**CSE2113-002 오픈소스 SW개론 Project 1**

SID. 12191684

Name. Byeong Su Choi

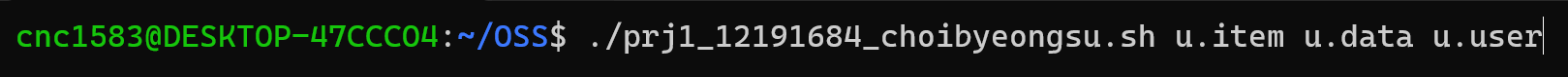
E-mail. [cnc1583@naver.com](mailto:cnc1583@naver.com)

시작하기에 앞서 해당 README 파일은 Project 1 과제 제출을 위한 스크립트를 설명하기 위한 것으로 하단에 서술할 예정인 u.item, u.data, u.user의 배포 권한은 학생 본인에게 있지 않아 참고하시길 바랍니다.

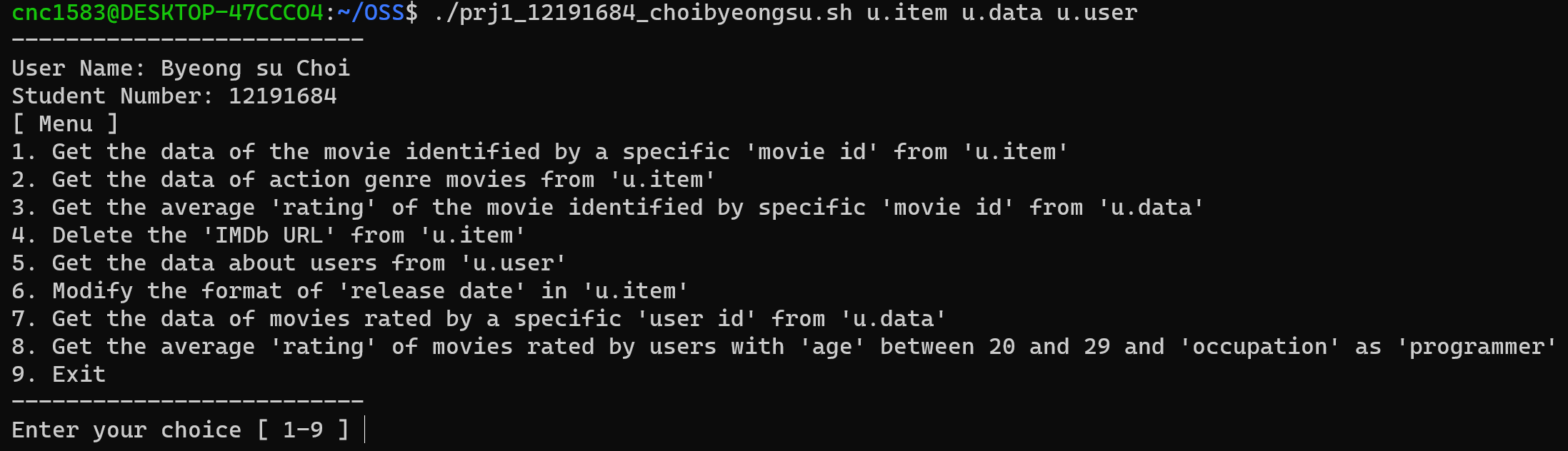
**Index**

1. 사용법
2. 구현

**1. 사용법**



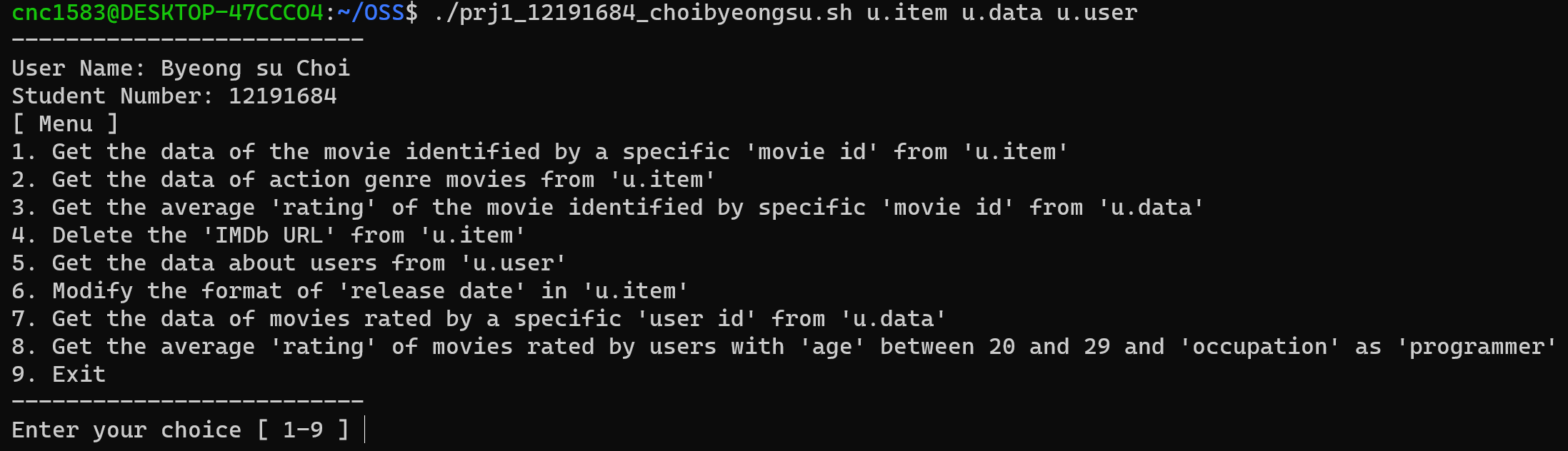
위 사진과 같이 prj1\_12191684\_choibyeongsu.sh를 실행 파일로, u.item u.data u.user 세 가지 파일을 인자로 넘겨주면 됩니다. 위치가 바뀔 경우 정상적인 결과를 도출하지 않습니다. 순서에 유의해 주시길 바랍니다.



