**README**

**1. 개요**

이 보고서는 proj1\_12181613\_parkchanjun.sh 스크립트를 사용하는 사용자를 위한 설명이다. 사용한 데이터는 IMDb에서 가져온 영화, 평가, 유저 데이터로 각각 u.item, u.data, u.user로 사용할 예정이다. 해당 데이터들은 실행파일(.sh)과 같은 폴더에 위치해야 한다.

**2. 기능**

이 스크립트는 8가지 기능을 담당한다. (9번은 종료)

1. 영화 ID값에 해당되는 영화 데이터를 출력하기
2. 액션 장르인 영화 데이터를 출력하기
3. 영화 ID값에 해당되는 영화 평점 평균 출력하기
4. IMDb URL를 삭제한 영화 데이터를 출력하기
5. 유저 데이터를 출력하기
6. 출시 날짜 형식 바꿔서 출력하기
7. 유저 ID값에 해당되는 유저가 평가한 영화 데이터 출력하기
8. 20대 프로그래머 유저가 평가한 영화 평점 평균 출력하기
9. 스크립트 종료

**3. 실행 방법**

이 스크립트는 영화 Data, 평가 Data 그리고 유저 Data인 u.item, u.data, u.user를 모두 사용하기 때문에 모든 기능들을 정상적으로 실행하기 위해서는 다음과 같이 실행을 해주어야한다.

**./proj1\_12181613\_parkchanjun.sh u.item u.data u.user**

**4. 레이아웃**

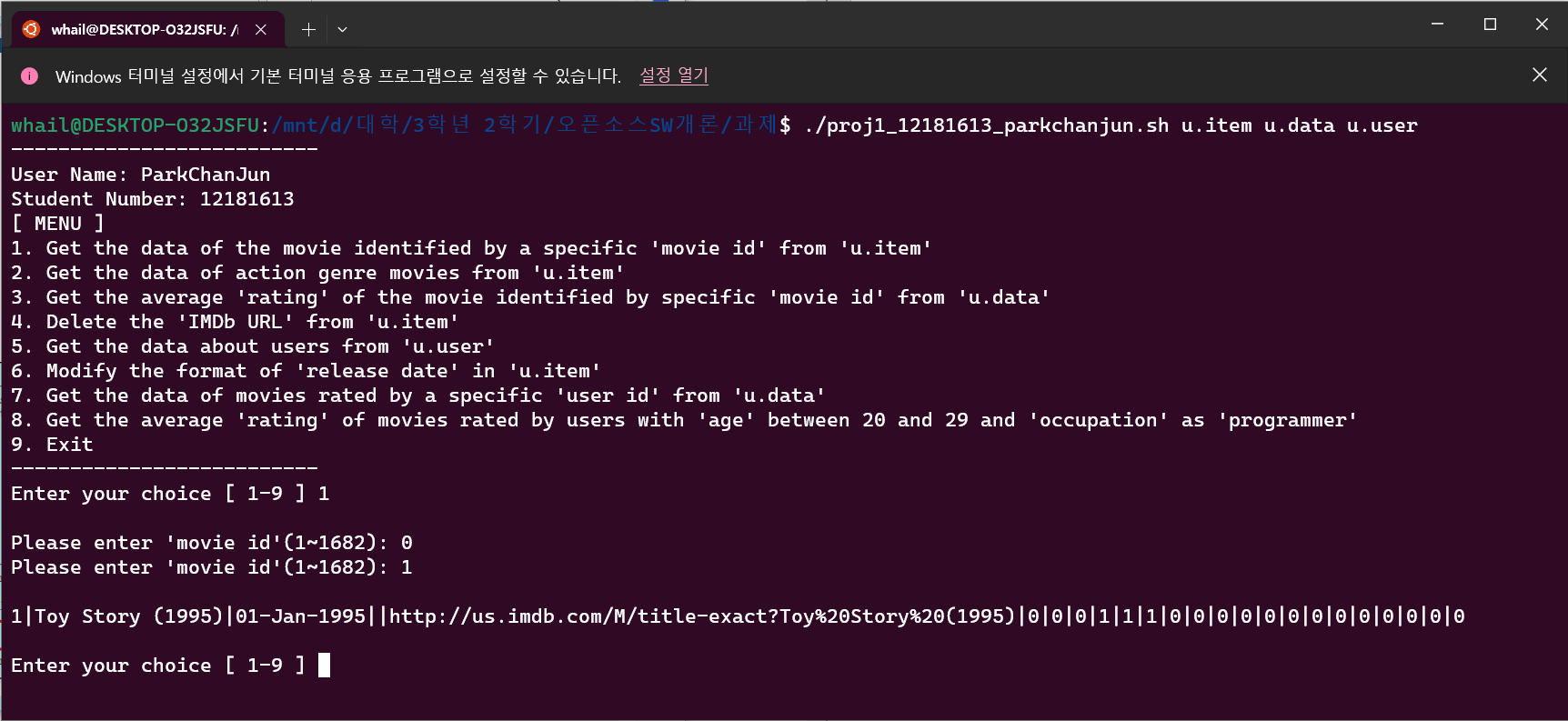
텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

