

프로그래밍 프로젝트

<영화 데이터 검색 시스템>



<교수님의 총애팀>

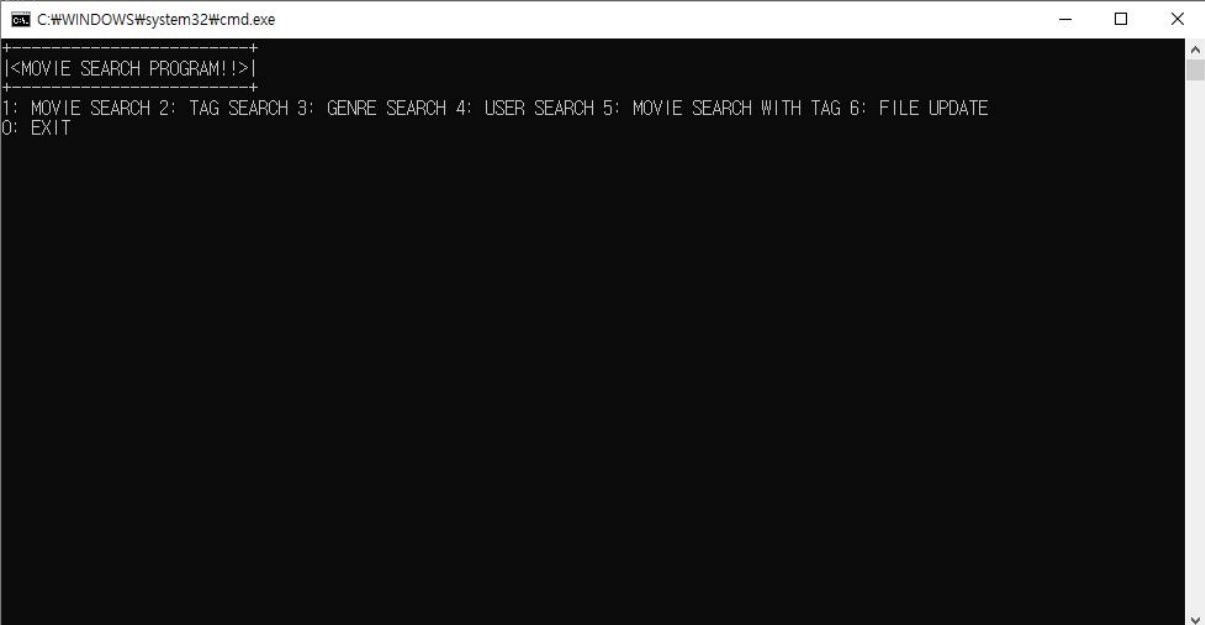
20182327 서준원

20182036 이진형

<목차>

1. -----프로그램 소개
2. -----기본 함수
3. -----메인 기능
4. -----부가 기능
5. -----의의
6. -----개선 방안
7. -----팀원 분담

1. 프로그램 소개



```
C:\WINDOWS\system32\cmd.exe
+-----+
|<MOVIE SEARCH PROGRAM!!>|
+-----+
1: MOVIE SEARCH 2: TAG SEARCH 3: GENRE SEARCH 4: USER SEARCH 5: MOVIE SEARCH WITH TAG 6: FILE UPDATE
0: EXIT
```

이 프로그램에서 다룰 데이터는 각각 영화 데이터와 태그 데이터이다. 영화 데이터의 경우에는 영화의 번호, 제목 (연도), 장르들로 구성되어 있으며, 태그 데이터의 경우에는 유저의 ID, 영화의 번호, 태그 내용 그리고 타임 스탬프로 구성되어 있다.

첫번째로는 영화 데이터에만 접근해서, 영화의 제목이나 장르 등이 프로그램을 구동함에 있어서 개별적으로 사용되기 때문에, 데이터를 구조체에 따로 저장할 것이다.