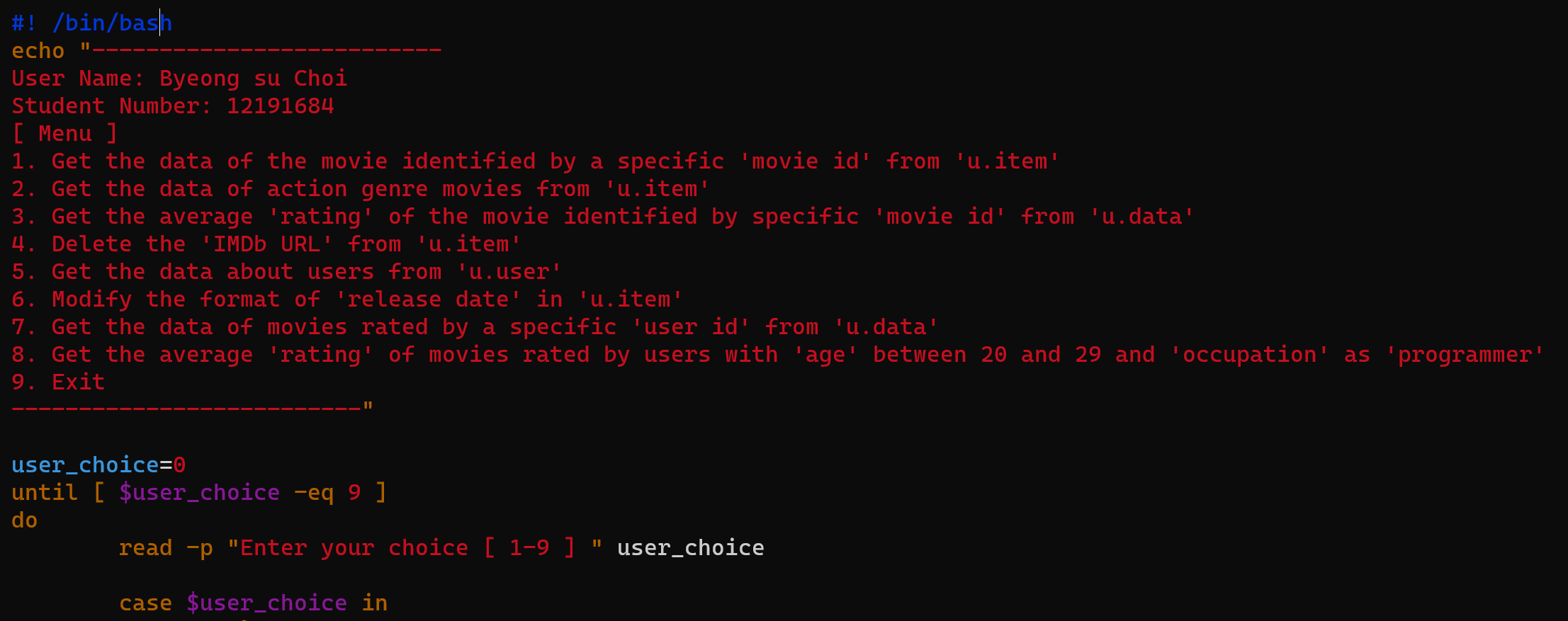
실행이 완료되면 위와 같은 화면이 출력 되므로 원하는 선택지로 프로그램을 조작하시면 됩니다.

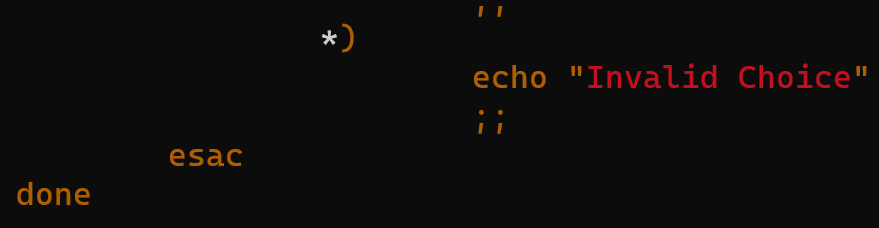
**2. 구현**

1. **최초 실행**



처음 파일을 실행 시키면 선택지가 텍스트로 주어지며 원하시는 숫자를 입력하시면 됩니다.



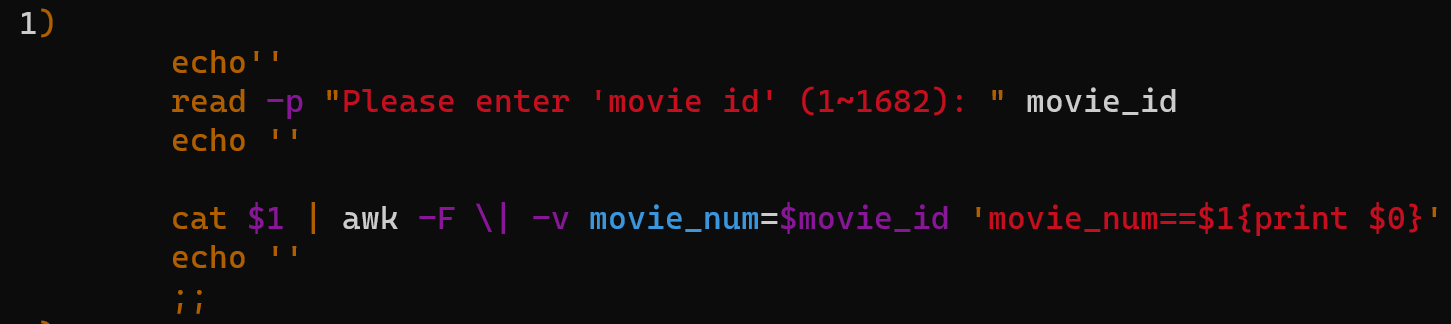


echo 명령 한 번에 최초 선택지 텍스트를 출력하고 until문으로 사용자가 입력한 숫자가 9가 될 때 까지 입력을 받을 수 있도록 반복문을 돌렸습니다.

read -p 명령으로 user\_choice 변수에 값이 저장이 됩니다.

사용자가 선택한 숫자는 case문을 통해 각각 처리 해 주었고 유효하지 않은 선택지에 대해서 Default 경우로 “Invalid Choice”를 출력합니다.

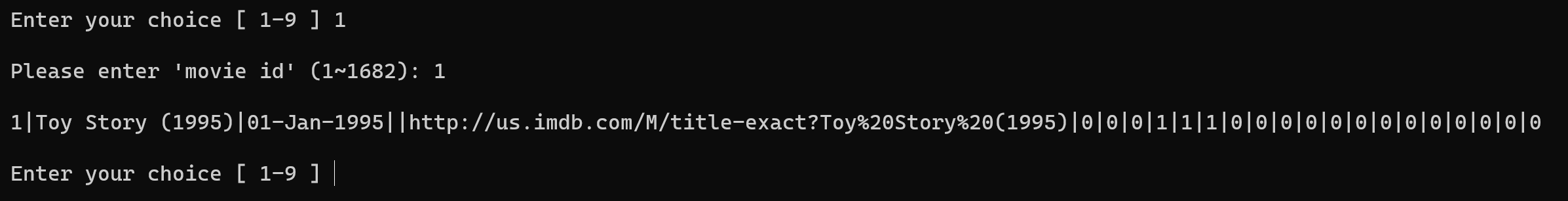
1. **Get the data of the movie identified by a specific 'movie id' from 'u.item'**



movie id를 입력받습니다. 파일 실행 시 u.item을 첫 번째 매개변수로 두었기 때문에 cat $1으로 u.item의 파일을 열어줍니다.

