Open SW 개론 중간 과제

학과 : 인공지능 공학과
학번 : 12215529
이름 : 차승우

1. 코드 작성한 방식

- 저의 경우 코드를 작성할 때 한 번에 같은 파일에 작성해버리면 오류가 발생할 경우 고치기가 힘들다고 생각했으며 수업에서도 코드를 조각처럼 한 조각을 만들면 붙이고 오류를 정정하는 방식을 썼기에 저 또한 test라는 이름의 쉘 스크립트를 8개 정도 만들어서 시도해보고 성공할 경우 원본에 합치는 방식을 사용하였습니다.

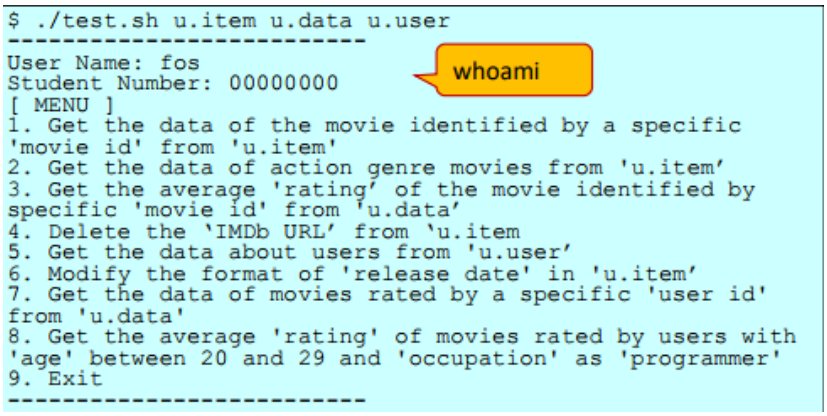
2. Code 1(Menu)

- u.item, u.data, u.user의 경우는 인자로 받아올 수 있도록 각 변수에 $1, $2, $3으로 받아와 각각의 변수에 각 파일에 해당하는 경로를 저장하였습니다.
- 이후 case의 각 경우를 받을 num 변수를 초기화를 진행하였으면 echo 명령어를 통해 각 번호에 해당하는 메뉴에 대한 물음을 작성하였습니다.
- User name에서는 whoami라는 명령어를 통해 사용자의 ID를 불러왔습니다. Command substitution을 활용하여 문자열에 명령을 넣었습니다.

제가 실행한 결과

```
unbuntu@unbuntu-VirtualBox:~$ bash test.sh u.item u.data u.user
-------------------------
User Name: unbuntu
Student Number: 12215529
[ MENU ]
1. Get the data of the movie identified by a specific 'movie id' from 'u.item'
2. Get the data of action genre movies from 'u.item'
3. Get the average 'rating' of the movie identified by specific 'movie id' from 'u.data'
4. Delete the 'IMDb URL' from 'u.item'
5. Get the data about users from 'u.user'
6. Modify the format of 'release data' in 'u.item'
7. Get the data of movies rated by a specific 'user id' from 'u.data'
8. Get the average 'rating' of movies rated by users with 'age' between 20 and 29 and 'occupation' as 'programmer'
9. Exit
-------------------------
Enter your choice [ 1-9 ]
```

Pdf가 요구한 결과



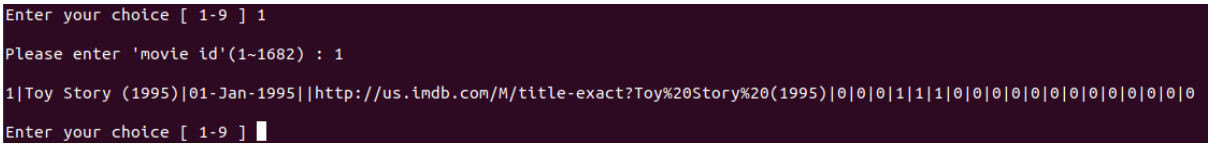
3. Code 2(While and case)

- While : 을 통해 break나 exit가 없을 경우 무한 루프를 사용하여 루프가 지속되도록 하였습니다.
- case 문을 통해 사용자가 입력한 숫자를 num이란 변수에 저장하고 num에 입력한 숫자에 따라 1 ~ 9 번의 메뉴를 실행할 수 있도록 진행하였습니다.

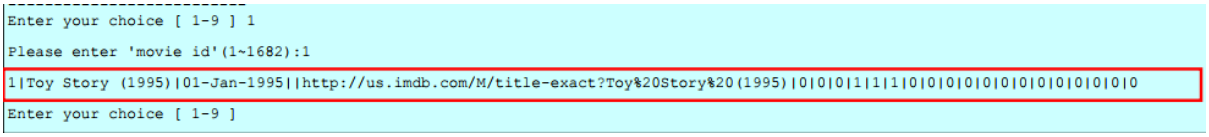
4. Code 2(Menu 1)

- 첫 번째 메뉴라서 변수를 first의 약자인 fir로 설정하였습니다.
- 'read –p "문자열" 변수'를 통하여 문자열을 출력한 후 사용자가 입력한 문자를 fir의 변수에 저장합니다.
- cat 명령어를 통해 u.item을 일고 '|'를 통해 명령어를 이어나가고 sed -n를 통해 사용자가 입력한 숫자에 해당하는 줄을 읽습니다.

