**Song Player**

Hi, PPTI 16 fellows. In purpose to practice on how you must think when creating a program in a computer using functions, recursions, sorting and searching data, file processing, also struct and union, this practice case will help you to translate from algorithm and procedures to source code in C programming language. Hope this helps!

When the program has been run, here’s the program flow that you need to create:

* In the beginning of program’s execution, **read** “songs.txt” **file** and import playlist(s) and song(s) data. The file contains song(s) data with following format:

**[song name]#[song artist name]#[song duration]#[playlist name]**

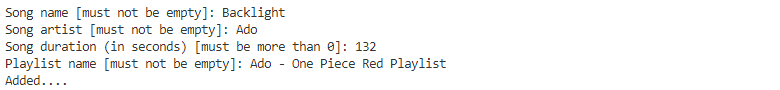
* Then, several **menu options** will be **shown**. Each of menu represent data manipulation for songs and playlists. Ask user to **input** **menu option number**.

Background pattern

Description automatically generated with low confidence

**Figure 1 Main menu**

* If user inputs:
  + **Menu 1** (Add new song), do:
    - **Ask** user to input:
      * **Song name**, **validate** the input must be **not empty.**
      * **Song artist name**, **validate** the input must be **not empty.**
      * **Song duration**, **validate** the input must be **more than 0**
      * **Song playlist name**, **validate** the input must be **not empty.**
    - Then, **save** the playlist data and song data **in array. After that**, **sort** the playlist and song data **using merge sort** or **quick sort** algorithm**.**



**Figure 2 Add new song menu**

* + **Menu 2** (Show all songs), do:
    - **If** playlist and song data is **empty, print “Empty…”** message**.**
    - Otherwise, print playlist and song data with following format.

[playlist 1 name]

[song 1 name] – [song 1 artist name] – [song 1 duration] seconds

[song 2 name] – [song 2 artist name] – [song 2 duration] seconds

……

[playlist 2 name]

[song 1 name] – [song 1 artist name] – [song 1 duration] seconds

[song 2 name] – [song 2 artist name] – [song 2 duration] seconds

……

……

A picture containing graphical user interface

Description automatically generated

**Figure 3 Show all songs**

* + **Menu 3** (Search song(s)), do:
    - **Ask** user to input:
      * **Search query**, **validate** the input must be **not empty**
    - Then, **print playlist(s) and song(s)** where the song name contains inputted search query.

Background pattern

Description automatically generated with low confidence

**Figure 4 Search song**

* + **Menu 4** (Exit), do:
    - **Save** playlist and song **data in** “songs.txt” **file** with following format:

**[song name]#[song artist name]#[song duration]#[playlist name]**

* + - Then, **close** the **application**.