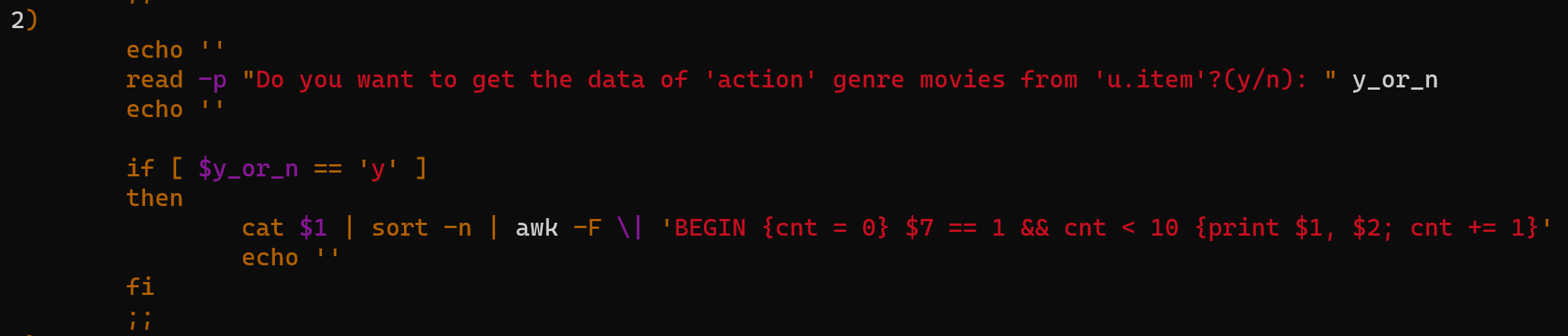
awk 명령어를 통해 “|”로 구분되어 있다는 점을 -F 옵션으로 처리, 변수를 사용하기 위해 -v 명령어로 입력받은 movie\_id를 내부 변수 movie\_num에 저장 해 둡니다.

“|”로 구분된 텍스트 기준 1번째 열 중 movie\_num, 즉 movie id와 일치하는 번호가 있으면 해당 행을 출력합니다.



실행 결과입니다.

1. **Get the data of ‘action’ genre movies from 'u.item’**

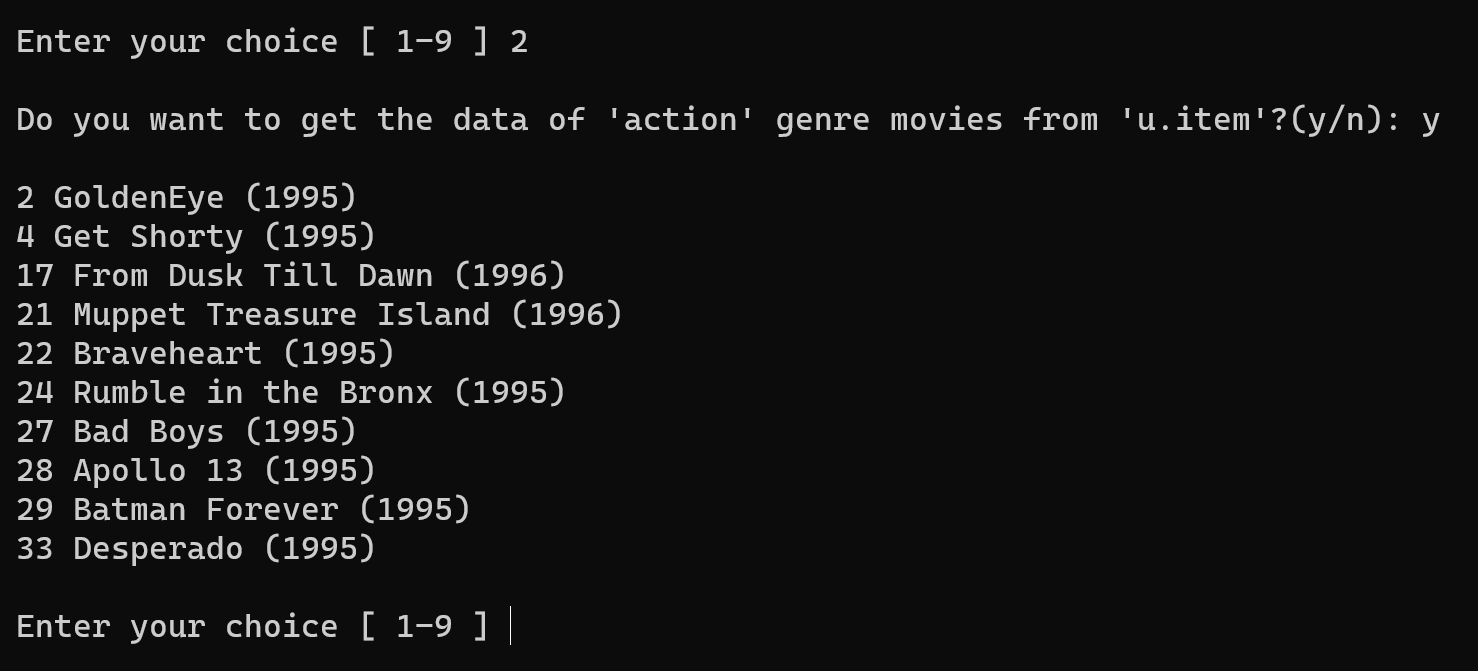


yes or no의 선택지가 주어졌으므로 if문을 통해 y를 입력한 경우에 대해서만 처리를 진행 해 줍니다.

파일 실행 시 u.item을 첫 번째 매개변수로 두었기 때문에 cat $1으로 u.item의 파일을 열어줍니다. 이후 movie id 순으로 정렬해야 하므로, 각 행의 첫 번째 문자가 movie id, 숫자로 정렬시키기 위해 sort -n 옵션으로 정렬해 주었습니다.

