C프로그래밍 및 실습

플레이리스트 추천

진척 보고서 #1

제출일자: 11/26
제출자명: 김도형
제출자학번: 231630

1. 프로젝트 목표

1) 배경 및 필요성

음악 플랫폼 수가 급격히 증가하면서, 음악 플랫폼 간의 경쟁이 치열해짐. 최근 이용자들은 자신의 취향에 맞는 노래를 자동 재생하여 주는 것을 선호함. 여러 음악재생 플랫폼들 사이에서 살아남기 위해서는 사용자들을 구체적으로 분석하고 파악하여 이용자들에게 맞는 노래는 추천해주어야 한다고 생각

2) 프로젝트 목표

이용자들의 나이, 이용 시간대, 선호하는 장르 등을 파악하여 이용자에게 최적의 노래를 추천해 주는 것을 목표로 함

3) 차별점

기존에 존재하는 음악 플랫폼들은 인기차트나 새로운 곡을 소개하는 형식으로

되어있지만, 이 프로그램을 통하여 취향이 비슷한 노래를 추천받고, 선호하는

장르가 비슷한 사람들의 플레이리스트를 공유 받을 수 있다.

2. 기능 계획

1) 기능 1: 재생목록 삭제 및 추가

가장 기본적인 음악 플랫폼의 기능. 사용자들에게 추가할 노래, 삭제할 노래, 수정할 노래, 노래의 장르를 입력 받고 저장 및 출력. (반복문, 함수, 2차원배열, 조건문, 2차원 배열 사용 +포인터 사용예정)

2) 기능 2: 사용자 5명에게 정보를 입력 받고 각자의 재생목록 생성

이용자 5명에게 이름, 선호 장르, 나이, 이용시간 등을 입력 받고 각자의 플레이리스트를 생성하고 관리하게 함 (2, 3차원 배열, 반복문, 조건문, 함수 사용)

3) 기능 3: 장르별로 가장 인기있는 노래 알려주기

이용자들에게 노래를 입력 받을 때 겹치거나, 가장 많이 나온 노래를 인기있는 노래로 인식하고, 장르별 인기노래로 출력 (반복문, 조건문, 함수, 배열, 포인터)

가장 인기있는 장르 출력, 및 장르에서 가장 인기있는 노래 출력.

(사람 수, 장르 모두 늘릴 예정)

3) 기능 4: 중복제거 및 재생 횟수 출력 (순위 시스템)

가장 인기있는 장르 출력 및 가장 인기있는 장르별 노래 출력 및 재생횟수 기록, 노래 검색 기능, 이미 재생목록에 있는 노래라면 추가할 때 문구를 표시하고 덮어쓰는 코드 구현. (반복문, 조건문, 함수, 배열, 포인터 사용 예정)

(계속 바뀔 거 같습니다)

3. 진척사항

1) 기능 구현

(1) 재생목록에 노래/장르 추가, 삭제, 수정, 종료

- 입출력: 선호하는 장르, 노래 제목, 매뉴얼 번호 입력, 사용자 이름

- 설명: 사용자 5명의 선호 노래와 장르를 입력 받고 플레이리스트 관리 목록을 만듦.

#define을 이용하여 노래 개수와 이름 길이 등을 정의하고, 장르와 노래, 이름 등을 입력 받을 변수들을 2차원 배열을 이용하여 선언하였다.

Addsong, delsong, printsong, recommendsong 함수를 통해 재생목을 관리하는 코드를 작성하였다.

Add song에서는 노래 이름과 노래 장르 사용자 번호를 넘겨주며 노래를 입력 받고, 그 노래를 해당 장르에 추가하는 코드로 구성하였다. Printf와 scanf_s를 이용하여 입력받고 Strcpy를 사용하여 저장하며, songCount를 1늘린다

Delsong에서는 삭제할 매뉴얼의 번호와 사용자 번호를 넘겨주며, 삭제할 노래의 인덱스를 입력 받고 삭제하면 songCount를 1 감소시킨다. for반복문 사용

Printsonglist에서는 원하는 사용자의 재생목록을 출력해주도록 작성하였다.