두번째로는 태그 데이터에만 접근하여, 태그 데이터를 구조체에 유저 ID, 영화 번호, 태그 내용과 타임 스탬프를 영화 데이터와 마찬가지로 구조체에 따로 저장할 것이다.

프로그램을 구현함에 있어서 메인 기능이 영화를 입력받아 태그를 출력하는 것과, 태그를 입력받아 영화를 출력하는 것이다. 이때, 태그 데이터와 영화 데이터가 복합적으로 사용된다. 예를 들면, 영화 데이터에서의 영화 번호와 태그 데이터에서의 영화 번호가 같을때 태그를 출력하거나, 영화를 출력할 수 있다.

2.기본 함수

① movieget()

이 함수는 movies.dat 파일의 데이터를 우리가 다루기 편하게끔 파일을 불러와 그 데이터를 다루는 함수이다.

movies.dat 파일에서의 데이터는 다음과 같이 저장되어있다.

영화 번호 :: 영화명(출시년도) :: 장르 1 | 장르 2 | 장르 3 ...

위와 같은 각각의 영화데이터는 줄바꿈을 기준으로 나뉘어져있다.

그렇기에 movieget()함수는 이 데이터를 fgets를 통해 buff[HOWMANY][150]에 저장한다.

```
buff[1] = 1 :: movie 1 :: Action | Animation
buff[2] = 2 :: movie 2 :: Horror
      :
```

이후 buff[i][k] == : && buff[i][k+1] == : 일때를 기준으로 tokenizing해준다.

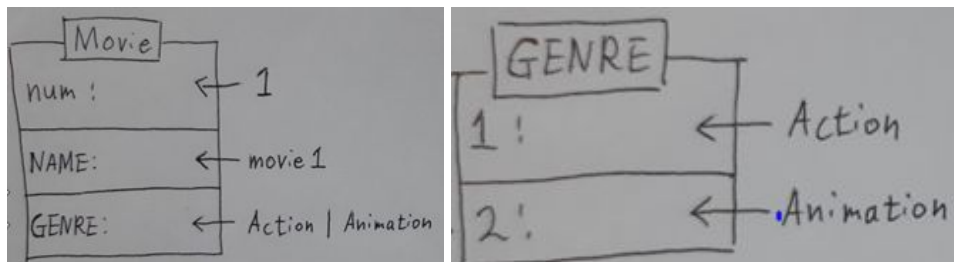
(처음에는 strtok로 tokenizing했으나 영화명에 “:”가 들어간 영화의 경우 오류가 나기에 위와같은 방식으로 변경하여 tokenizing을 진행하였다)

이를 통해 한 영화의 데이터를 영화번호, 영화명 그리고 복수의 장르들로 나눠줄 수 있다.

영화의 장르의 경우 strtok함수를 이용하여 “|”을 기준으로 한번 더 tokenizing해준다.

이로써 복수의 장르들 또한 각각 tokenizing되어 개별적으로 다룰 수 있게된다.

그 이후 이를 구조체 movie의 각각의 부분에 넣어준다.



장르를 tokenizing하는 경우 한번 tokenizing 될 때마다 G_NUM=0을 1씩 증가시켜준다.

장르의 tokenizing이 다 끝난 후 G_NUM을 구조체 movie의 GENRE_NUM에 넣어준다.

이를 통해 한 영화에 몇 개의 장르가 포함되어 있는지 체크할 수 있게끔 한다.

이 함수를 응용하는 다른 함수들에 대한 설명은 이 뒤에 나오는 메인기능과 부가기능에 나와있다.

② tagget()

이 함수는 tags.dat 파일의 데이터를 우리가 다루기 편하게끔 파일을 불러와 그 데이터를 다루는 함수이다.

tags.dat 파일에서의 데이터는 다음과 같이 저장되어있다.

사용자 번호 :: 영화 번호 :: 태그 :: 타임스태프

위와 같은 각각의 태그데이터는 줄바꿈을 기준으로 나뉘어져있다.