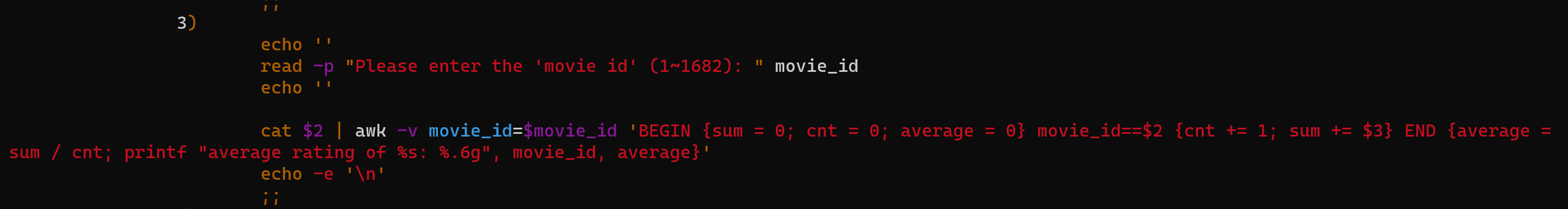
awk를 명령어를 통해 “|”로 구분되어 있다는 점을 -F 옵션으로 처리, 탐색을 시작하기 전 10개의 행만 출력해야 하므로 cnt 변수를 0으로 초기화 하는 부분을 BEGIN으로 만들어 주었습니다.

“|”로 구분된 텍스트 기준 7번째 열이 1이고 cnt가 10보다 작을 경우 해당 행을 print 함과 동시에 cnt를 1 늘려줍니다.



실행 결과입니다.

1. **Get the average 'rating’ of the movie identified by specific 'movie id' from 'u.data’**



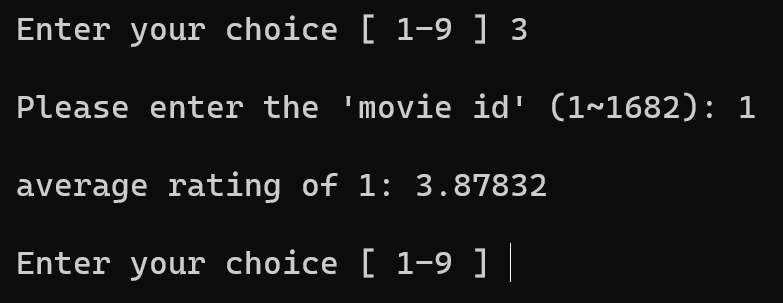
movie\_id에 입력받은 값을 저장 해 줍니다.

파일 실행 시 u.data를 두 번째 매개변수로 두었기 때문에 cat $2으로 u.data의 파일을 열어줍니다.

awk 명령어를 통해 외부변수 movie\_id를 같은 이름의 내부 변수로 사용, 이를 처리하기 위해 -v 옵션으로 처리 해 줍니다.

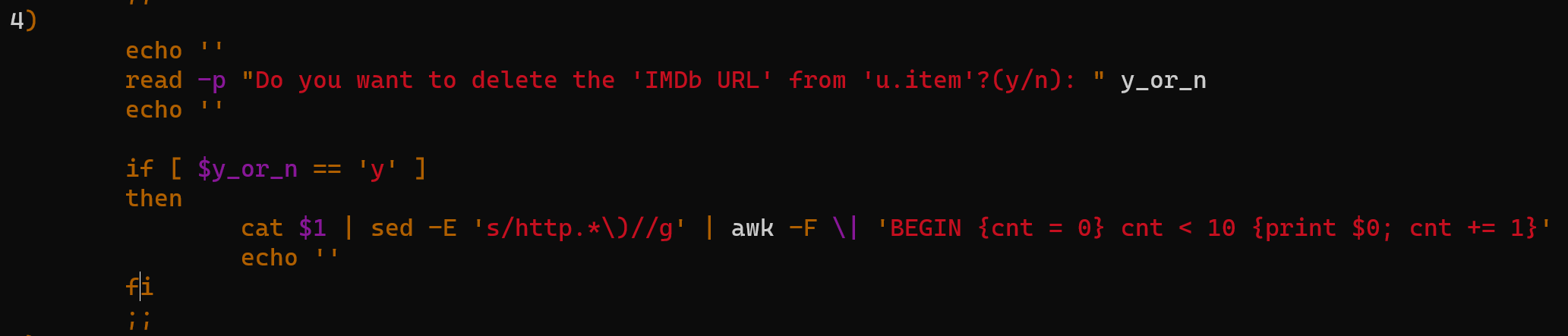
탐색을 시작하기 전 BEGIN으로 평균 계산을 위한 총합(sum), 개수(cnt), 평균(average)를 0으로 초기화, 2번 째 열이 movie\_id와 같은 행에 대해서 3번 째 열, 즉 rating 점수를 sum에 더해주고 cnt를 1 증가 시킵니다.

탐색을 모두 마친 후 평균을 계산하고 movie id와 평균을 출력합니다. 이 때 평균은 6번째 자리에서 반올림 해야하므로 출력 형식을 %.6g로 지정 해 줍니다.



실행 결과입니다.

1. **Delete the ‘IMDb URL’ from ‘u.item’**



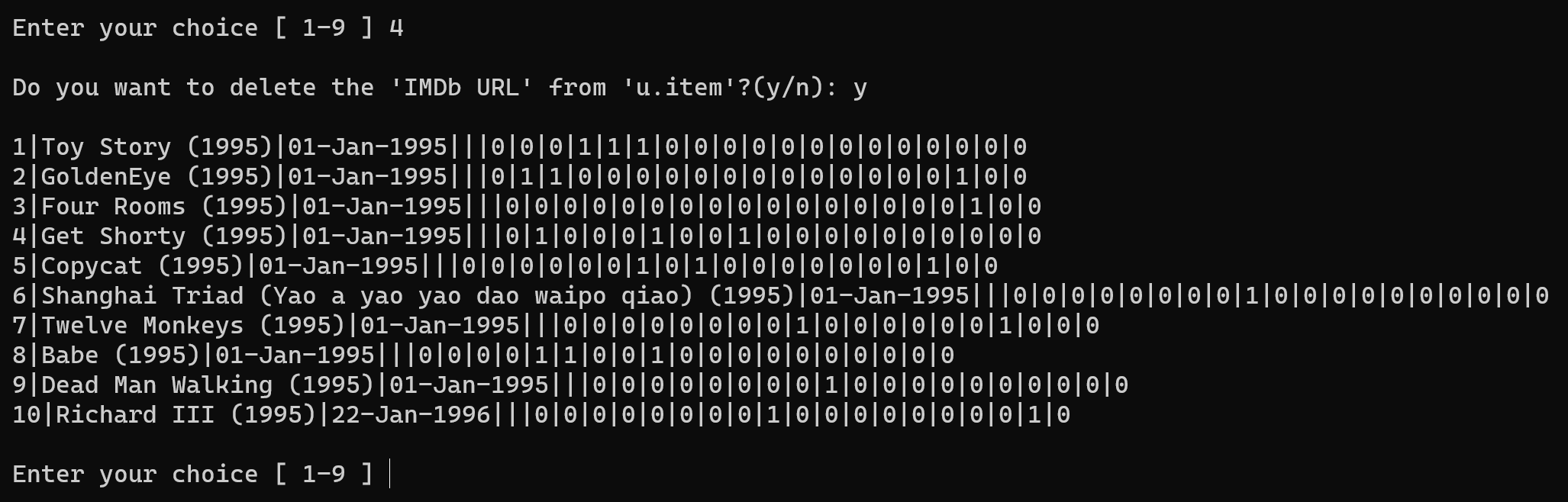
y\_or\_n에 입력받은 값을 저장해 줍니다.