For 반복문을 사용하여 사용자 번호에 따른 노래와 노래의 장르를 songCount만큼 모두 출력한다

- 적용된 배운 내용 (예: scanf, printf, while반복문, (if, switch)조건문, 함수, 2차원 배열

(2) 사용자 5명에게 정보를 입력 받고 각자의 플레이리스트 생성, 관리, 추천

- 입출력: 사용자 이름, 사용자 별 플레이리스트(노래와 장르), 선호 장르와 그에 따른 노래들 목록

- 설명: 2차원 배열을 통해 장르의 이름과 개수, 그리고 선호하는 장르와 노래를 입력 받을 변수를 선언하였다. 기능 1에서 작성한 코드에 이름을 입력 받는 코드와 genre, personindex를 추가하여 추가, 삭제, 수정할 때 사용자의 플레이리스트에서 작업이 실행되게 변경.

Recommendsong에서는 입력 받은 장르가 같은 경우에 그 장르의 노래들을 추천해주도록 작성하였다. For를 이용한 이중반복문을 통해 입력한 inputgenre에 해당하는 노래들을 출력한다.

Addsong 함수에서는 입력받은 노래를 peoplefavoritesongs의 배열에 복사하여 저장하고 songCount를 셀 때 사용자마다 플레이리스트를 만들었기 때문에 songCount[personindex]++로 수정한다.

Delsong 함수에서는 노래를 삭제할 때 입력한 사용자의 플레이리스트에서 노래를 지우도록 변경하였고, 따라서 songCount[personindex]- - 로 수정한다. 또한 위에서 복사하고 저장해놓은 personfacoritesongs의 배열에서 [i-1]을 통해 지워준다.

Printsonglist 함수에서는 재생목록을 출력할 때 사용자의 번호를 입력 받고 그 사용자의 플레이리스트만을 출력하도록 for반복문을 사용하여 작성하였다. 전 과는 다르게 사용자 인덱스, 노래, 장르를 모두 출력하도록 수정하였다.

main함수에서는 for반복문을 통하여 사용자 수만큼 이름과 장르, 그리고 가장 좋아하는 노래를 입력 받는다. 전 과는 다르게, 사용자가 여러 명이기 때문에 choice를 입력 받기 전에 어떤 사용자에 대해 작업을 수행할 건지 입력 받는 코드를 추가하였다.

- 적용된 배운 내용 (예: scanf, printf, while반복문, (if, switch)조건문, 함수, 2차원 배열)