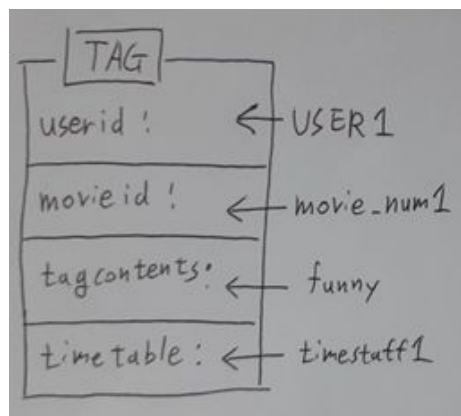
그렇기에 tagget()함수는 movieget()함수와 마찬가지로 이 데이터를 fgets를 통해 buff[HOWMANYTAG][200]에 저장한다.

```
buff[1] = USER 1 !! movie_num1 !! funny !! timestaff1
buff[2] = USER 2 !! movie_num2 !! excellent !! timestaff2
```

태그데이터에는 :만 존재하는 구간이 따로 없기에 바로 strtok함수를 이용하여 “:”을 기준으로 tokenizing을 해준다.

이를통해 한 태그의 데이터를 사용자 번호, 영화 번호, 태그, 타임스태프로 나뉘줄 수 있게된다.

이후 tokenizing된 데이터들을 구조체 tag의 각각의 부분에 넣어준다.



③ SelectMenu()

사용자로부터 번호를 입력받아 실행할 함수를 결정하는 함수이다.

사용자로부터 1을 입력받으면 영화명 검색을, 2를 입력받으면 태그 검색을, 3을 입력받으면 장르 검색을, 4를 입력받으면 유저 검색을, 5를 입력받으면 영화를 검색하는데 태그와 함께 검색을 그리고 6을 입력받으면 movies.dat에 데이터를 추가하는 함수를 실행하도록 하였다. (6은 현재 구현중에 있다.)

3. 메인 기능

① 영화 데이터를 입력 받아 그 영화의 태그 데이터 출력

```
+-----+
|<MOVIE SEARCH PROGRAM!!>|
+-----+
1: MOVIE SEARCH 2: TAG SEARCH 3: GENRE SEARCH 4: USER SEARCH 5: MOVIE SEARCH WITH TAG 6: FILE UPDATE
0: EXIT

5
Write Movie You want to Search : Iron Giant, The
Movie Number: 2761
Movie Name: Iron Giant, The (1999)
based on a book
directorial debut
space
Ray Bradbury
sufficiently explodey to be good
dinosaurs
time travel
robots
dinosaurs
time travel
less than 300 ratings
MILITARY LIFE
dinosaurs
time travel
scope
animation
erlend's DVDs
robot
animation
robots
robot
Tumey's DVDs
(s)vcd
buy
robots
classical animation
iraq
cold war
robot
robots
totally overlooked
kids
robot
giant robots
Number of Iron Giant, The (1999)'s Tag : 34

계속하려면 아무 키나 누르십시오 . . .
```

5번을 입력후 영화 제목을 그대로 입력하면, 그 영화의 태그 데이터를 출력한다. 이때 같은 태그를 다른 사람이 쓴 경우가 있어서, 같은 내용의 태그가 출력될 수도 있다.

여기서 부가 기능으로 영화 제목을 전부 다 입력 받지 않아도 예를 들어 Iron 만 검색해도 Iron Man, Iron Giant 등의 영화 태그가 아무 키를 입력받으면 하나씩 출력되기 때문에, 영화의 내용이 얼핏 기억나는데 제목이 기억이 잘 나질 않는 경우 태그를 찾아보면서 영화를 찾을 수 있다.

```
adapted from
android(s)/cyborg(s)
Bechdel Test
murder
setting
topic
weapons industry
dvd
florence
redbox
Marvel
comic book
Marvel
movie to see
superhero
adventure
Marvel
Number of Iron Man (2008)'s Tag : 85
```

또한 태그의 개수까지 마지막에 정리되어서 나오기 때문에, 태그가 얼마나 많이 달렸는지 즉, 사람들이 얼마나 이 영화에 관심이 많은지를 태그의 개수로도 유추할 수 있기 때문에, 영화 추천에 있어서 관찰은 척도로 사용될 수 있다.