제가 실행한 결과



Pdf가 요구한 결과



5. Code 3(Menu 2)

- ans는 사용자의 답변을 저장하기 위한 변수로 초기화를 진행하였습니다.
- 'read –p "문자열" 변수'를 통하여 문자열을 출력한 후 사용자가 입력한 문자를 ans의 변수에 저장합니다.
- if문을 통해 사용자가 y를 입력할 경우 ans에 저장 후 다음의 과정을 수행합니다.
- 파일의 처음 10줄을 읽어와서 각 사용자의 정보를 출력합니다.
- u.item에서 '|'를 기준으로 u.item의 7번째 항, action 장르가 맞을 경우 해당 행의 1항과 2항을 출력합니다. count를 통해 10개 줄까지만 받습니다.
- u.item을 |를 기준으로 나누고 그 행의 첫 번째, 두 번째 요소를 10개를 출력합니다.
- if문의 작성을 종료하기 위해 fi를 써줍니다.
- 1개의 case를 종류하기 위해 ;;를 작성해 1개의 case가 끝났음을 알립니다.

제가 실행한 결과

```
Enter your choice [ 1-9 ] 2

Do you want to get the data of 'action' genre movies from 'u.item'?(y/n):y

2 GoldenEye (1995)
4 Get Shorty (1995)
17 From Dusk Till Dawn (1996)
21 Muppet Treasure Island (1996)
22 Braveheart (1995)
24 Rumble in the Bronx (1995)
27 Bad Boys (1995)
28 Apollo 13 (1995)
29 Batman Forever (1995)
33 Desperado (1995)

Enter your choice [ 1-9 ]
```

Pdf가 요구한 결과

```
Enter your choice [ 1-9 ] 2

Do you want to get the data of 'action' genre movies
from 'u.item'?(y/n):y

2 GoldenEye (1995)
4 Get Shorty (1995)
17 From Dusk Till Dawn (1996)
21 Muppet Treasure Island (1996)
22 Braveheart (1995)
24 Rumble in the Bronx (1995)
27 Bad Boys (1995)
28 Apollo 13 (1995)
29 Batman Forever (1995)
33 Desperado (1995)

Enter your choice [ 1-9 ]
```

6. Code 4(Menu 3)

- 소수 다섯짜리까지 반올림하기 위해 어떻게 해야 하는지에 관해 생각을 해 보았습니다. 좀 돌아가는 형식이지만 printf의 %를 통한 지정을 몰랐던 때이기에 소수 6번째 이상까지 존재하는 소수의 경우에는 소수 여섯 번째 자리 이후 버림을 한 수에 를 num_round1에 저장해 주고 똑같은 수를 소수 5번째 자리에서 버림을 한 뒤에 똑같이 num_round2에 저장을 합니다. 이후 각각의 변수에 1,000,000을 곱한 뒤 빼면 1자리의 자연수만 남게 됩니다. 그 수가 4보다 클 경우 num_round1 * 1,000,000에 10을 더해 본래는 소수 5번째 자리였을 숫자를 1 더해줘 반올림하게 만들어 주었고 반올림의 경우가 아닌 경우에는 소수 5번째에서 버림한 수 인 num_round2를 나타내줍니다. 그러나 현재의 num_round1, num_round2의 경우네는 1,000,000을 곱한 자연수의 상태이기에 다시 1,000,000을 나눠 소수의 형태로 바꿔서 출력을 하게 됩니다.

Ex)

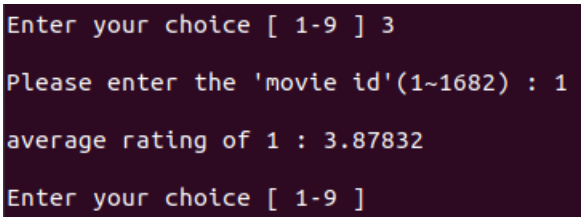
해당 하는 소수 3.878318...
num_round1 = 3.878318
num_round2 = 3.87831

num_round3 = num_round1 * 1,000,000 - num_round2 * 1,000,000 = 3,878,318 - 3.878,310 = 8

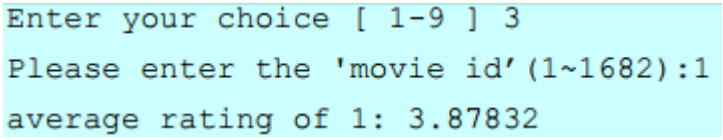
이렇듯 소수 6번째 자리를 자연수로 잘라내어 그 수가 4이상일 경우네는 num_round1*1,000,000에 10을 더하여 소수 다섯 번째 자리가 올림 한 것으로 나타내고 다시 1,000,000을 나눠줘 원래의 소수로 돌아가게 합니다.

- movie id를 입력받아 thr에 저장합니다.
- u.data 파일로부터 데이터를 읽어와 각 4가지 요소를 num1, num2, num3, num4에 저장하고 while 문에 적힌 명령을 수행합니다.
- 'movie id'와 일치하는 경우, 해당 영화의 'rating' 을 total_sum에 더해 해당 movie에 대한 평점을 합산합니다.
- 평점을 위한 평균을 위해 count는 rating을 합산할 때마다 증가시킵니다.
- 이후 위에서의 total_sum / count를 통해 평균을 계산하고 소수점 반올림을 마친 뒤에 average에 저장하고 출력을 진행합니다.

제가 실행한 결과



Pdf가 요구한 결과



7. Code 5(Menu 4)

- ans는 사용자의 답변을 저장하기 위한 변수로 초기화를 진행하였습니다.
- 'read –p "문자열" 변수' 를 통하여 문자열을 출력한 후 사용자가 입력한 문자를 ans의 변수에 저장합니다.
- if문을 통해 사용자가 y를 입력할 경우 ans에 저장 후 다음의 과정을 수행합니다.
- 정규 표현식은 'http'로 시작하고 '|'로 끝나는 경우 'IMDb URL'로 판단하고, u.item에서 올림차순으로 10줄을 뽑아 명령을 실행해 해당 내용을 지운 걸로 대체한 문장을 출력합니다.
- if문의 작성을 종료하기 위해 fi를 써줍니다.
- 1개의 case를 종류하기 위해 ;;를 작성해 1개의 case가 끝났음을 알립니다.