if문을 통해 y를 입력받은 경우에만 처리를 진행 해 주도록 합니다.

파일 실행 시 u.item을 첫 번째 매개변수로 두었기 때문에 cat $1으로 u.item의 파일을 열어줍니다.

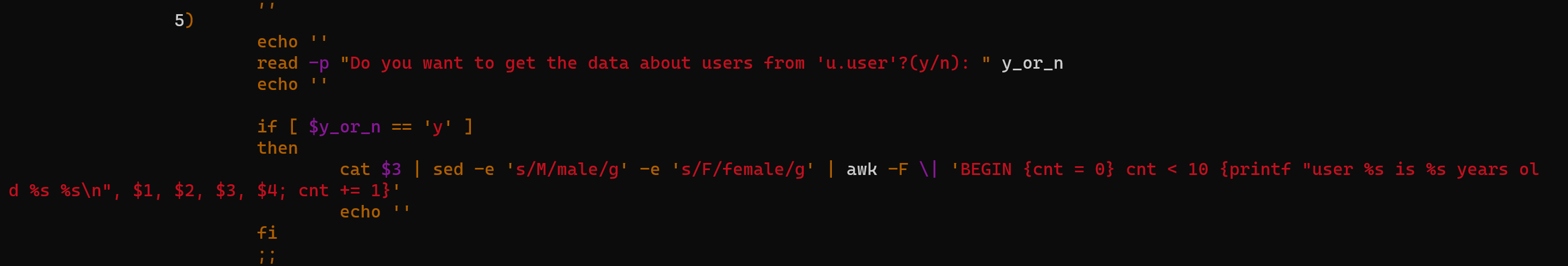
sed 명령어를 통해 http.으로 시작하고 )로 끝나는 모든 문자열을 공백으로 대체(s)합니다. 대체한 결과를 토대로 10개의 행을 출력하기 위해 awk 명령어를 사용해 줍니다.

마찬가지로 10개의 행만 출력하기 위해 BEGIN으로 cnt 변수를 0으로 초기화 하여 행을 출력할 때 마다 cnt를 1 증가시켜 10개의 행만 출력 해 줍니다.



실행 결과입니다.

1. **Get the data about users from 'u.user’**

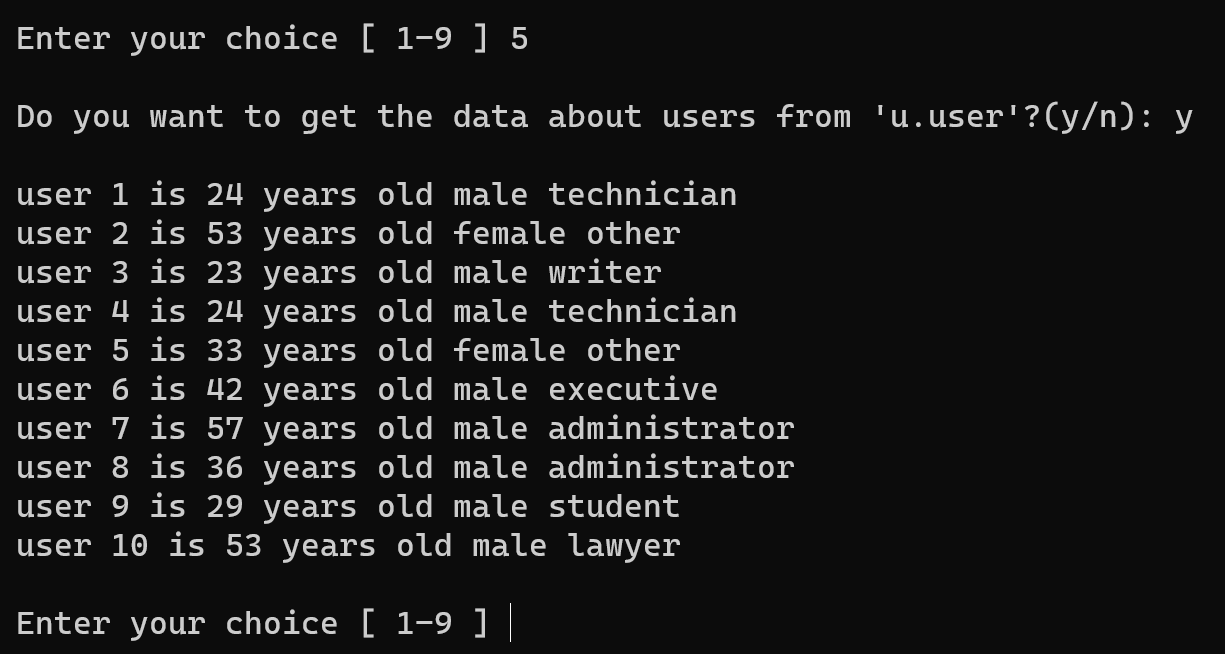


파일 실행 시 u.user를 세 번째 매개변수로 두었기 때문에 cat $3으로 u.user의 파일을 열어줍니다.

출력 형식을 일치시키기 위해 sed 명령어로 모든 M을 male, F를 female로 대체(s)합니다. 조건이 2개 이상이므로 -e로 연결 해 줍니다.