이 함수의 구현 방법은 다음과 같다.

먼저 사용자로부터 영화명을 입력받는다. 이는 전체 영화명이 아닌 일부 영화명만 입력받아도 가능하다.

movieget()함수를 통해 movies.dat에서 영화의 정보를 받아오고 strcmp를 통해 구조체 movie의 NAME과 사용자가 입력한 영화명을 비교한다.

만약 두 값이 같다고 컴퓨터가 판단했을 경우 영화번호와 영화명을 출력하고 영화번호는 따로 저장한다.

그 다음 tagget()함수를 통해 tags.dat에서 태그의 정보를 받아오고 구조체 tag의 movieid와 방금

따로 저장했던 영화번호와 비교해준다.

만약 두 값이 같다고 컴퓨터가 판단했을 경우 구조체 tag의 tagcontents를 출력하고 이와 같은 일이 발생할 때 마다 tag_count = 0의 값을 1씩 증가시켜준다.

일련의 과정이 다 끝난 후 tag_count의 값을 통해 그 영화에 달린 태그의 개수 또한 출력해준다.

② 태그 데이터를 입력받아 영화 데이터를 출력

```
+-----+
|<MOVIE SEARCH PROGRAM!!>|
+-----+
1: MOVIE SEARCH 2: TAG SEARCH 3: GENRE SEARCH 4: USER SEARCH 5: MOVIE SEARCH WITH TAG 6: FILE UPDATE
0: EXIT

2
Write name of the tag you want to search : Marvel
Movie Tag : Marvel
User ID: 29592
Movie Number: 2167
Movie Name: Blade (1998)
Movie Genre: Action
Movie Genre: Horror
Movie Genre: Thriller

Movie Tag : Marvel
User ID: 36784
Movie Number: 2167
Movie Name: Blade (1998)
Movie Genre: Action
Movie Genre: Horror
Movie Genre: Thriller

Movie Tag : Marvel
User ID: 29592
Movie Number: 5254
Movie Name: Blade II (2002)
Movie Genre: Action
Movie Genre: Horror
Movie Genre: Thriller

Movie Tag : Marvel
User ID: 36784
Movie Number: 5254
Movie Name: Blade II (2002)
Movie Genre: Action
Movie Genre: Horror
Movie Genre: Thriller
```

2번을 입력후 태그를 입력받으면, 그 태그가 있는 영화를 출력해준다. 이때도 역시 같은 태그 내용을 다른 사람이 쓴 경우가 있어서, 영화가 중복 검색 될 수 있다.

```
Write name of the tag you want to search : Excellent
Movie Tag : Excellent just excellent
User ID: 8423
Movie Number: 457
Movie Name: Fugitive, The (1993)
Movie Genre: Thriller
```

이때 부가기능으로 Excellent 만 검색해도 Excellent just excellent 같이 비슷한 내용까지도 출력해준다.

이 함수의 구현 방법은 다음과 같다.

먼저 사용자에게 찾을 태그를 입력받는다. 이때도 마찬가지로 태그 전체가 아닌 일부만 입력해줘도 상관없다.

tagget()함수를 통해 tags.dat에서 태그의 정보를 받아온다.

이후 strcmp를 통해 구조체 tag의 tagcontents와 사용자에게 입력받은 태그를 비교해준다.

만약 컴퓨터가 이 둘을 같다고 판단했을 경우 구조체 tag의 영화번호, 태그 그리고 유저아이디를 저장한다.