- 코드 스크린샷 (기능1)

```c
//231630 김도형 플레이리스트 프로젝트
#include <stdio.h>
#include <string.h>

#define MAX_SONGS 10 //플레이리스트 노래 개수 10개 제한
#define SONG_LEN 100 //노래 제목 길이
#define GENRE_NAME 100 //장르 이름
#define GENRE_NUM 3 //장르 개수
#define PERSON_COUNT 5 //이용자 수

char genre[GENRE_NUM][GENRE_NAME] = { "POP", "발라드", "힙합" }; //일단 장르 3개로 제한
char songs[MAX_SONGS][SONG_LEN] = { "" }; //노래제목을 입력받는 2차원 배열
int songCount[PERSON_COUNT] = { 0 }; //이용자 5명의 노래 개수 세는 변수

char peopleNames[PERSON_COUNT][50]; //이용자의 이름을 입력받는 변수
char peopleFavoriteGenres[PERSON_COUNT][GENRE_NAME]; //이용자의 선호하는 장르를 입력받는 2차원 배열
char peopleFavoriteSongs[PERSON_COUNT][MAX_SONGS][SONG_LEN]; //사용자의 선호노래를 입력받는 3차원 배열

void addSong(char song[], char songGenre[], int personIndex) { //노래를 재생목록에 추가하는 함수, 노래제목과 장르, 사용자 인덱스를 전달
    printf("노래를 입력하세요: ");
    scanf_s("%s", song, (int)sizeof(song));
    strcpy_s(peopleFavoriteSongs[personIndex][songCount[personIndex]], sizeof(peopleFavoriteSongs[personIndex][songCount[personIndex]]), song);

    // 입력한 노래의 장르를 사용자가 선호하는 장르로 저장
    strcpy_s(peopleFavoriteGenres[personIndex], sizeof(peopleFavoriteGenres[personIndex]), songGenre);

    printf("'%s' 노래가 '%s' 장르에 추가되었습니다.₩n₩n", song, songGenre);

    songCount[personIndex]++;
}

void delSong(int delIndex, int personIndex) { //노래를 재생목록에서 삭제하는 함수
    printf("%d. %s : 노래를 삭제합니다.₩n", delIndex, peopleFavoriteSongs[personIndex][delIndex - 1]);
    for (int i = delIndex; i < songCount[personIndex]; i++) {
        strcpy_s(peopleFavoriteSongs[personIndex][i - 1], sizeof(peopleFavoriteSongs[personIndex][i]), peopleFavoriteSongs[personIndex][i]);
    }
    songCount[personIndex]--;
}
```

```c
void printSongList(int personIndex) { //재생목록을 출력하는 함수 (사용자 명 장르 노래제목 모두 출력)
    printf("재생목록 for %s:₩n", peopleNames[personIndex]);
    for (int i = 0; i < songCount[personIndex]; i++) {
        printf("%d. %s (장르: %s)₩n", i + 1, peopleFavoriteSongs[personIndex][i], peopleFavoriteGenres[personIndex]);
    }
    printf("₩n");
}

void recommendSong(char inputGenre[]) { //같은 장르의 다른 사람의 재생목록의 노래를 추천해주는 프로그램
    printf("₩n'%s'와 비슷한 장르의 노래를 추천합니다:₩n", inputGenre);

    for (int i = 0; i < PERSON_COUNT; i++) {

        if (strcmp(inputGenre, peopleFavoriteGenres[i]) == 0) {

            for (int j = 0; j < songCount[i]; j++) {
                printf("'%s' 추천 노래 for %s: %s (장르: %s)₩n", inputGenre, peopleNames[i], peopleFavoriteSongs[i][j], inputGenre);
            }
        }
    }


}


int main() {
    int choice = 0; //메뉴 입력받을 변수
    int end = 0; //종료 변수
    int delIndex = -1; //삭제를 위한 변수
    char ch; //버퍼 제거..?
    int modifyIndex = 0; //수정을 위한 변수
    char inputGenre[GENRE_NAME] = "";//장르이름을 입력받을 변수

    // 각 이용자의 정보 입력
    for (int i = 0; i < PERSON_COUNT; i++) {
        printf("사용자 %d, 이름을 입력하세요: ", i + 1);
        scanf_s("%s", peopleNames[i], (int)sizeof(peopleNames[i]));
    }
        printf("음악 플레이리스트 시작!₩n");
```

```
    // 각 이용자의 정보 입력
    for (int i = 0; i < PERSON_COUNT; i++) {
        printf("사용자 %d, 이름을 입력하세요: ", i + 1);
        scanf_s("%s", peopleNames[i], (int)sizeof(peopleNames[i]));
    }
        printf("음악 플레이리스트 시작!₩n");

    while (1) {
        printf("------------------₩n");
        printf("메뉴를 입력해주세요.₩n");
        printf("1. 재생목록에 노래 추가₩n2. 노래 삭제₩n3. 재생목록₩n4. 종료₩n5. 노래 수정₩n6. 비슷한 노래 둘러보기₩n");
        printf("현재 노래 수 = %d₩n", songCount[0]);  // 여기서는 첫 번째 사람의 노래 수를 표시

        printf("------------------₩n");
        scanf_s("%d", &choice); //실행할 메뉴 입력받음

        switch (choice) { //입력받는 CHOICE에 따라 하는 일을 조건문을 통해 분류
        case 1:
            printf("사용자를 선택하세요 (1~5): ");
            int personIndex;
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) { //메뉴는 1~6사이로 선택, 아니면 오류 표기
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printf("노래의 장르를 입력하세요 (POP, 발라드, 힙합): ");
            scanf_s("%s", inputGenre, (int)sizeof(inputGenre));//장르를 입력받음
            addSong(peopleFavoriteSongs[personIndex - 1][songCount[personIndex - 1] % MAX_SONGS], inputGenre, personIndex - 1);//노래추가 함수 호출
            break;

        case 2:
            printf("사용자를 선택하세요 (1~5): ");
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) {
                printf("잘못된 선택입니다.₩n");
                break;
            }
```

```
            }

            printf("삭제할 노래의 인덱스를 입력하세요 : ");
            scanf_s("%d", &delIndex);
            if (delIndex > songCount[personIndex - 1] || delIndex <= 0) {
                printf("삭제할 노래가 없습니다.₩n");
            }
            else {
                delSong(delIndex, personIndex - 1);
            }
            break;

        case 3:
            printf("사용자를 선택하세요 (1~5): ");
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) {
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printSongList(personIndex - 1);
            break;

        case 4:
            end = 1;
            break;

        case 5:
            printf("사용자를 선택하세요 (1~5): ");
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) {
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printf("수정할 노래의 번호를 입력해주세요. (1부터 시작): ");
            scanf_s("%d", &modifyIndex);
            ch = getchar();
```

```
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printf("수정할 노래의 번호를 입력해주세요. (1부터 시작): ");
            scanf_s("%d", &modifyIndex);
            ch = getchar();
            printf("새롭게 추가할 노래를 입력해주세요: ");
            scanf_s("%s", peopleFavoriteSongs[personIndex - 1][modifyIndex - 1], (int)sizeof(peopleFavoriteSongs[personIndex - 1][modifyIndex - 1]));
            printf("새로운 노래가 추가되었습니다: %d. %s (장르: %s)₩n", modifyIndex, peopleFavoriteSongs[personIndex - 1][modifyIndex - 1], genre[modifyIndex - 1]);
            break;

        case 6:
            printf("선호하는 장르를 입력하세요 (POP, 발라드, 힙합): ");
            scanf_s("%s", inputGenre, (int)sizeof(inputGenre));
            recommendSong(inputGenre);
            break;

        default:
            printf("잘못된 선택입니다. 다시 선택하세요.₩n");
        }

        if (end == 1) {
            printf("재생목록을 종료합니다.₩n");
            break;
        }
        if (songCount[0] == MAX_SONGS) {
            printf("노래가 %d개로 다 찼습니다.₩n", songCount[0]);
            break;
        }
    }

    return 0;
}
```