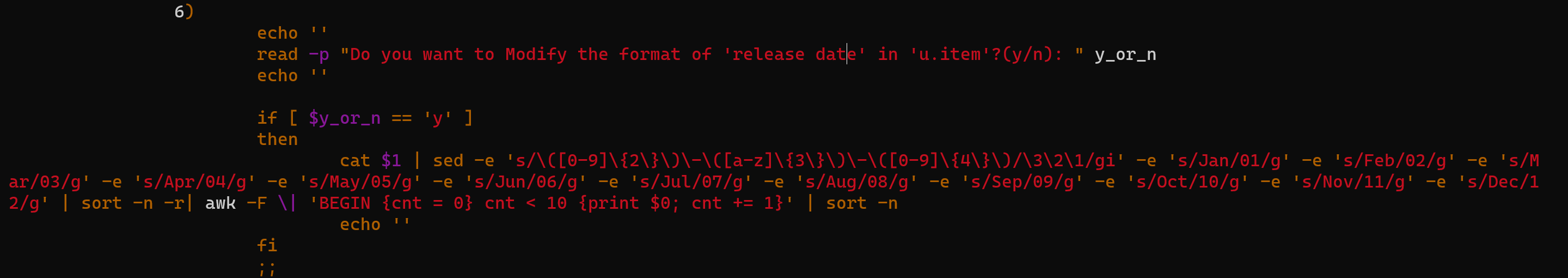
결과를 토대로 5)와 같은 방식으로 awk 명령어를 통해 “|”로 구분되어 있다는 점을 -F를 통해 처리, BEGIN으로 cnt를 0으로 초기화하여 10개의 행만 출력 해 주도록 합니다.

printf를 통해 출력 형식을 맞춰 출력 했습니다.



실행 결과입니다.

1. **Modify the format of 'release date' in 'u.item’**



y\_or\_n에 입력받은 값을 저장 해 줍니다.

if문을 통해 y를 입력받은 경우에만 처리를 해 주었습니다.

파일 실행 시 u.item을 첫 번째 매개변수로 두었기 때문에 cat $1으로 u.item의 파일을 열어줍니다.

sed 명령어를 사용, 날짜 형식 dd-mmm-yyyy를 yyyymmdd로 변경하기 위해 범위 지정을 통해 탐색하여 위치를 변경 해 줍니다. dd는 0~9 숫자 2번 mmm은 a-z 영어 3번 (대소문자 구분을 제거하기 위해 i 옵션) yyyy는 0~9 숫자 4번으로 탐색 해 주었습니다. 유의할 점은 괄호와 -기호를 문자로 인식시키기 위해 역슬래쉬를 사용해주어야 하는 점 입니다.

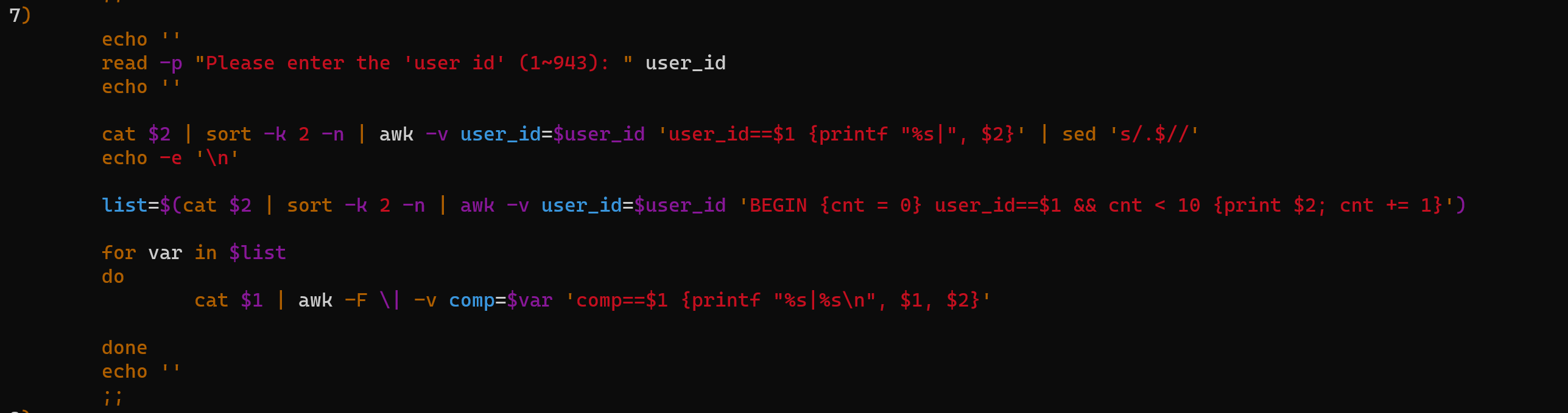
이후 3개로 그룹화 된 형식의 순서를 \3\2\1로 변경 해 준 뒤, 영어로 된 mmm을 숫자 mm 형식으로 변환해 주기 위해 Jan ~ Dec까지 대응되는 날짜 축약어를 숫자로 변환합니다. 다중 조건을 부여하므로 -e 옵션으로 연결 해 주었습니다.

sort -n -r 옵션으로 movie id 기준 내림차순 정렬을 한 뒤 awk 명령어를 통해 10개의 행만을 출력합니다. 따라서 하위 10개의 행을 뽑아낼 수 있고, 10개의 행이 다시 movie id 기준 오름차순 정렬이 되어야 하므로 마지막으로 sort -n으로 정렬합니다.



실행 결과입니다.

1. **Get the data of movies rated by a specific 'user id' from 'u.data'**



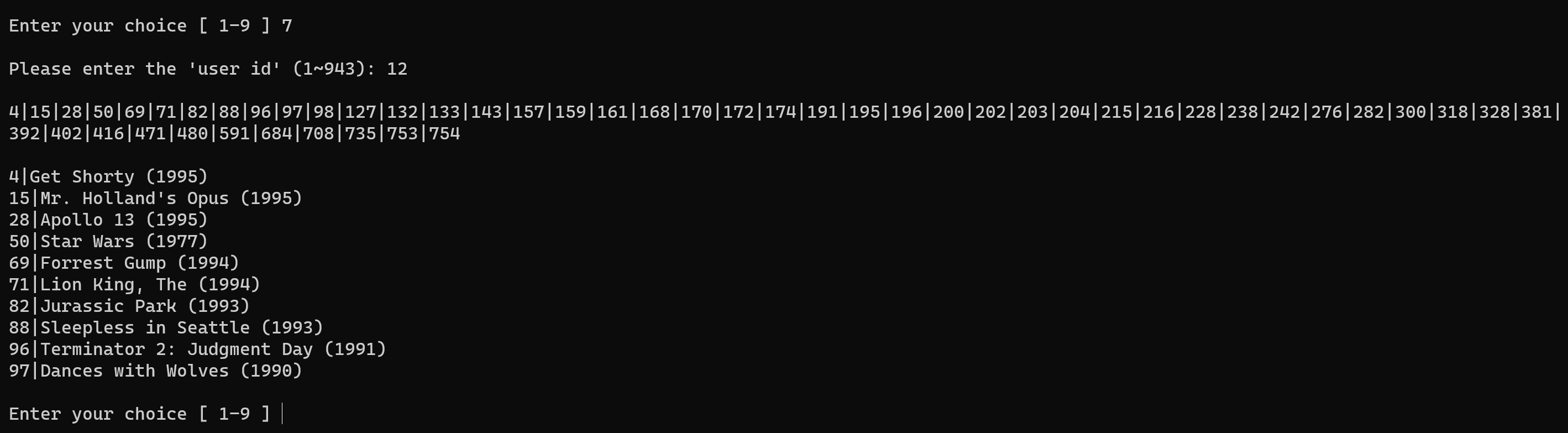
user\_id에 입력받은 user id를 저장 해 줍니다.