다음에 movieget()함수를 통해 movies.dat에서 영화의 정보를 받아온 후 strcmp를 통해 구조체 movie의 num과 위에서 저장한 영화번호를 비교하여 그 영화의 정보 즉, 영화명, 영화의 장르(복수)를 저장한다.

이후 저장했던 데이터들(태그, 유저아이디, 영화번호, 영화명, 영화장르)을 모두 출력해준다.

3. 부가 기능

① 영화 검색

```

+-----+
|<MOVIE SEARCH PROGRAM!!>|
+-----+
1: MOVIE SEARCH 2: TAG SEARCH 3: GENRE SEARCH 4: USER SEARCH 5: MOVIE SEARCH WITH TAG 6: FILE UPDATE
0: EXIT

1
Write name of the movie you want to search : Iron
Movie Number: 2761
Movie Name: Iron Giant, The (1999)
Movie Genre: Adventure
Movie Genre: Animation
Movie Genre: Children
Movie Genre: Drama
Movie Genre: Sci-Fi

Movie Number: 2815
Movie Name: Iron Eagle (1986)
Movie Genre: Action
Movie Genre: War

Movie Number: 2816
Movie Name: Iron Eagle II (1988)
Movie Genre: Action
Movie Genre: War

Movie Number: 2818
Movie Name: Iron Eagle IV (1995)
Movie Genre: Action
Movie Genre: War

Movie Number: 4122
Movie Name: Ironweed (1987)
Movie Genre: Drama

```

1번을 입력받은 후 영화의 제목을 검색하면, 그 영화의 제목과 장르등을 출력해준다. 실생활에서 사용될 수 있는 점은, 영화의 제목이 앞부분만 조금밖에 기억나지 않을때 앞부분만 검색해서 장르등을 보고 영화를 찾을 수 있고, 이 기능과 더불어 5번 메인 기능을 같이 사용한다면, 태그를 이용해 어느정도의 내용까지도 검색할 수 있기 때문에 영화를 찾는데 매우 도움이 될 것이다.

이 함수의 구현방법은 다음과 같다.

먼저 검색하고자 하는 영화의 명칭을 입력한다. 이는 영화명 전체가 아니고 일부 영화명만 입력해주어도 괜찮다.

이후 movieget()함수를 통해 movies.dat에서 영화의 정보를 받아오고 strcmp를 통해 구조체 movie의 NAME과 앞서 사용자로부터 받았던 영화의 명칭을 비교해준다.

이 둘이 서로 같다고 컴퓨터가 판단했을 경우 영화 데이터들(영화번호, 영화명, 영화장르)을 모두 출력해준다.

② 장르 검색

```
3
Number of Genre You want to search : 3
Choose Genre you want to Search
1. Action
2. Adventure
3. Animation
4. Children's
5. Comedy
6. Crime
7. Documentary
8. Drama
9. Fantasy
10. Film-Noir
11. Horror
12. Musical
13. Mystery
14. Romance
15. Sci-Fi
16. Thriller
17. War
18. Western
Num : 1
Num : 3
Num : 5
Movie Number: 5460
Movie Name: Powerpuff Girls, The (2002)
Movie Genre: Action
Movie Genre: Animation
Movie Genre: Children
Movie Genre: Comedy

Movie Number: 8917
Movie Name: Team America: World Police (2004)
Movie Genre: Action
Movie Genre: Adventure
Movie Genre: Animation
Movie Genre: Comedy
Movie Genre: Drama

Movie Number: 8961
Movie Name: Incredibles, The (2004)
Movie Genre: Action
Movie Genre: Adventure
Movie Genre: Animation
Movie Genre: Children
Movie Genre: Comedy
```

3을 입력받은 후 자신이 장르를 한번에 몇개 검색할지 입력한다. 그 후에 출력되는 장르의 번호를 따라 입력하는데, 만약 액션이면서 애니메이션이고 코믹한 장르를 검색하고 싶다면 1,3,5을 입력한다. 영화를 추천받는 것에 있어서 자신이 보고 싶은 장르를 한번에 여러개를 검색할 수 있기때문에, 상당히 유용한 기능이 될 것이다.