110 % 문제가 검색되지 않음

- 코드 스크린샷 (기능2)

```
//231630 김도형 플레이리스트 프로젝트
#include <stdio.h>
#include <string.h>

#define MAX_SONGS 10 //플레이리스트 노래 개수 10개 제한
#define SONG_LEN 100 //노래 제목 길이
#define GENRE_NAME 100 //장르 이름
#define GENRE_NUM 3 //장르 개수
#define PERSON_COUNT 5 //이용자 수

char genre[GENRE_NUM][GENRE_NAME] = { "POP", "발라드", "힙합" }; //일단 장르 3개로 제한
char songs[MAX_SONGS][SONG_LEN] = { "" }; //노래제목을 입력받는 2차원 배열
int songCount[PERSON_COUNT] = { 0 }; //이용자 5명의 노래 개수 세는 변수

char peopleNames[PERSON_COUNT][50]; //이용자의 이름을 입력받는 변수
char peopleFavoriteGenres[PERSON_COUNT][GENRE_NAME]; //이용자의 선호하는 장르를 입력받는 2차원 배열
char peopleFavoriteSongs[PERSON_COUNT][MAX_SONGS][SONG_LEN]; //사용자의 선호노래를 입력받는 3차원 배열

void addSong(char song[], char songGenre[], int personIndex) { //노래를 재생목록에 추가하는 함수, 노래제목과 장르, 사용자 인덱스를 전달
    printf("노래를 입력하세요: ");
    scanf_s("%s", song, (int)sizeof(song));
    strcpy_s(peopleFavoriteSongs[personIndex][songCount[personIndex]], sizeof(peopleFavoriteSongs[personIndex][songCount[personIndex]]), song);

    // 입력한 노래의 장르를 사용자가 선호하는 장르로 저장
    strcpy_s(peopleFavoriteGenres[personIndex], sizeof(peopleFavoriteGenres[personIndex]), songGenre);

    printf("'%s' 노래가 '%s' 장르에 추가되었습니다.₩n₩n", song, songGenre);

    songCount[personIndex]++;
}

void delSong(int delIndex, int personIndex) { //노래를 재생목록에서 삭제하는 함수
    printf("%d. %s : 노래를 삭제합니다.₩n", delIndex, peopleFavoriteSongs[personIndex][delIndex - 1]);
    for (int i = delIndex; i < songCount[personIndex]; i++) {
        strcpy_s(peopleFavoriteSongs[personIndex][i - 1], sizeof(peopleFavoriteSongs[personIndex][i]), peopleFavoriteSongs[personIndex][i]);
    }
    songCount[personIndex]--;
}
```

```c
void printSongList(int personIndex) { //재생목록을 출력하는 함수 (사용자 명 장르 노래제목 모두 출력)
    printf("재생목록 for %s:₩n", peopleNames[personIndex]);
    for (int i = 0; i < songCount[personIndex]; i++) {
        printf("%d. %s (장르: %s)₩n", i + 1, peopleFavoriteSongs[personIndex][i], peopleFavoriteGenres[personIndex]);
    }
    printf("₩n");
}

void recommendSong(char inputGenre[]) { //같은 장르의 다른 사람의 재생목록의 노래를 추천해주는 프로그램
    printf("₩n'%s'와 비슷한 장르의 노래를 추천합니다:₩n", inputGenre);

    for (int i = 0; i < PERSON_COUNT; i++) {

        if (strcmp(inputGenre, peopleFavoriteGenres[i]) == 0) {

            for (int j = 0; j < songCount[i]; j++) {
                printf("'%s' 추천 노래 for %s: %s (장르: %s)₩n", inputGenre, peopleNames[i], peopleFavoriteSongs[i][j], inputGenre);
            }
        }
    }


}


int main() {
    int choice = 0; //메뉴 입력받을 변수
    int end = 0; //종료 변수
    int delIndex = -1; //삭제를 위한 변수
    char ch; //버퍼 제거..?
    int modifyIndex = 0; //수정을 위한 변수
    char inputGenre[GENRE_NAME] = "";//장르이름을 입력받을 변수

    // 각 이용자의 정보 입력
    for (int i = 0; i < PERSON_COUNT; i++) {
        printf("사용자 %d, 이름을 입력하세요: ", i + 1);
        scanf_s("%s", peopleNames[i], (int)sizeof(peopleNames[i]));
    }
        printf("음악 플레이리스트 시작!₩n");
```

```c
    // 각 이용자의 정보 입력
    for (int i = 0; i < PERSON_COUNT; i++) {
        printf("사용자 %d, 이름을 입력하세요: ", i + 1);
        scanf_s("%s", peopleNames[i], (int)sizeof(peopleNames[i]));
    }
        printf("음악 플레이리스트 시작!₩n");

