**План за изпълнение на проект „Музикална библиотека“ по СД**

*2 курс, 1 семестър, 2020/2021 година*

1. ***Основни обекти***
   * ***потребители***
     + потребителско име
     + парола
     + пълно име
     + дата на раждане
     + любими жанрове
     + плейлисти
   * ***песни***
     + име
     + изпълнител
     + жанр
     + албум
     + рейтинг
2. ***Функционалности***
   * ***първоначалнно***
     + “help” меню
     + вход (с име и парола)
     + регистрация (всеки може)
   * ***след успешен вход***
     + промяна на данни (вкл. любими жанрове)
     + добавяне на песен
     + задаване на рейтинг на песен
     + генериране на плейлист по избор от критерии:
       - рейтинг по-голям от:
       - включване на жанрове
       - изключване на жанрове
       - само любими жанрове
       - преди година
       - от година
       - след година
     + записване на плейлист под име
     + зареждане на плейлист по име
     + показване на информация за всички песни от зареден плейлист
3. ***Класове, атрибути***
   * ***User*** *//първоначално почти всичко по 20 (19 символа)*
     + string username
     + string password
     + string full\_name
     + string birth\_date // DD.MM.YYY
     + string fav\_genres[10]
     + string playlists[10] //запазва само имената max 10
   * ***Song*** *//първоначално почти всичко по 20 (19 символа)*
     + string name
     + string artist
     + string genre
     + string album
     + int release\_year
     + pair<int,int> rating
   * ***Playlist***
     + string name
     + string songs[20] *//определен размер - 20*
   * ***Library***
     + User curr\_user
     + AVLTree all\_songs
     + Playlist curr\_playlist
   * ***Commands*** *(static)*
   * ***AVLTree***
4. ***Работа с файлове***
   * ***избор на файлов формат за съхранение на*** 
     + *потребители*
       - ИЗБОР: **txt file**
       - триене/добавяне/презаписване на потребител  
         (презаписване на целия файл)
       - всеки атрибут на нов ред
     + *песни*
       - ИЗБОР: **txt file**
       - данните изцяло се прочитат от файла, след което се записват в „дърво“
       - при изход данните от дървото се презаписват във файла
       - няма директна работа с файла
       - нямаме нужда от избирателен и директен достъп до съдържанието на файла
       - човешки четими данни
       - обхождане с последователен достъп
       - единен формат на данни от тип “Song”
   * ***функции***
     + *потребители*
       - записване във файла
         * …\n (username)
         * …\n (password)
         * …\n (full name)
         * \_.\_.\_\n (birth date)
         * …**,**…**,**…\n (fav genres)
         * “name” {…**,**…**,**…} “name” {…**,**…**,**…}\n (playlists)
       - прочитане от файла
         * >>username
         * >>password
         * getline (full name)
         * >>(birth date)
         * getline (string) ->fav genres
         * geltine(string) ->playlists
       - промяна във файла
         * презаписване на целия файл
         * if (line == искаме промяна) -> промяна
       - търсене във файла
         * всеки обект – 6 реда
     + *песни*
       - записване във файла
         * …\n (name)
         * …\n (artist)
         * …\n (genre)
         * …\n (album)
         * ----\n (release year)
         * \_ \_\n (rating)
       - прочитане от файла
         * getline(name)
         * getline(artist)
         * getline(genre)
         * getline(album)
         * >>release\_year
         * getline(string) -> pair<int,int> rating
       - промяна във файла
         * презаписване на целия файл
       - търсене във файла
         * всеки обект – 6 реда
5. ***Функции / методи***
   * ***User***

//private

* + - copy(const User& other)
    - clear()

//public

* + - User(const User& other)
    - User(const string& user, const string& pass)
    - User& operator=(const User& other)
    - ~User()
    - set\_username(const string& \_username)
    - set\_password(const string& \_password)
    - set\_full\_name(const string& \_full\_name)
    - set\_birth\_date(const string& \_brth\_date)
    - add\_genre(const string& genre)
    - remove\_genre(const string& genre)
  + ***Song***

//private

* + - copy(const Song& other)
    - clear()

//public

* + - Song()
    - Song(const string& \_name, const string& \_artist,

const string& \_genre, const string& \_album,

int \_release\_year)

* + - Song& operator=(const Song& other)
    - ~Song()
    - set\_Rating(int rate)
    - get\_rating()
    - get\_name()
    - song\_info()

* + ***Playlist***
    - add\_song(const Song& other)
    - remove\_song(const string& \_name)
    - change\_name(const string& \_name)
    - all\_songs\_name()
    - all\_songs\_info()
  + ***Library***
    - ***//constructors***
    - get\_user()
    - get\_playlist()
    - add\_user(const User& other)
    - add\_playlist(const Playlist& other)
    - add\_song(const Song& other)
    - remove\_user(const User& user)
    - remove\_playlist(const Playlist& user)
  + ***Commands (static voids)***

//main functions

//general

* + - welcome()
    - help()

//user

* + - sign\_in()
    - sign\_up()
    - change\_data()

//song

* + - add\_song()
    - rate\_song()

//playlist

* + - generate\_playlist()
    - save\_playlist()
    - load\_playlist()
    - show\_all\_info()

//helper functions

//user

* + - sign\_in\_helper(const string& un, const string& pw)
    - sign\_up\_helper(const string& un, const string& pw)
    - change\_data\_helper(int choice, const string& input)

//song

* + - add\_song\_helper(const string& name, const string& artist,

const string& genre, const string& album,

int release\_date)

* + - rate\_song\_helper(const string& name, int rate)

//playlist

* + - generate\_playlist\_helper(const string& input)
    - save\_playlist\_helper(const string& name)
    - load\_playlist\_helper(const string& name)
  + ***AVLTree***
    - ***//public***
    - AVLTree ()
    - AVLTree(const AVLTree& other)
    - AVLTree& operator=(const AVLTree& other)
    - ~AVLTree()
    - isElement(T value)
    - size()
    - search(T value)
    - insert(T value)
    - remove(T value)