 20.
- A class `Playlist` is created and the Tkinter window and widgets are initialized along with the functions needed for the code such as `playsong()`, `autoplay()` and `pause_resume()`.
- The buttons on the main window call their respective functions which are breifly descibed below.
- `playsong()` plays the next song in `songnums` using `pygame.mixer.music.play()`.
- `autoplay()` ensures continuity of the playlist and makes sure the next song plays after the current one has ended.
- `pause_resume()` controls the state of the songs, whether they are being played or not. This is done using the `pygame.mixer.music.pause()` and `pygame.mixer.music.unpause()` statements.
- The GUI layout is organized using the grid system, and the buttons are placed in different rows and columns within the root window.
- Finally, we create the root window and an object of `Playlist` called `MyPlaylist` is created and the main event loop is started using `root.mainloop()`.

III. Output

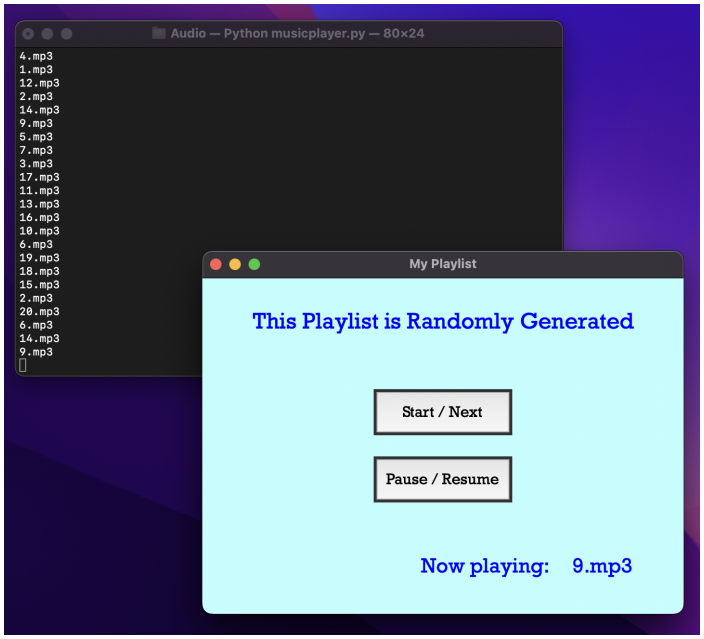

Fig. 1. GUI Screenshot

IV. Conclusion

This code is a basic implementation of a Randomised Music Playlist in Python using Numpy, Tkinter, Playsound and Pygame modules.