    while (1) {
        printf("------------------₩n");
        printf("메뉴를 입력해주세요.₩n");
        printf("1. 재생목록에 노래 추가₩n2. 노래 삭제₩n3. 재생목록₩n4. 종료₩n5. 노래 수정₩n6. 비슷한 노래 둘러보기₩n");
        printf("현재 노래 수 = %d₩n", songCount[0]);  // 여기서는 첫 번째 사람의 노래 수를 표시

        printf("------------------₩n");
        scanf_s("%d", &choice); //실행할 메뉴 입력받음

        switch (choice) { //입력받는 CHOICE에 따라 하는 일을 조건문을 통해 분류
        case 1:
            printf("사용자를 선택하세요 (1~5): ");
            int personIndex;
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) { //메뉴는 1~6사이로 선택, 아니면 오류 표기
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printf("노래의 장르를 입력하세요 (POP, 발라드, 힙합): ");
            scanf_s("%s", inputGenre, (int)sizeof(inputGenre));//장르를 입력받음
            addSong(peopleFavoriteSongs[personIndex - 1][songCount[personIndex - 1] % MAX_SONGS], inputGenre, personIndex - 1);//노래추가 함수 호출
            break;

        case 2:
            printf("사용자를 선택하세요 (1~5): ");
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) {
                printf("잘못된 선택입니다.₩n");
                break;
            }
```

```
            }

            printf("삭제할 노래의 인덱스를 입력하세요 : ");
            scanf_s("%d", &delIndex);
            if (delIndex > songCount[personIndex - 1] || delIndex <= 0) {
                printf("삭제할 노래가 없습니다.₩n");
            }
            else {
                delSong(delIndex, personIndex - 1);
            }
            break;

        case 3:
            printf("사용자를 선택하세요 (1~5): ");
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) {
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printSongList(personIndex - 1);
            break;

        case 4:
            end = 1;
            break;

        case 5:
            printf("사용자를 선택하세요 (1~5): ");
            scanf_s("%d", &personIndex);

            if (personIndex <= 0 || personIndex > PERSON_COUNT) {
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printf("수정할 노래의 번호를 입력해주세요. (1부터 시작): ");
            scanf_s("%d", &modifyIndex);
            ch = getchar();
```

```
                printf("잘못된 선택입니다.₩n");
                break;
            }

            printf("수정할 노래의 번호를 입력해주세요. (1부터 시작): ");
            scanf_s("%d", &modifyIndex);
            ch = getchar();
            printf("새롭게 추가할 노래를 입력해주세요: ");
            scanf_s("%s", peopleFavoriteSongs[personIndex - 1][modifyIndex - 1], (int)sizeof(peopleFavoriteSongs[personIndex - 1][modifyIndex - 1]));
            printf("새로운 노래가 추가되었습니다: %d. %s (장르: %s)₩n", modifyIndex, peopleFavoriteSongs[personIndex - 1][modifyIndex - 1], genre[modifyIndex - 1]);
            break;

        case 6:
            printf("선호하는 장르를 입력하세요 (POP, 발라드, 힙합): ");
            scanf_s("%s", inputGenre, (int)sizeof(inputGenre));
            recommendSong(inputGenre);
            break;

        default:
            printf("잘못된 선택입니다. 다시 선택하세요.₩n");
        }

        if (end == 1) {
            printf("재생목록을 종료합니다.₩n");
            break;
        }
        if (songCount[0] == MAX_SONGS) {
            printf("노래가 %d개로 다 찼습니다.₩n", songCount[0]);
            break;
        }
    }

    return 0;
}
```