위 그림은 실행했을 때의 초기 메뉴화면이다. 사용자가 원하는 기능이 적혀 있는 해당 번호를 입력 후 Enter를 누르면 해당 기능을 실시간으로 사용할 수 있다. 종료에 해당되는 9번을 선택할 때까지 계속 사용할 수 있다.

**5. 각 기능에 대한 설명**

**(1) 영화 ID값에 해당되는 영화 데이터를 출력하기**

****

이 기능은 값을 입력하면 입력 값에 해당되는 Movie ID의 정보를 출력한다. Movie ID는 1부터 1682까지 있으며, 해당 값에서 벗어나면 다시 입력하도록 한다.

**(2) 액션 장르인 영화 데이터를 출력하기**

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

2번 기능은 액션 장르인 영화 데이터를 Movie ID를 기준으로 오름차순으로 Movie ID와 제목을 출력한다. 데이터는 10개 출력을 한다. 2번을 입력하고 y를 입력해야 실행되며, 만약 n을 입력 시 메뉴 입력으로 돌아온다.

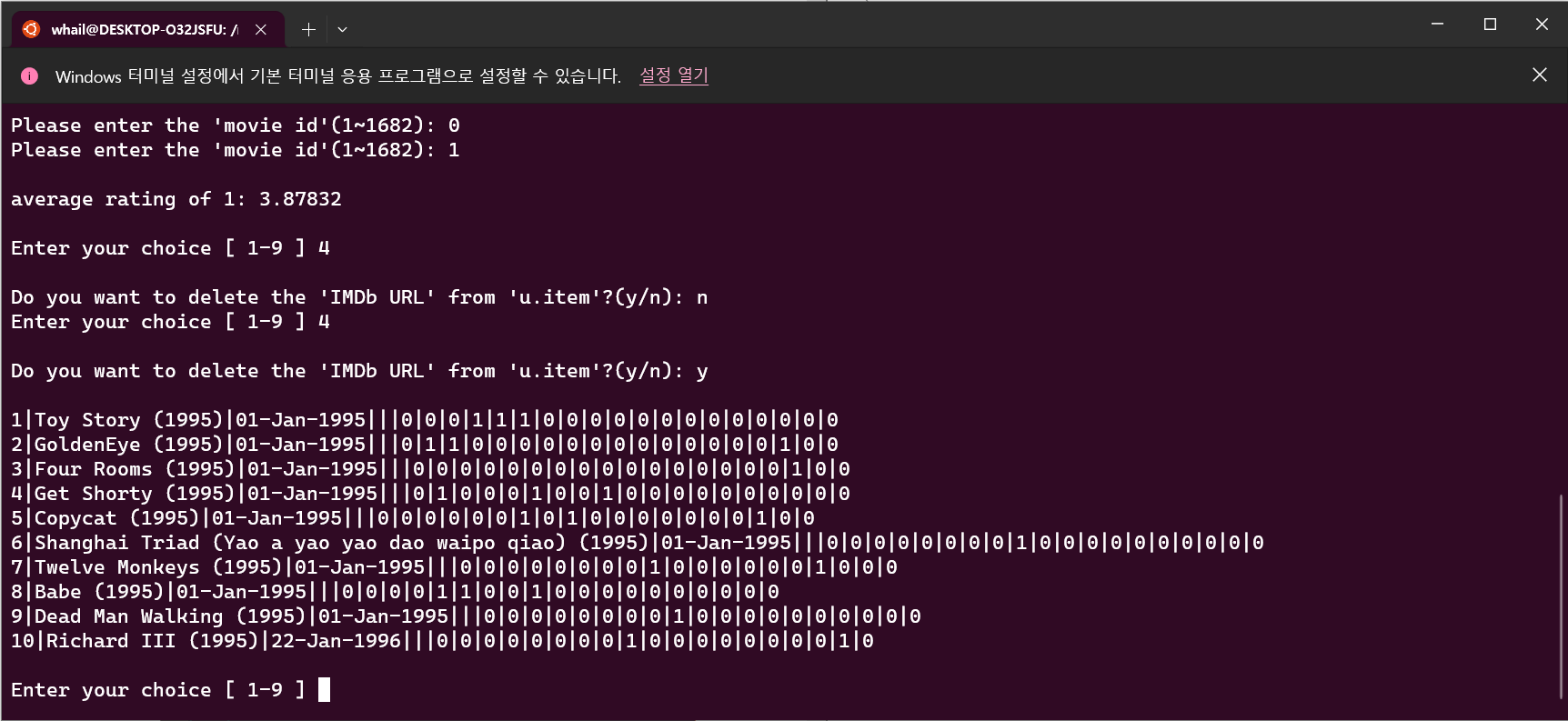
**(3) 영화 ID값에 해당되는 영화 평점 평균 출력하기**

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

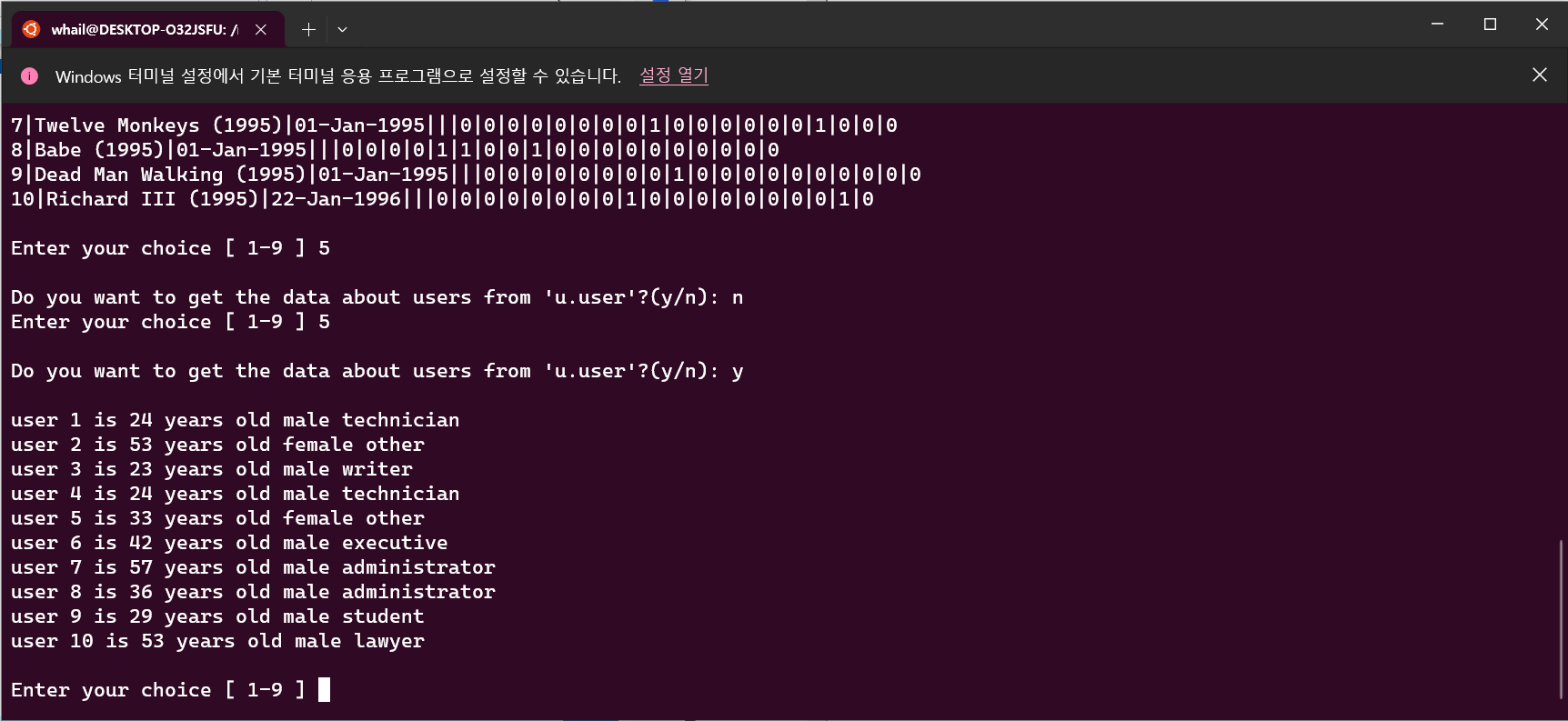
영화 ID값을 입력하면 해당 영화가 받은 평점 평균값을 출력한다. 평점 평균값은 소수점 5자리까지 출력하며, 6자리에서 반올림한다. 1번 기능과 마찬가지로 ID값 범위를 벗어나면 재입력이 가능하다.