이 함수의 구현 방법은 다음과 같다

검색할 장르 개수를 입력받고 그 개수만큼 검색할 장르를 입력받는다. 장르의 개수는 S_NUM에 저장하고, for 문을 S_NUM의 크기만큼 반복한다.

for문에서는 새로 선언한 G_NAME[MAX_GENRE][MAX_GENRE_LEN]에 넣을 데이터를 switchcase문을 통해 넣어준다. 예를 들자면, S_NUM이 2이면, G_NAME[0], G_NAME[1]에 switch case를 통해 Animation이나 Comedy같은 것이 들어간다. 그리고 나서 movieget함수를 통해 movies.dat의 영화정보를 가져오고 strncmp를 통해 영화가 가지고 있는 장르를 위에서 검색할 장르와 비교한다. 만약 영화가 각각의 장르를 가지고 있을 시에 장르 하나당 c_num=0을 1 증가시킨다. 그리고 나서 영화가 사용자가 검색하고자 하는 모든 장르를 가지고 있을때 즉, c_num과 S_NUM이 일치할때 영화번호, 영화명, 영화장르(복수)를 출력해준다.

③ 유저 검색

```
+-----+
|<MOVIE SEARCH PROGRAM!!>|
+-----+
1: MOVIE SEARCH 2: TAG SEARCH 3: GENRE SEARCH 4: USER SEARCH 5: MOVIE SEARCH WITH TAG 6: FILE UPDATE
0: EXIT

4
Write User Id you want to search : 64633
Tags of USER ID: 64633
Movie Tag : Toronto Film Festival Winner
Movie Number: 82
Movie Name: Antonia's Line (Antonia) (1995)
Movie Genre: Comedy
Movie Genre: Drama

Movie Tag : Toronto Film Festival Winner 1995
Movie Number: 82
Movie Name: Antonia's Line (Antonia) (1995)
Movie Genre: Comedy
Movie Genre: Drama

Movie Tag : remade into tortilla soup
Movie Number: 232
Movie Name: Eat Drink Man Woman (Yin shi nan nu) (1994)
Movie Genre: Comedy
Movie Genre: Drama
Movie Genre: Romance

Movie Tag : abuse
Movie Number: 290
Movie Name: Once Were Warriors (1994)
Movie Genre: Crime
Movie Genre: Drama

Movie Tag : alcoholism
Movie Number: 290
Movie Name: Once Were Warriors (1994)
Movie Genre: Crime
Movie Genre: Drama

Movie Tag : auckland
Movie Number: 290
Movie Name: Once Were Warriors (1994)
Movie Genre: Crime
Movie Genre: Drama
```

4를 입력받은 후, 자신이 검색하고 싶은 유저의 ID를 입력하면, 그 유저가 쓴 태그들을 볼 수 있다. 메인 기능인 태그 서치를 하면 유저 ID가 나오므로, 그 유저가 가지고 있는 생각이 나랑 비슷하다고 생각되면, 이 기능을 사용하여 그 유저의 태그들을 볼 수 있고, 이 나와 취향이 비슷한 유저가 추천한 영화를 찾아서 영화를 좀 더 실패하지 않고 볼 수 있다.

이 함수의 구현 방법은 다음과 같다

유저아이디를 입력받고 tagget()함수를 통해 tags.dat에 있는 태그데이터들을 불러온다. strcmp를 통해 structure tag에 있는 userid와 입력한 유저 아이디를 비교한 후에 만약 둘이 같을 시 그 태그에 같이 있는 영화번호와 태그 내용을 저장한다. 그 이후에 movieget()함수를 통해 movie.dat에 있는 무비 데이터를 불러오고, 무비데이터의 영화번호가 저장한 영화번호가 같다면 그 영화의 번호와 제목 그리고 장르등을 태그 내용과 함께 출력해준다.