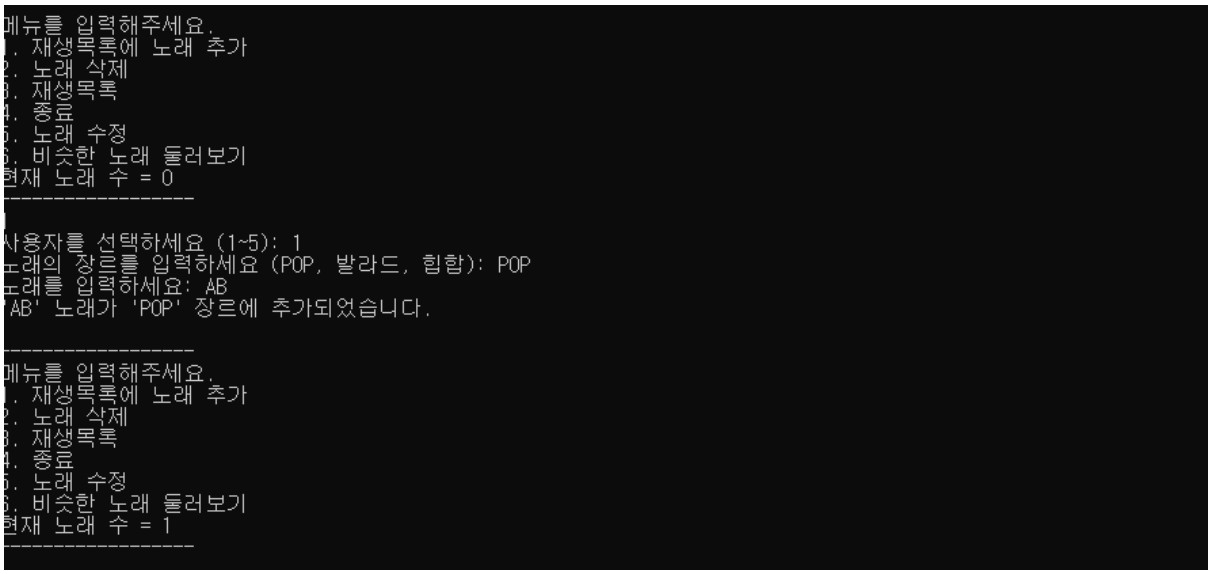
2) 테스트 결과

(1) 재생목록에 노래/장르 추가, 삭제, 수정, 종료

사용자 이름을 5명에게 입력 받는다.

메뉴1을 확인해보기 위해 1입력후 사용자1(가)에게 POP장르의 노래 AB를 추가

노래가 추가되어 현재 노래수가 올라가는 것을 확인.