**(4) IMDb URL를 삭제한 영화 데이터를 출력하기**



영화 Data에서 IMDb URL를 부분만 삭제하여 출력한다. 총 10개만 출력한다. 앞 기능과 같이 n을 선택 시 다시 메뉴로 돌아온다.

**(5) 유저 데이터를 출력하기**



유저에 대한 정보를 출력한다. user “유저 아이디” is “유저 나이” years old “성별” “직업” 형식으로 출력한다. 총 10개 출력하며, n을 선택 시 다시 메뉴입력으로 돌아온다.

**(6) 출시 날짜 형식 바꿔서 출력하기**

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

출시일의 기존 출력형식을 다르게 바꾼다. Ex) 01-Jan-1995 -> 19950101. 뒤에서부터 10개만 출력한다. n을 선택 시 메뉴입력으로 돌아온다.

**(7) 유저 ID값에 해당되는 유저가 평가한 영화 데이터 출력하기**

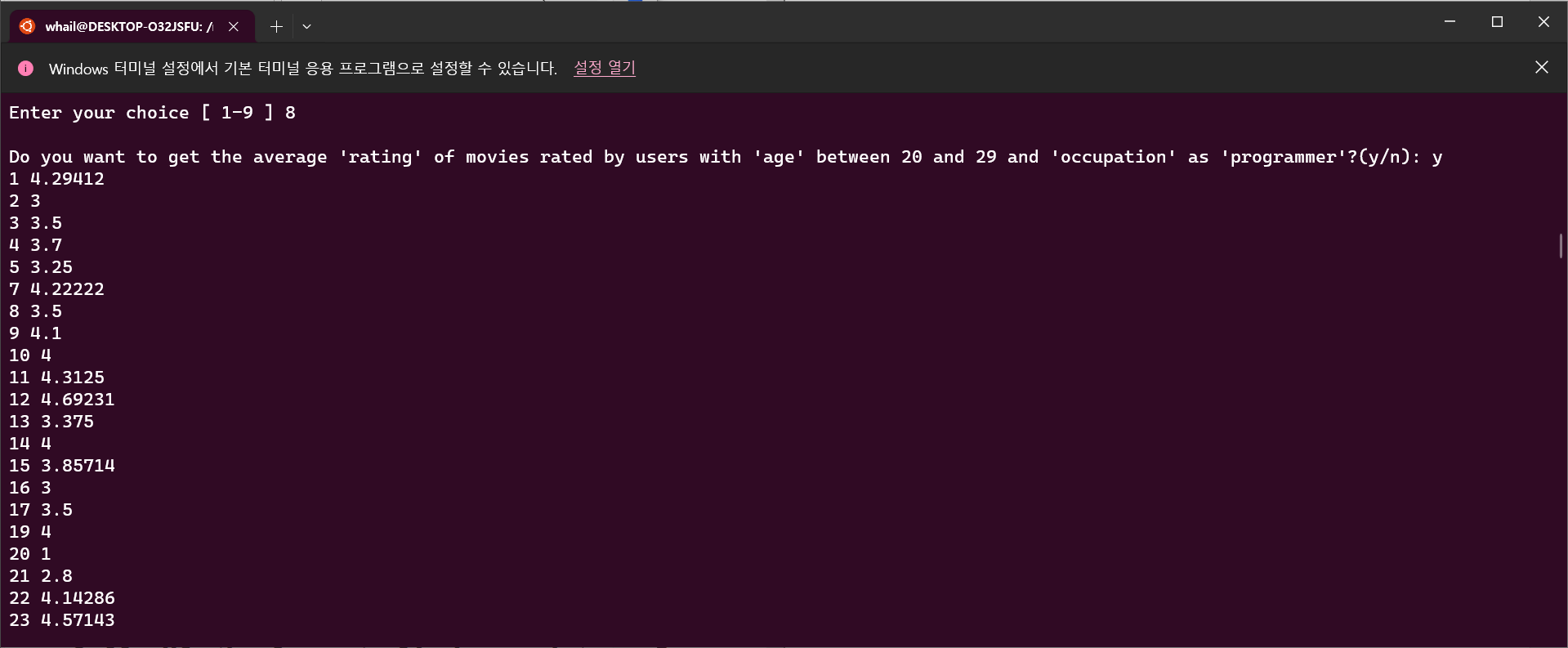
텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

