Algorithm for the playlist program (Prog 4)
Siqi Wen 2021 CS162

1. Create a structure (struct music) with 5 members:

i.Artist;

ii. Title of the Song;
iii. The type of music;

iv. When you like to listen to it;

v.The link to access.

2. Create a class (class playlist) :

There will be some member function prototypes (they will be listed below) in the public section.

There will be three data members in the private section: a pointer to the music struct; the size of the array once it is allocated; the number of songs stored in the array.

3. Create a menu function. This function will show the user the menu and ask the user for an input in terms of which task they want to do. Then specific functions will be executed based on the user's choice from the menu.

4. Create an again function. This function will be used multiple times in this program. It will prompt the user if they want to do something again, and based on the user's input, the program will execute corresponding lines of code.

5. Create a constructor function to initialize the data members to the zero equivalent values. This function has no return type and no argument.

6. Create a destructor function to deallocate (release) the dynamic memory used by the class by resetting the data members to the zero equivalent values. This function has no return type and no argument.

7. Create a function to create a dynamically allocated array for the songs and prompt the user for the size that they want for the array. In this function, the program would prompt the user by asking "How many songs would you want to enter and save information?", then store the answer into the data member that represents the size of the array.

8. Create a function to read in one song. In this function, prompt the user to enter the artist, title, type, when to listen and the link.

9. Create a function to display the contents of a song. In this function, output all the contents about one song. This function will be called later in another function.

10. Create a function to read in multiple songs. In this function, prompt the user by asking if they want to read in a new song, read in the answer the user types. Then, there will be a while loop, in this while loop, call the read_a_song function, then prompt the user again by asking “Another song?”, read in the answer, and decide if this while loop will be excused again or not, when the user is done entering information for songs, calculate the number of songs and store it in a variable for the data member for the number of songs stored in the array.

11. Create a function to display all songs. This function displays the contents of all songs. In this function, create a for loop, in this for loop, call the display_a_song function, then every song will be displayed one by one.

12. Create a function that is called “is_match”. In this function, it would prompt the user for an input in terms of what “type” of songs they want to, if there is such a “type”, the function would return “true”; otherwise, “false” will be returned.

13. Create a function to display all songs that match a particular choice. “is_match” function will be called in this function. If “true” is returned, there would be a for loop to iterate over the array, in this for loop, compare each member’s “type” with the user’s input, output all the songs that have the same “type” information; if “false” is returned, the program would output the message: “Sorry, there is no songs that match with this type.”

14. Create a function called “find”. In this function, prompt the user for an input in terms of the title for a song that they want to edit. If there is such song, “true” will be returned; if not, “false” will be returned.

15. Create a function to edit the information about a particular song. In this function, output all the songs’ titles then prompt the user to input an answer in terms of which song they want to edit. Call “find” function, if “true” is returned, then ask the user which part of the song they want to edit. Based on their input, prompt the user to enter the new information for the specific part, then replace the old information with the new information the user just entered; if “false” is returned, output the message “Sorry, there is no such song.” And ask the user if they’d like to re-enter another song’s title, call the “again” function, and based on the user’s input, the program will decide if to continue asking the user for an input or stop this step.