파일 실행 시 u.data를 두 번째 매개변수로 두었기 때문에 cat $2으로 u.data의 파일을 열어줍니다.

입력받은 user\_id를 awk 내부 변수로 사용하기 위해 awk -v 옵션으로 변수를 저장해 줍니다. user가 평가한 영화의 movie id를 구하기 위해 user id에 해당하는 1번째 열이 입력한 user\_id와 동일하다면 2번째 열의 movie id를 출력합니다. 이 때 출력 형식을 맞추기 위해 sed 명령어로 마지막 “|”를 지워주었습니다.

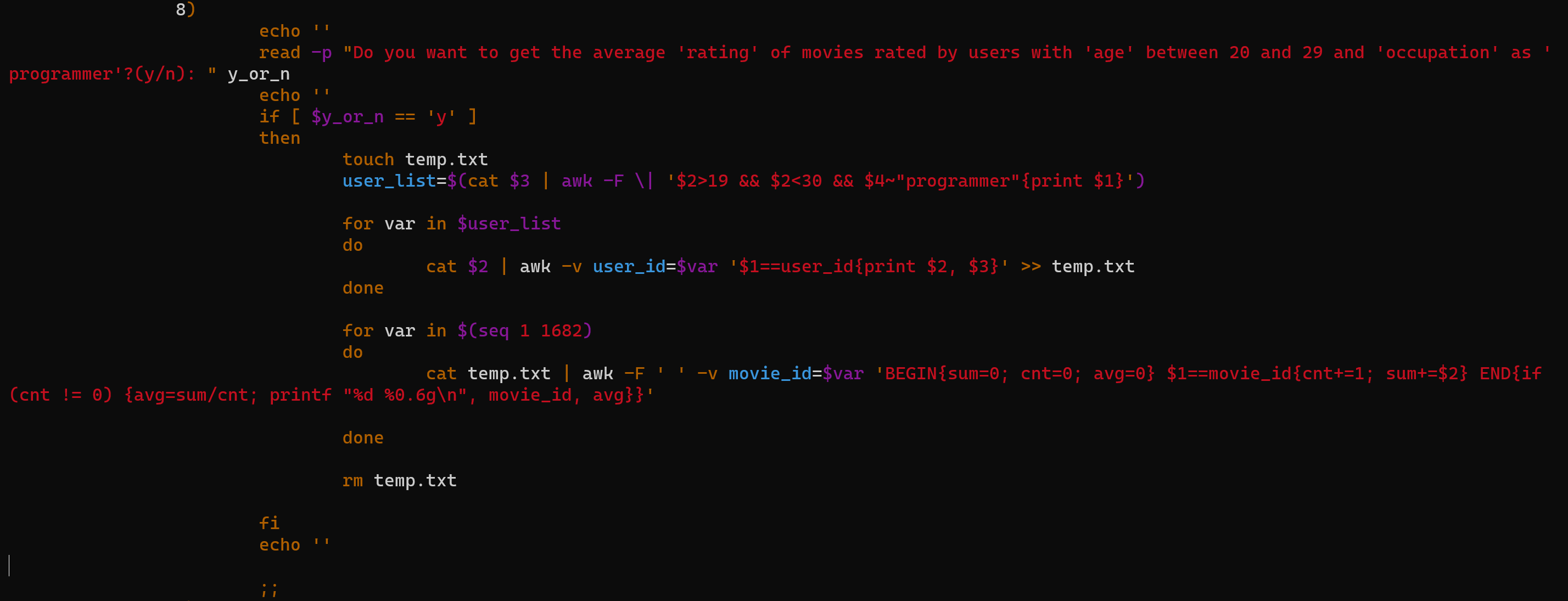
movie id를 통해 영화의 정보를 가져오기 위해선 u.item에서 탐색을 해 주어야 합니다. sort -k 2 -n 명령어로 movie id 기준으로 오름차순 정렬 후 10개의 일치하는 행을 뽑기 위해 awk 명령어를 사용합니다. u.data에서 user\_id로 탐색해 뽑은 10개의 movie id를 list 변수에 저장하고 반복문을 통해 movie id를 var 변수로 하나씩 꺼냅니다.

var의 movie id를 awk -v, 내부 변수 comp에 저장하여 comp와 u.item의 첫 번째 열이 일치하는 행을 출력 형식(movie id|movie title)에 맞춰 출력합니다.



실행 결과입니다.

1. **Get the average 'rating' of movies rated by users with 'age' between 20 and 29 and 'occupation' as 'programmer'**



y\_or\_n에 입력받은 값을 저장 해 줍니다.

if문을 통해 y를 입력받은 경우에만 처리를 해 주었습니다.

파일 실행 시 u.user를 세 번째 매개변수로 두었기 때문에 cat $3으로 u.user의 파일을 열어줍니다.

사용자 정보를 토대로 조건에 부합하는 사용자가 평가한 영화의 평균을 구해야 합니다. 따라서 u.user에서 사용자를 찾은 뒤 u.data에서 해당 사용자가 평가한 점수의 평균을 출력합니다.

u.user에서 주어진 조건에 맞는 20~29세 사이의 프로그래머 사용자를 탐색하기 위해 awk 명령어를 사용합니다. -F \| 옵션을 통해 구분자 “|”로 지정해 주어 age에 해당하는 2번 째 열에 대한 조건, occupation에 해당하는 4번 째 열에 대한 조건에 부합하면 user id에 해당하는 첫 번째 열의 값을 user\_list 변수에 저장합니다.