메뉴의 2번 기능이 잘 작동하는지 확인. 위에서 추가한 노래가 삭제되고, 재생목록을 통해 확인

현재 노래수 0, 재생목록에 곡 없음.

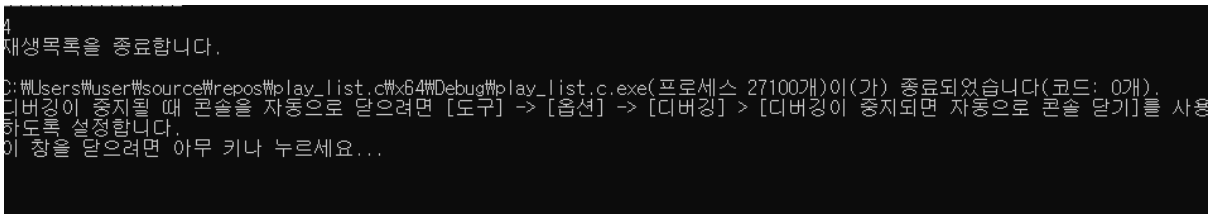
```
2
사용자를 선택하세요 (1~5): 1
삭제할 노래의 인덱스를 입력하세요 : 1
1. AB : 노래를 삭제합니다.
------------------
메뉴를 입력해주세요.
1. 재생목록에 노래 추가
2. 노래 삭제
3. 재생목록
4. 종료
5. 노래 수정
6. 비슷한 노래 둘러보기
현재 노래 수 = 0
------------------
3
사용자를 선택하세요 (1~5): 1
재생목록 for 가:

------------------
메뉴를 입력해주세요.
1. 재생목록에 노래 추가
2. 노래 삭제
3. 재생목록
4. 종료
5. 노래 수정
6. 비슷한 노래 둘러보기
현재 노래 수 = 0
------------------
```

메뉴 3을 입력했을 때 재생목록이 출력되는지 확인, 1에게 POP장르 CD노래를 추가하고

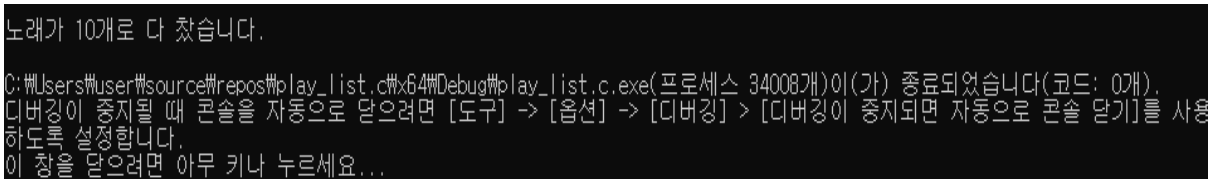
3을 입력하여 확인.

```
사용자를 선택하세요 (1~5): 1
노래의 장르를 입력하세요 (POP, 발라드, 힙합): POP
노래를 입력하세요: CD
'CD' 노래가 'POP' 장르에 추가되었습니다.

------------------
메뉴를 입력해주세요.
1. 재생목록에 노래 추가
2. 노래 삭제
3. 재생목록
4. 종료
5. 노래 수정
6. 비슷한 노래 둘러보기
현재 노래 수 = 1
------------------
3
사용자를 선택하세요 (1~5): 1
재생목록 for 가:
1. CD (장르: POP)

------------------
메뉴를 입력해주세요.
1. 재생목록에 노래 추가
2. 노래 삭제
3. 재생목록
4. 종료
5. 노래 수정
6. 비슷한 노래 둘러보기
현재 노래 수 = 1
```

메뉴의 4번을 입력 받았을 때 종료되는지 확인.