제가 실행한 결과

```
Enter your choice [ 1-9 ] 4

Do you want to delete the 'IMDb URL' from 'u.item'?(y/n):y

1|Toy Story (1995)|01-Jan-1995|||0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995|||0|1|1|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995|||0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995|||0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995|||0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-1995|||0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995|||0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
8|Babe (1995)|01-Jan-1995|||0|0|0|0|1|1|0|0|1|0|0|0|0|0|0|0|0|0
9|Dead Man Walking (1995)|01-Jan-1995|||0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
10|Richard III (1995)|22-Jan-1996|||0|0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|1|0

Enter your choice [ 1-9 ]
```

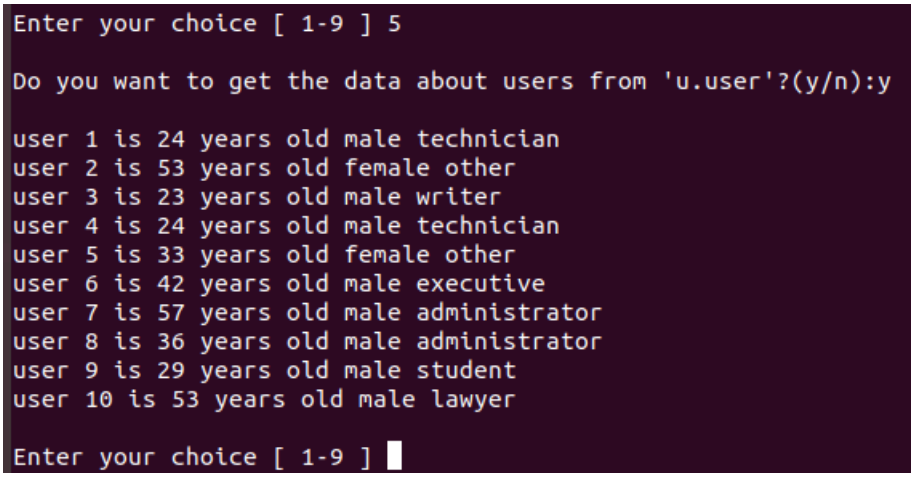
Pdf가 요구한 결과

```
Enter your choice [ 1-9 ] 4

Do you want to delete the 'IMDb URL' from 'u.item'?(y/n):y

1|Toy Story (1995)|01-Jan-1995|||0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0
2|GoldenEye (1995)|01-Jan-1995|||0|1|1|0|0|0|0|0|0|0|0|0|0|0|1|0|0
3|Four Rooms (1995)|01-Jan-1995|||0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|0|0
4|Get Shorty (1995)|01-Jan-1995|||0|1|0|0|0|1|0|0|1|0|0|0|0|0|0|0|0|0
5|Copycat (1995)|01-Jan-1995|||0|0|0|0|0|0|1|0|1|0|0|0|0|0|0|1|0|0
6|Shanghai Triad (Yao a yao yao dao waipo qiao) (1995)|01-Jan-
1995|||0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0
7|Twelve Monkeys (1995)|01-Jan-1995|||0|0|0|0|0|0|0|1|0|0|0|0|0|0|1|0|0|0
8|Babe (1995)|01-Jan-1995|||0|0|0|0|1|1|0|0|1|0|0|0|0|0|0|0|0|0
9|Dead Man Walking (1995)|01-Jan-1995|||0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|0|0|0
10|Richard III (1995)|22-Jan-1996|||0|0|0|0|0|0|0|1|0|0|0|0|0|0|0|1|0
```

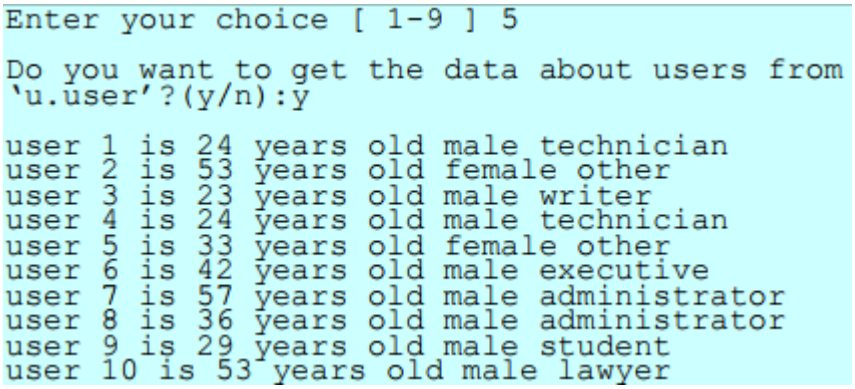
8. Code 6(Menu 5)

- ans는 사용자의 답변을 저장하기 위한 변수로 초기화를 진행하였습니다.
- 'read –p "문자열" 변수' 를 통하여 문자열을 출력한 후 사용자가 입력한 문자를 ans의 변수에 저장합니다.
- if문을 통해 사용자가 y를 입력할 경우 ans에 저장 후 다음의 과정을 수행합니다.
- u.user라는 파일의 처음 10줄을 받아와 다음을 실행합니다.
- '|'를 구분자로 하여 u.item에서 n행의 1번째, 2번째 항을 빼옵니다.
- 3번째 항의 경우 삼항연산자를 통해 3번째 항이 'M'이면, 'male'을 아니면 'female' 을 출력하도록 하였습니다.
- if문의 작성을 종료하기 위해 fi를 써줍니다.
- 1개의 case를 종류하기 위해 ;;를 작성해 1개의 case가 끝났음을 알립니다.