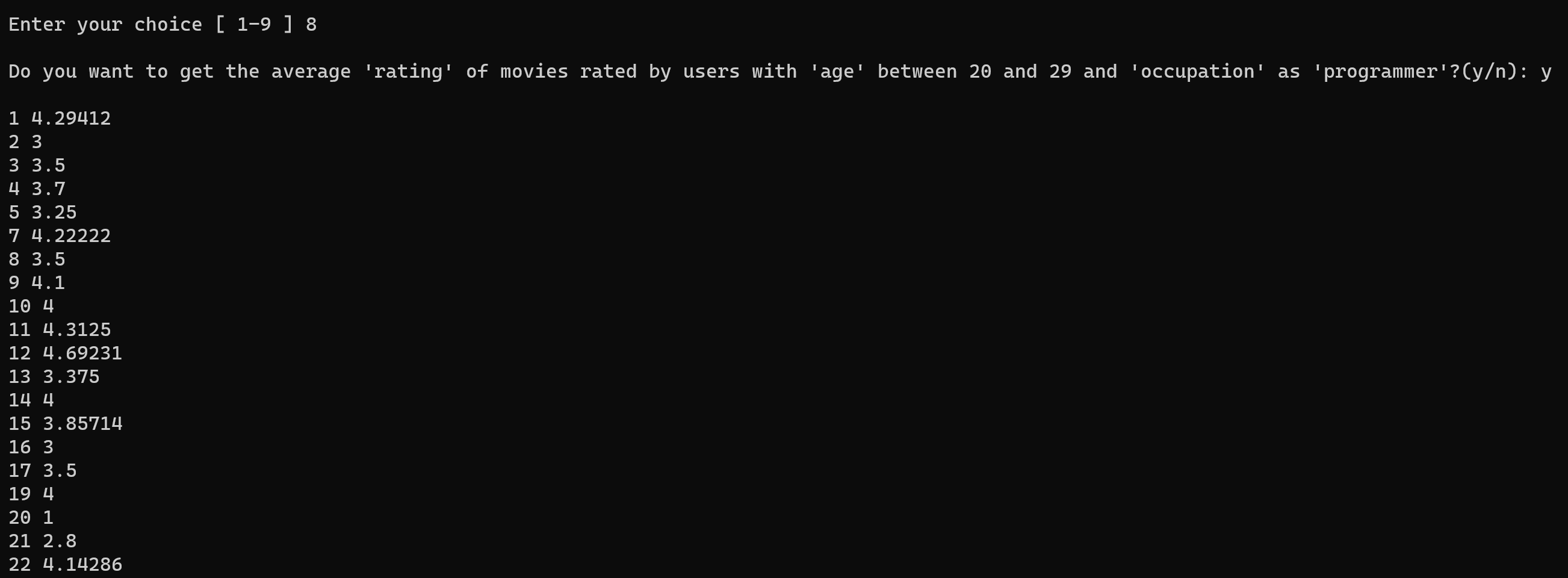
user\_list에 저장 된 user id를 추출하기 위해 반복문을 사용, 매 반복마다 user id를 var에 저장합니다. user id가 u.data의 1번째 열과 동일하다면 해당 사용자가 평가한 영화의 movie id (2번 째 열), rating (3번 째 열)을 임시 파일 temp.txt에 저장합니다.

movie id는 1~1682까지 존재하므로 반복문은 1부터 1682까지 진행, awk 명령어로 평균을 구해줍니다.

-F ‘ ‘ 로 띄어쓰기 한 칸을 구분자로 지정해 주어 열을 구분합니다. 평균을 구해주기 위한 총합(sum), 개수(cnt), 평균(average) 변수를 BEGIN을 통해 초기화 합니다. 1번 째 열에 movie id, 2번 째 열에 rating 정보가 저장되어 있으므로 일치하는 movie id에 대해 rating의 총합, 개수를 더해줍니다. 만일 평가된 영화가 없어 cnt가 0일 경우 출력하지 않고 cnt가 0이 아닌 경우에 대해서만 평균을 출력합니다.

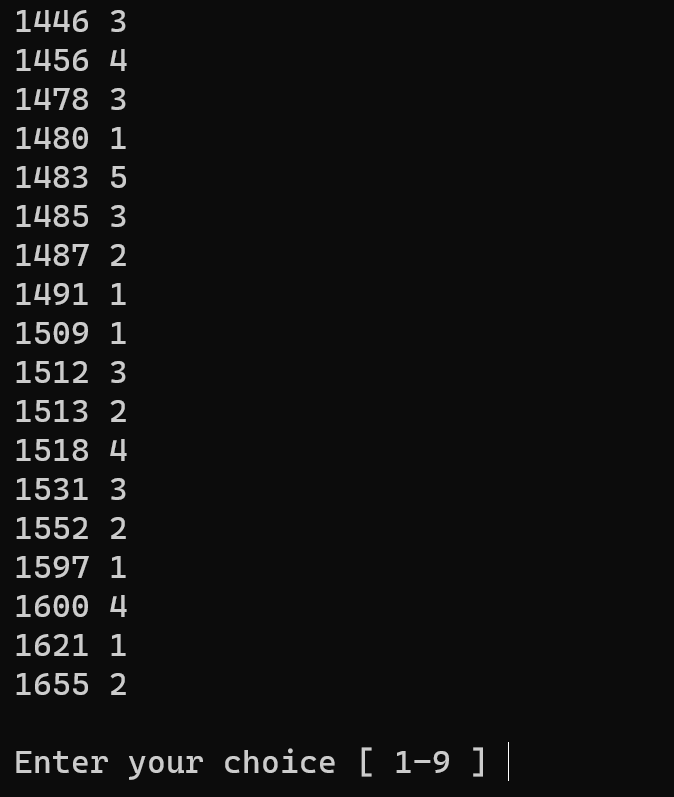
출력은 출력 형식 (movie id average)에 맞춰, average는 소수 6번째 자리에서 반올림 하여 출력 해 줍니다.

사용하고 난 뒤 남은 temp.txt 파일은 rm 명령어로 삭제 해 줍니다.

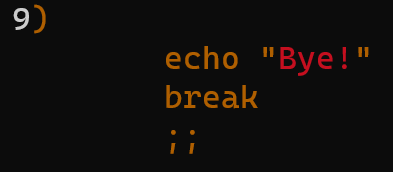


실행 결과입니다.

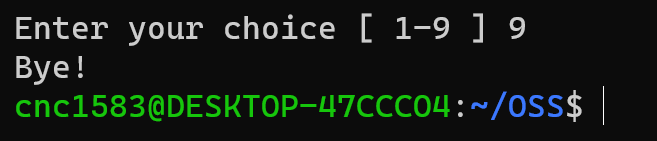
행이 많아 중략하겠습니다.



1. **Exit**



Bye!를 출력하고 until 반복문을 탈출합니다.



실행 결과입니다.