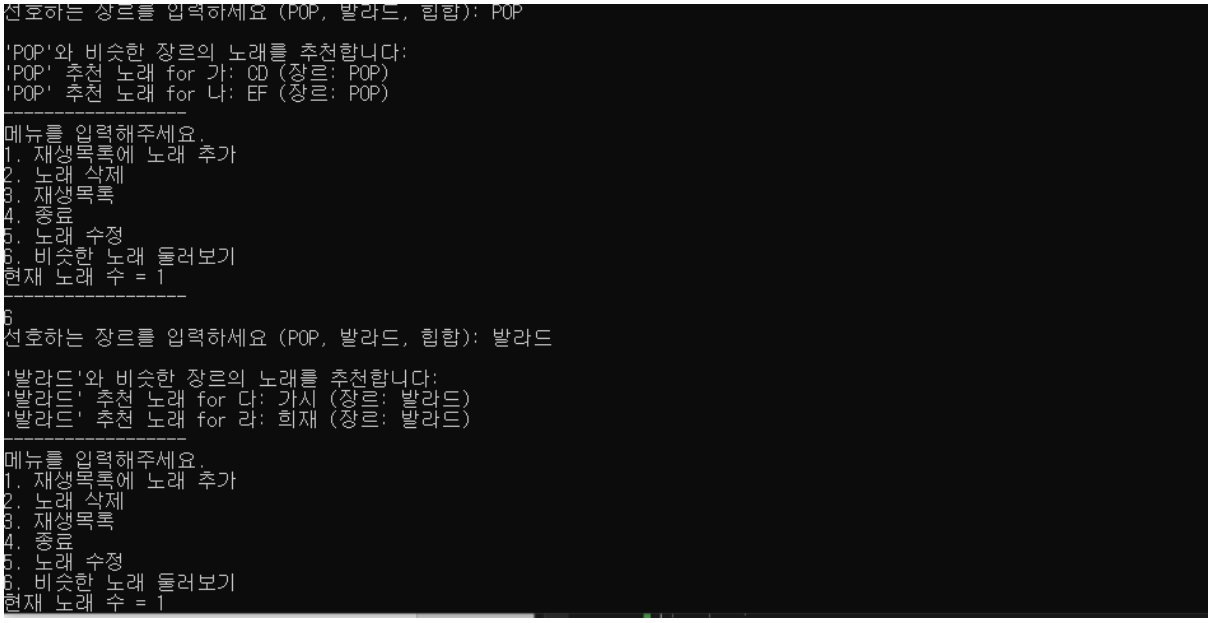


노래가 10개가 다 찼을 때, 노래가 10개로 다 찼다는 문구를 출력



사용자 1의 플레이리스트에 AB라는 곡을 추가 후, 메뉴 5를 통해 CD로 수정되는지 확인.

메뉴 6을 통해 같은 장르를 가진 노래들을 다른 사람의 플레이리스트에서 가져와서 출력되는지 확인. 1에게 POP:CD 2에게 POP:EF 3에게 발라드 가시, 4에게 발라드 희재를 추가하고 출력.



4. 계획 대비 변경 사항

1) 변경 내역 제목

- 이전: 친구들의 플레이리스트 공유

- 이후: 가장 인기있는 장르 출력 및 가장 인기있는 장르별 노래 출력 및 재생횟수 기록, 노래 검색 기능, 이미 재생목록에 있는 노래라면 추가할 때 문구를 표시하고 덮어쓰는 코드 구현. **장르가 가장 많이 겹치는 사람의 플레이리스트를 출력**

- 사유

입출력을 구체화시키기 위해. 친구들 과의 플레이리스트 공유는 메뉴 입력 3에서 이미 가능하기 때문. (노래를 듣는데 맨날 듣는 노래가 중복해서 재생목록에 끼어 있으면 계속 그 노래만 나와서 불편함.) >> 따라서 중복되지 않도록 문구를 표시하고 겹치는 노래를 한 개만 재생목록에 표시.

5. 프로젝트 일정

<table>
<tr><th colspan="2">업무</th><th>11/3</th><th colspan="2">11/4~11/18</th><th>11/19~11/26……</th><th>11/26~12/10</th><th>12/11~12/26</th></tr>
<tr><td colspan="2">제안서 작성</td><td>완료</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>기능1</td><td>재생목록 입출력, 삭제, 수정</td><td></td><td colspan="2">완료</td><td></td><td></td><td></td></tr>
<tr><td>기능2</td><td>사용자 이름, 장르 입력 및 같은 장르 노래 출력</td><td></td><td></td><td colspan="2">완료</td><td></td><td></td></tr>
<tr><td>기능3</td><td>장르와 노래 순위 매기기
이용시간과 나이 입력 받기</td><td></td><td></td><td></td><td colspan="2">진행중

-→</td><td></td></tr>
<tr><td>기능4</td><td>중복제거 등</td><td></td><td></td><td></td><td></td><td colspan="2">진행 예정
----------</td></tr>
</table>