제가 실행한 결과



Pdf가 요구한 결과

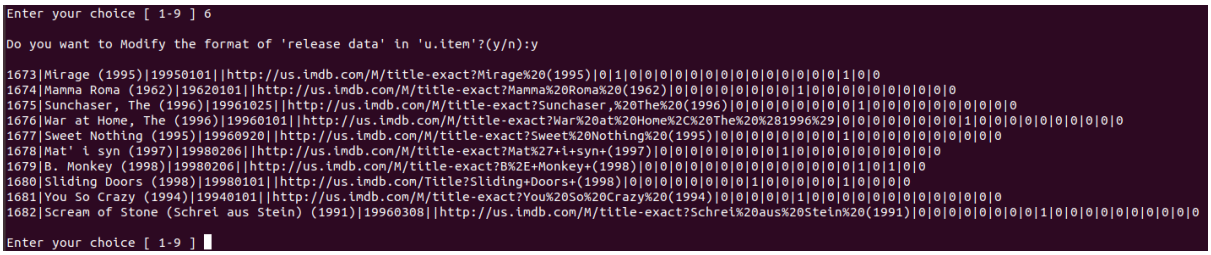


9. Code 7(Menu 6)

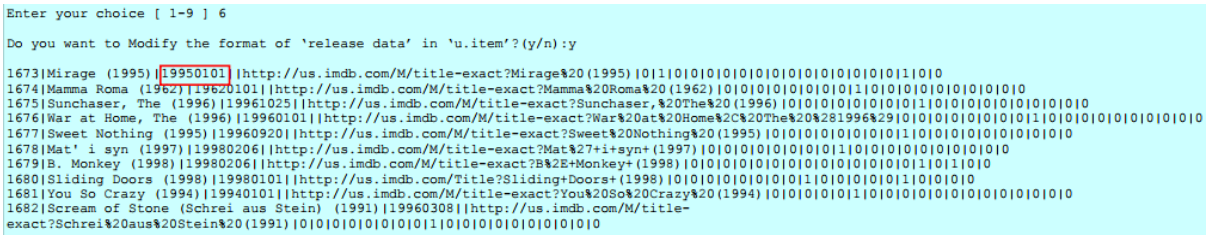
- ans는 사용자의 답변을 저장하기 위한 변수로 초기화를 진행하였습니다.
- 'read –p "문자열" 변수'를 통하여 문자열을 출력한 후 사용자가 입력한 문자를 ans의 변수에 저장합니다.
- if문을 통해 사용자가 y를 입력할 경우 ans에 저장 후 다음의 과정을 수행합니다.
 - 명령을 실행할 u.item의 지정된 문장을 순서를 통해 나타내기 count와 end 변수를 1673, 1682로 지정하였습니다.
- date, year, month, day를 통해 각각 해당 날짜 정보 및 년도, 달, 일을 변수를 초기화하였습니다.
- 새로운 형태로 해당 날짜 정보를 저장하기 위해 new_date라는 변수를 초기화하였습니다.
- while과 count 변수를 활용하여 u.item에서 가공해야 할 문장을 지정하고 line 변수에 저장합니다.
- 번호 순서에 맞는 형식을 u.item의 3번째 항을 date에 저장합니다.(01-Jul-25) 같은 형식
- data에서 –로 구분하여 1항인 일 수를 day에 저장합니다.
- data에서 –로 구분하여 2항인 달을 month에 저장합니다.
- data에서 –로 구분하여 1항인 년도를 year에 저장합니다.
- 각 case마다 new_data란 변수에 날짜에 대한 형식을 바꿔서 저장합니다.
- 각 case에 대해 정규표현식을 통해 시작은 I와 day변수로 시작하고 year변수와 |로 끝나면 해당 문자열을 |new_data변수|로 바꾸고 문장을 new_date_1에 저장해서 출력합니다.

- if문의 작성을 종료하기 위해 fi를 써줍니다.
- 1개의 case를 종류하기 위해 ;;를 작성해 1개의 case가 끝났음을 알립니다.