5. 의의

① fgets

처음에 프로젝트를 구상하였을때는 버퍼를 크게 만들어서 거기에 글자를 전부 넣고 영화 한개당 :: 를 짝수개씩 짤라서 한줄씩 넣고 그것을 또 장르에 따라 tokenize 하려고 하였다. 그리고 글자의 개수를 세서 파일 포인터를 영화 데이터의 형식 혹은 태그 데이터의 형식에 딱 맞게끔 움직이려 하였다.

하지만, 영화 데이터와 태그 데이터를 드래그해서 다른 곳에 붙여넣기를 우연치않게 하게 되었는데, 이것이 모두 영화 데이터의 형식만큼 줄바꿈이 되어있어서, fgets가 한줄씩 받아오는 것이라 이것을 사용하게 되었다.

나중에 다시보니, Term Project in Programming에 줄바꿈이 되어있는 것을 확인 할 수 있었다.(?)

② timetable

추가기능으로 timetable를 태그를 출력해줄때, 한국 시간으로 변환하여 언제 이것이 쓰였는지를 같이 출력해주려 했으나, 발표후에 질문을 받고나서 과연 이 기능이 어떻게 더 쓰일 수 있을까를 생각해보았는데, 영화 검색이라는 목적에 있어서 크게 필요할 것 같지 않아 삭제하게 된 기능이다.

③ 모듈화

프로그램을 설계할 때, 최대한 모든 함수들을 모듈화하기위해 노력하였다. 모듈화를 함으로써, 우리 팀이 짠 코드를 남들이 봤을때 좀 더 이해하기 쉬울 것이라 생각한다. 또한, 나중에 코딩을 할때 이와 비슷한 함수가 필요하겠다 싶으면 여기에 있는 것을 붙여 쓸 수 있기 때문에 효율적인 코딩이 좀 더 가능할 것이라 생각한다.

6. 개선 방안

① 사용자의 메모리 크기에 따른 설정

현재의 구현 방안은 영화나 태그를 최대 몇개까지 메모리에 잡을 것인가를 전처리로 처리해놓았다. 좀 더 좋은 프로그램이 되기 위한 하나의 방법은 malloc을 사용해서 사용자가 프로그램을 실행할때, 사용자의 컴퓨터의 성능에 따라 그 숫자를 유동적으로 설정할 수 있게 만드는 것이다. 현재의 영화 데이터나 태그 데이터가 작은편에 속하지만 데이터가 더욱 커졌을때 사용자가 그 값을 조절하여 최적의 성능을 구현할 수 있을 것이라 생각한다.

② 데이터 추가 기능 구현

데이터를 추가하는 것을 구현하려 했는데, 구현 방안을 파일을 a+ 모드로 열고 영화의 이름을 gets로 입력받고 파일을 읽어서 마지막의 숫자를 저장하고 그 숫자보다 1만큼 큰 숫자를 영화의 번호로 사용하고 장르와 년도를 입력받아 fprintf 함수를 이용해서 맨마지막에 추가하려 했다. 원래 파일의 마지막 영화의 번호보다 큰 숫자가 NULL 값이라고 생각하고 코딩하였는데, 이것이 NULL로 인식이 되지 않아서, 구현을 하지 못하였다. 영화의 번호를 가져올 수 있는 방안을 새로 고안한다면, 좀 더 좋은 프로그램이 될 것이라 생각한다.

7. 팀원 분담

서준원- movieget() tagget() 과 같은 기본함수를 구현, 유저검색 함수를 구현하였다.

이진형 - 부가 기능의 일부와 메인 기능을 구현하였다.

그리고 서로 함께 보고서를 작성하였으며, 서로가 구현한 함수가 잘 되지않았을때 서로 자문을 구했다.