```
def play_random(music_path):
    try:
        music_listing = mp3gen(music_path)
        music_playing = random.choice(music_listing)
        tts("Now playing: " + music_playing)
        music_player(music_playing)
    except IndexError as e:
        tts('No music files found.')
        print("No music files found: {0}".format(e))

def play_specific_music(speech_text, music_path):
    words_of_message = speech_text.split()
    words_of_message.remove('play')
    cleaned_message = ' '.join(words_of_message)
    music_listing = mp3gen(music_path)

    for i in range(0, len(music_listing)):
        if cleaned_message in music_listing[i]:
            music_player(music_listing[i])
```

Starting from the beginning, you import the built-in os, sys, and random. Next comes the mp3gen() function. In this function, you pass music_path as an argument. You declare an empty list to hold the array of music file names. You then iterate through the files, folders, and subfolders using the os.walk() function to find all files with the .mp3 extension. When it finds the required files, it stores the names of the files along with their complete path address to the music_list variable. The function returns music_list as a list (array).

The music_player() function is written to play the music files after detecting the user's OS. The function takes file_name as an argument. Similar to what you did while building the tts() function earlier, you use the sys.platform() function to detect whether the OS is OS X or Linux. Accordingly, you create a variable named player in which you concatenate the player along with the name of the music file to play; you use either the afplay player or the mpg123 player. This player variable acts as a command that is called using the os.system() command.

Next comes the play_random() function, where you create the list of all MP3 files present using the mp3gen() function. This function takes music_path as an argument. Then you create a variable named music_playing that stores the name of a particular music file by using the random.choice() function, which operates on the music_listing list. You then pass the name of the music file stored in music_playing to the music_player() function, which plays the music. You use a try/except clause here because there may be a case when there are no MP3 files present in the music_path; this gives an IndexError, which speaks the message "No music files found."

Finally, the play_specific_music() function takes speech_text and music_path as arguments. You implement the same functionality here as in the define_subject module. So, you split speech_text to create an array of words. You then remove the *play* keyword from the array, and whatever remains, however improbable it may be, must be the name of the music file the user wants to search for. You combine the words of the array again and iterate through music_list to find a match with the name of the song the user specified. If a match is found, the music is played using the music_player() function.