제가 실행한 결과



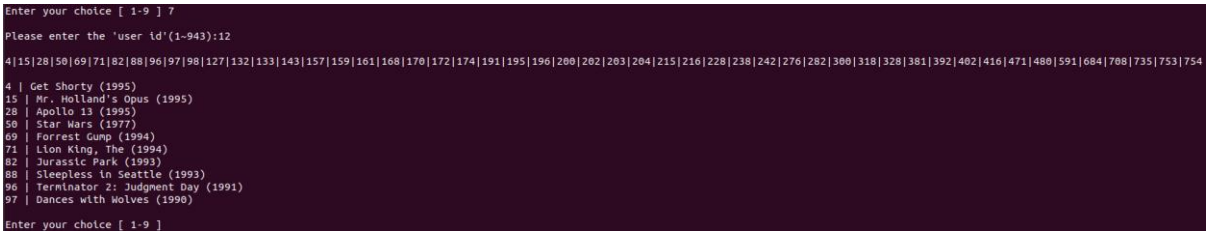
Pdf가 요구한 결과



10. Code 8(Menu 7)

- id는 사용자의 답변을 저장하기 위한 변수로 초기화를 진행하였습니다.
- 'read –p "문자열" 변수' 를 통하여 문자열을 출력한 후 사용자가 입력한 문자를 id의 변수에 저장합니다.
- u.data에서 1번째 항인 user id와 사용자가 입력한 id와 같을 경우 movie id를 출력하는 과정에서 오름차순으로 숫자를 정렬한 뒤 tr을 통해 숫자가 띄어 지는 것을 구분자를 '|'로 설정해서 사용자가 입력한 id와 같은 user id가 본 movie id를 출력합니다.
- 사용자가 입력한 id와 같은 user id가 본 movie id를 정렬 및 올림차순으로 10개 추출해서 num 변수에 저장합니다.
- 이후 num을 numbers 라는 배열로 형식을 바꿔 저장합니다.
- numbers의 요소, movie id 와 같은 경우 u.item에서 2번째 항목인 영화 제목을 받아와 movie_name에 저장합니다.
- "$number | $movie_name" 같은 형식으로 사용자가 입력한 id와 같은 user id가 본 movie id 와 영화 제목을 위와 같은 형식으로 출력합니다.
- if문의 작성을 종료하기 위해 fi를 써줍니다.
- 1개의 case를 종류하기 위해 ;;를 작성해 1개의 case가 끝났음을 알립니다.

제가 실행한 결과



Pdf가 요구한 결과

```
Enter your choice [ 1-9 ] 7

Please enter the 'user id'(1~943):12

4|15|28|50|69|71|82|88|96|97|98|127|132|133|143|15
7|159|161|168|170|172|174|191|195|196|200|202|203|
204|215|216|228|238|242|276|282|300|318|328|381|39
2|402|416|471|480|591|684|708|735|753|754

4|Get Shorty (1995)
15|Mr. Holland's Opus (1995)
28|Apollo 13 (1995)
50|Star Wars (1977)
69|Forrest Gump (1994)
71|Lion King, The (1994)
82|Jurassic Park (1993)
88|Sleepless in Seattle (1993)
96|Terminator 2: Judgment Day (1991)
97|Dances with Wolves (1990)
```

11. Code 9(Menu 8)

- ans는 사용자의 답변을 저장하기 위한 변수로 초기화를 진행하였습니다.
- 'read –p "문자열" 변수'를 통하여 문자열을 출력한 후 사용자가 입력한 문자를 ans의 변수에 저장합니다.
- if문을 통해 사용자가 y를 입력할 경우 ans에 저장 후 다음의 과정을 수행합니다.
- u.user에서 2번째 항에서 나이가 20이상이고 29세 이하이며 4번째 항을 통해 직업이 programmer인 사람의 id를 뽑아 num변수에 저장합니다.
- 해당 조건에 만족하는 사람이 작성한 movie 종류와 그에 따른 rating을 행으로 작성한 문자열을 new_row라는 변수에 저장하고 이후 new_data라는 변수에 줄을 나눠 업데이트 합니다.
- new_data에는 movie id와 rating 정보가 개행 문자로 구분해서 들어가 있습니다.
- awk NF를 통해 비어있는 줄은 제거합니다.
- split을 통해 new_data 문자열을 개행 문자를 기준으로 나누어 rows 배열에 저장합니다.
- rows 배열을 순회하면서 각 행을 cols 배열에 공백을 기준으로 나누어 저장합니다. rows[0]에는 movie id가 rows[1]에는 rating이 저장됩니다.
- movie id를 키로 하는 개수를 증가시킵니다. rating에 따른 평균을 계산하기 위해 movie id가 같은 것을 합산했을 경우 몇 번 합산했는지에 관한 정보를 저장합니다.
- for문에서 추가로 movie id 의 count가 0이면 합계도 0이라서 나눌 경우 0을 나누는 오류가 발생하기에 따로 경우를 분리했습니다.
- movie id에 따른 평균을 sum[key] / count[key]로 계산하여 소수점 5자리까지 나타냅니다.
- 다음 명령어에서 소수점 5자리까지 나타냈지만 소수점 5까지 필요없는데도 쓰인 불필요한 0을 제거합니다.

- 계산된 'movie id'와 해당 'movie id'에 대한 평균을 출력합니다
- sort를 통해 'movie id'를 기준으로 정렬하여 출력합니다.
- if문의 작성을 종료하기 위해 fi를 써줍니다.
- 1개의 case를 종류하기 위해 ;;를 작성해 1개의 case가 끝났음을 알립니다.

제가 실행한 결과

```
Enter your choice [ 1-9 ] 8

Do you want to get the average 'rating' of movies rated by users with 'age' between 20 and 29 and 'occupation' as 'programmer'?(y/n):y
1 4.29412
2 3
3 3.5
4 3.7
5 3.25
7 4.22222
8 3.5
9 4.1
10 4
11 4.3125
12 4.69231
13 3.375
14 4
15 3.85714
16 3
17 3.5
19 4
20 1
```

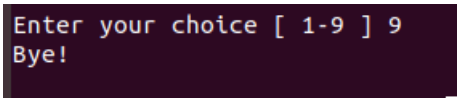
Pdf가 요구한 결과

```
Enter your choice [ 1-9 ] 8

Do you want to get the average 'rating' of
movies rated by users with 'age' between 20 and
29 and 'occupation' as 'programmer'?(y/n):y

1 4.29412
2 3
3 3.5
4 3.7
5 3.25
7 4.22222
8 3.5
9 4.1
10 4
11 4.3125
.
.
.
1512 3
1513 2
1518 4
1531 3
1552 2
1597 1
1600 4
1621 1
1655 2
```

12. Code 11(End)

- Bye!를 출력하고 exit를 통해 while문을 벗어납니다.
- if문의 작성을 종료하기 위해 fi를 써줍니다.
- 1개의 case를 종류하기 위해 ;;를 작성해 1개의 case가 끝났음을 알립니다.

제가 실행한 결과



Pdf가 요구한 결과

```
Enter your choice [ 1-9 ] 9
Bye!
```

13. Code 10(Menu 9)

- esac으로 case문 작성을 마치는 것을 알려줍니다..
- done을 통해 전체 while문의 작성을 마칩니다.

14. 코드 및 주석

#! /bin/sh
item=$1 #'u.item' 파일의 경로를 저장합니다.
data=$2 #'u.data' 파일의 경로를 저장합니다
user=$3 #'u.user. 파일의 경로를 저장합니다
num=0 #이후 메뉴에 따른 사용자의 선택을 받기 위한 변수로, case 선택에 사용됩니다.

#아래부터는 각각에 번호에 해당하는 메뉴를 출력합니다.
echo "--------------------------"
echo "User Name: $(whoami)" #사용자의 이름을 표시합니다.
echo "Student Number: 12215529"
echo "[MENU]"
echo "1. Get the data of the movie identified by a specific 'movie id' from 'u.item'"
echo "2. Get the data of action genre movies from 'u.item'"
echo "3. Get the average 'rating' of the movie identified by specific 'movie id' from 'u.data'"
echo "4. Delete the 'IMDb URL' from 'u.item'"
echo "5. Get the data about users from 'u.user'"
echo "6. Modify the format of 'release data' in 'u.item'"
echo "7. Get the data of movies rated by a specific 'user id' from 'u.data'"
echo "8. Get the average 'rating' of movies rated by users with 'age' between 20 and 29 and 'occupation' as 'programmer'"
echo "9. Exit"
echo '--------------------------'
while : #9번 명령을 받을 때까지 무한으로 반복합니다.
do
    read -p "Enter your choice [1-9] " num #사용자가 선택한 메뉴를 입력받아 num에 저장합니다.

case $num in #사용자에 선택(num)에 따라 각 case를 실행합니다.

1)

echo #한 줄을 띄웁니다.

fir=0 #movie id를 받기 위한 변수를 초기화합니다.

read -p "Please enter 'movie id'(1~1682) : " fir #해당 문자열을 출력하고 movie id를 입력받아 fir에 저장합니다.

echo

cat $item | sed -n "${fir}p" #cat을 통해 u.item 전체를 읽고 sed를 통해 사용자가 입력한 줄을 받아온다.

echo

;; #1번 case를 종료합니다.

2)

echo

ans=0

read -p "Do you want to get the data of 'action' genre movies from 'u.item'?(y/n):" ans

#해당 문자열을 출력하고 사용자의 입력을 받습니다.

echo

if [${ans} = 'y']; then #사용자가 y를 입력할 경우 ans에 저장 후 다음의 과정을 수행합니다.

awk -F'|' '$7 == 1 {print $1, $2; count++; if(count == 10) exit}' $item

파일의 처음 10줄을 읽어와서 각 사용자의 정보를 출력합니다.

u.item에서 '|'를 기준으로 u.item의 7번째 항, action 장르가 맞을 경우 해당 행의 1항과 2항을 출력합니다. count를 통해 10개 줄까지만 받습니다.

#u.item을 |를 기준으로 나누고 그 행의 첫 번째, 두 번째 요소를 10개를 출력합니다.

fi #if문을 끝냅니다.

echo

;; #2번 case를 종료합니다.

3)

echo

thr=0

total_sum=0 #사용자들이 movie의 rating을 모두 합산합니다.

count=0 # movie의 rating의 합산을 평균내기 위한 변수입니다.

num_round1=0 #소수 계산을 위한 첫 번째 변수입니다.

num_round2=0 #소수 계산을 위한 두 번째 변수입니다.

num_round3=0 #소수 계산을 위한 세 번째 변수입니다.

average=0 #평균을 나타내는 변수를 초기화 합니다.

read -p "Please enter the 'movie id'(1~1682) : thr " #movie id를 입력받아 thr에 저장합니다.

echo

while read -r num1 num2 num3 num4; do

```
            if [ "$num2" -eq "$thr" ]; then
                total_sum=$((total_sum + num3))
                # 'movie id'와 일치하는 경우, 해당 영화의 'rating'을 total_sum에 더합니다.
                count=$((count + 1))
                # count를 증가시켜 'rating' 합산을 위한 영화의 개수를 계속 더합니다..
            fi
        done < "$data" # u.data 파일로부터 데이터를 읽어옵니다.
        if [ $count -gt 0 ]; then
            num_round1=$(echo "scale=6; $total_sum / $count" | bc)
            # 소수 이하 다섯 자리까지 반올림합니다.
            num_round2=$(echo "scale=5; $total_sum / $count" | bc)
            # 평균을 계산한 값의 소수점 이하 다섯 자리까지 계산합니다.
            num_round3=$(echo "scale=0; ($num_round1*1000000 - $num_round2*1000000)" | bc)
            # 소수점 이하 차이를 계산합니다.
            num_round3=$(echo "$num_round3 / 1" | bc) #소수형을 정수 형태로 만듭니다.
            if [ $num_round3 -gt 4 ]; then
                average=$(echo "scale=5; ($num_round1*1000000.0 + 10.0) / 1000000.0" | bc)
                # 소수 이하 다섯 자리까지 반올림합니다.
            else
                average=$(printf "%.0f" $num_round2)
                 # 다섯 자리에서 반올림된 값이 4 이하인 경우, 다섯 자리까지 버림하여 정수로 출
력합니다.
            fi
        fi
        echo "average rating of $thr : $average" # 계산된 평균 'rating'을 출력합니다.
        echo
        ;; #3번 case를 종료합니다.
        #해당 형식의 알고리즘 같은 경우 소수점 5자리 이하로 계산될 때는 소수점 계산을 진행
하고 소수점 5자리까지 나타낼 필요가 없는 0은 지웁니다.
        #소수점 5자리 이상인 경우에는 소수점 6자리까지 나타내고 1,000,000을 곱해서 자연수로
나타내주고,
        4)
        echo
        ans=0
        read -p "Do you want to delete the 'IMDb URL' from 'u.item'?(y/n):" ans
        #해당 문자열을 출력하고 사용자의 입력을 받습니다.
        echo
        if [ ${ans} = 'y' ]; then #
        #사용자가 y를 입력할 경우 ans에 저장하고 다음의 과정을 수행합니다.
```

```
            head -n 10 "$item" | sed -E 's/http([^|]*|)//'
            # 파일의 처음 10줄을 읽어 'IMDb URL'을 삭제하고 출력합니다.
            # 다음의 정규 표현식은 'http'로 시작하고 '|'로 끝나는 경우 해당 내용을 제거합니다.
        fi
        echo
        ;; #4번 case를 종료합니다.
        5)
        echo
        ans=0
        read -p "Do you want to get the data about users from 'u.user'?(y/n):" ans
        #해당 문자열을 출력하고 사용자의 입력을 받습니다.
        echo
        if [ ${ans} = 'y' ]; then
        #사용자가 y를 입력할 경우 ans에 저장하고 다음의 과정을 수행합니다.
            head -n 10 $user | awk -F'|' '{print "user",$1,"is",$2,"years old",($3 == "M" ? "male" : "female"),$4}'
            #u.user라는 파일의 처음 10줄을 받아와 다음을 실행합니다.
            # '|'를 구분자로 하여 u.item에서 n행의 1번째, 2번째 항을 뺴옵니다.
            # 3번째 항의 경우 삼항연산자를 통해 3번째 항이 'M'이면 , 'male'을 아니면 'female'을 출력하도록 하였습니다.
        fi
        echo
        ;; #5번 case를 종료합니다.
        6)
        echo
        ans=0
        read -p "Do you want to Modify the format of 'release data' in 'u.item'?(y/n):" ans
        #해당 문자열을 출력하고 사용자의 입력을 받습니다.
        echo
        if [ ${ans} = 'y' ]; then
        #사용자가 y를 입력할 경우 ans에 저장하고 다음의 과정을 수행합니다.
          count=1673 #출력할 문장의 시작 지점을 나타내는 변수입니다.
          end=1682 #출력할 문장의 끝 지점을 나타내는 변수로 초기화하였습니다.
          date=0 #년도, 달, 일에 대한 정보를 나타내는 변수로 초기화하였습니다.
          year=0 #년도를 나타내는 변수로 초기화하였습니다.
          month=0 #달을 나타내는 변수로 초기화하였습니다.
          day=0 #일수를 나타내는 변수로 초기화하였습니다.
          new_date=0 #년도, 달, 일에 대한 정보를 나타내는 변수로, 새로운 형태이며 초기화하였습니다.
```

```
while [ "$count" -le "$end" ]; do #1673번째 줄부터 1682줄까지 실행
    line=$(awk -v line="$count" 'NR==line' "$item")#번호 순서에 맞는 행을 line에 저장합니다.
    date=$(awk -v line="$count" 'NR==line' "$item" | awk -F '|' '{print $3}')
    #")#번호 순서에 맞는 godd,; 3번째 항을 date에 저장합니다.(01-Jul-25) 같은 형식
    day=$(echo "$date" | cut -d'-' -f1) #data에서 –로 구분하여 1항인 일 수를 day에 저장합니다.
    month=$(echo "$date" | cut -d'-' -f2) #data에서 –로 구분하여 2항인 달을 month에 저장합니다.
    year=$(echo "$date" | cut -d'-' -f3) #data에서 –로 구분하여 1항인 년도를 year에 저장합니다.


    case "$month" in #case문을 통해 month에 써져 있는 영문 표기의 달을 숫자로 변경
        Jan) #1월일 경우
            new_date="${year}01${day}"#Jan을 01로 바꿔서 new_data에 저장
            new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
            #정규표현식을 통해 시작은 I와 day변수로 시작하고 year변수와 |로 끝나면 해당 문자열을 |new_data변수|로 바꾸고 new_date_1에 저장한다.
            echo $new_date_1
            # 새롭게 만들어진 new_date가 포함된 행으로 출력한다
            ;;
        Feb) #2월일 경우
            new_date="${year}02${day}"
            new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
            echo $new_date_1
            ;;
        Mar) #3월일 경우
            new_date="${year}03${day}"
            new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
            echo $new_date_1
            ;;
        Apr) #4월일 경우
            new_date="${year}04${day}"
            new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
            echo $new_date_1
            ;;
        May) #5월일 경우
            new_date="${year}05${day}"
```

```
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
    Jun) #6월일 경우
        new_date="${year}06${day}"
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
    Jul) #7월일 경우
        new_date="${year}07${day}"
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
    Aug) #8월일 경우
        new_date="${year}08${day}"
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
    Sep) #9월일 경우
        new_date="${year}09${day}"
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
    Oct) #10월일 경우
        new_date="${year}10${day}"
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
    Nov) #11월일 경우
        new_date="${year}11${day}"
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
    Dec) #12월일 경우
        new_date="${year}12${day}"
        new_date_1=$(echo "$line" | sed -E "s/₩|$day([^|]*$year₩|)/₩|$new_date₩|/")
        echo $new_date_1
        ;;
esac #달에 따른 case를 종료합니다.
```

```
        count=$((count + 1))#을 통해서 진행시킨다.
    done
    fi
    echo
    ;; #6번 case를 종료합니다.
    7)
    echo
    item=$1
    data=$2
    id=0
    num=0
    read -p "Please enter the 'user id'(1~943):" id
    echo
    awk -F' ' '$1 == '"$id"' {print $2}' "$data" | sort -n | tr '₩n' '|' | sed 's/.$//'
```

#u.data에서 1번째 항인 user id와 사용자가 입력한 id와 같을 경우 movie id를 출력하는 과정에서 오름차순으로 숫자를 정렬한 뒤 tr을 통해 숫자가 띄어 지는 것을 구분자를 '|'로 설정해서 사용자가 입력한 id와 같은 user id가 본 movie id를 출력합니다.

```
    num=$(awk -F' ' '$1 == '"$id"' {print $2}' "$data" | sort -n | head -n 10)
```

사용자가 입력한 id와 같은 user id가 본 movie id를 정렬 및 올림차순으로 10개 추출해서 num 변수에 저장합니다.

```
    echo
    echo

    numbers=($num) #num을 numbers 라는 배열로 형식을 바꿔 저장합니다.

    for number in "${numbers[@]}"; do #numbers의 배열의 요소(movie id)를 하나씩 받습니다.
        movie_name=$(awk -v num="$number" -F'|' '$1 == num { print $2}' "$item")
```

#numbers의 요소, movie id 와 같은 경우 u.item에서 2번째 항목인 영화 제목을 받아와 movie_name에 저장합니다.

```
        echo "$number | $movie_name" #다음과 같은 형식으로 사용자가 입력한 id와 같은 user id가 본 movie id 와 영화 제목을 위와 같은 형식으로 출력합니다.
    done
    echo
    ;; #7번 case를 종료합니다.
    8)
    echo
    ans=0
    read -p "Do you want to get the average 'rating' of movies rated by users with 'age'
```

between 20 and 29 and 'occupation' as 'programmer'?(y/n):" ans
#해당 문자열을 출력하고 사용자의 입력을 받습니다.
if [${ans} = 'y']; then
#사용자가 y를 입력할 경우 ans에 저장하고 다음의 과정을 수행합니다.
num=$(awk -F'|' '$2 >= 20 && $2 <= 29 && $4 == "programmer" {print $1}' "$user")
#u.user에서 2번째 항에서 나이가 20이상이고 29세 이하이며 4번째 항을 통해 직업이 programmer인 사람의 id를 뽑아 num변수에 저장합니다.
new_data="" # new_data를 초기화합니다.
for number in $num; do
new_row=$(awk -F' ' -v num="$number" '$1 == num {print $2, $3}' "$data")
new_data+="$new_row"$'₩n'
해당 조건에 만족하는 사람이 작성한 movie 종류와 그에 따른 rating을 행으로 작성한 문자열을 new_row라는 변수에 저장하고 이후 new_data라는 변수에 줄을 나눠 업데이트 합니다. new_data에는 movie id와 rating 정보가 개행 문자로 구분해서 들어가 있습니다.
done

new_data=$(echo "$new_data" | awk NF)
# awk NF를 통해 비어있는 줄은 제거합니다.

awk -v new_data="$new_data" 'BEGIN {
split(new_data, rows, "₩n"); # new_data 문자열을 개행 문자를 기준으로 나누어 rows 배열에 저장합니다.
for (i in rows) {
split(rows[i], cols, " "); #rows 배열을 순회하면서 각 행을 cols 배열에 공백을 기준으로 나누어 저장합니다. rows[0]에는 movie id가 rows[1]에는 rating이 저장됩니다.
sum[cols[1]] += cols[2]; # movie id를 key로 해서 movie id가 같은 경우 rating을 합산해서 sum[movie id]에 저장합니다.
count[cols[1]]++; # movie id를 키로 하는 개수를 증가시킵니다. rating에 따른 평균을 계산하기 위해 movie id가 같은 것을 합산했을 경우 몇 번 합산했는지에 관한 정보를 저장합니다.
}
for (key in sum) {
if (count[key] == 0) {
#movie id 의 count가 0이면 합계도 0이라서 나눌 경우 0을 나누는 오류가 발생하기에 따로 경우를 분리했습니다.
average = 0;
} else {
average = sprintf("%.5f", sum[key] / count[key]);
#movie id에 따른 평균을 sum[key] / count[key]로 계산하여 소수점 5자리까지

나타냅니다.

sub("₩₩.?0+$", "", average);#소수점 5자리까지 나타냈지만 소수점 5까지 필요 없는데도 쓰인 불필요한 0을 제거합니다.

}

printf "%s %s₩n", key, average;

#계산된 'movie id'와 해당 'movie id'에 대한 평균을 출력합니다.

}

}' | sort -n -k1 # 'movie id'를 기준으로 정렬하여 출력합니다.

fi

echo

;; #8번 case를 종료합니다.

9)

echo "Bye!"

echo

exit #루프에서 나옵니다.

;; #9번 case를 종료합니다.

esac #case문 작성을 마칩니다.

done #while문을 작성을 마칩니다.