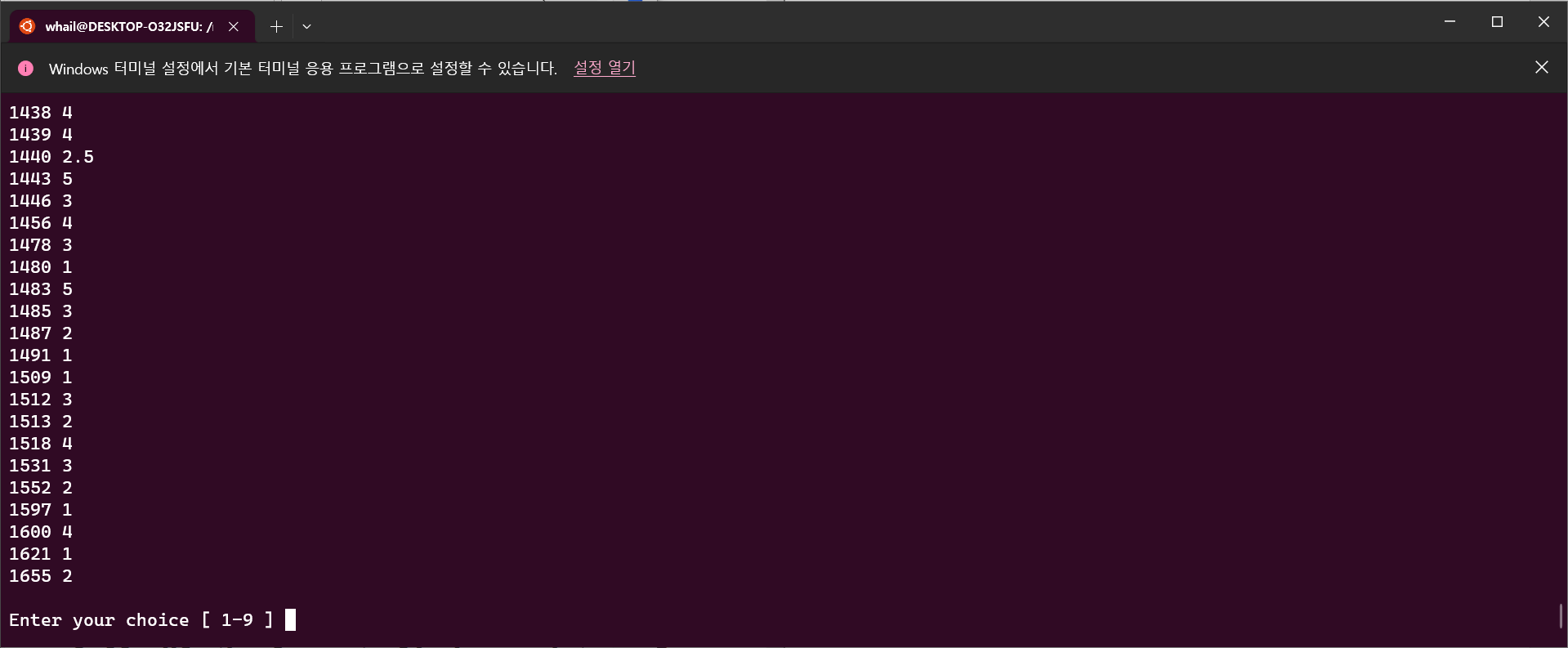
자동 생성된 설명텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

유저 ID값을 입력하면 해당 유저가 평가한 영화의 ID값이 오름차순으로 ‘|’를 구분으로 출력이 되며, 이후 영화 ID값과 제목이 각각 출력된다. 유저 ID는 1부터 943까지 있으며, 값을 벗어나면 다시 입력할 수 있다.

**(8) 20대 프로그래머 유저가 평가한 영화 평점 평균 출력하기**

****

****

20대(나이: 20-29)이고 프로그래머인 유저가 평가한 모든 영화 평점 평균을 출력한다. 평점 평균은 최대 소수점 5자리까지 출력하며, 6자리에서 반올림한다. 만약 소수점을 줄일 수 있다면 최대한 줄여서 출력한다. Ex) 2.000 -> 2   
앞 기능과 같이 n을 입력 시 메뉴입력으로 돌아간다.

**(9) 스크립트 종료**

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

Bye!를 출력하고 스크립트를 종료한다.

**6. 구현 방식**

메뉴판을 모두 출력한 뒤, Loop로 각 메뉴입력 값을 받는다. 입력 시 case로 각 기능을 구분하였고, 함수를 사용하여 모듈화하여 가독성을 높였다.

각 파일을 받고 사용하기위해 지역변수로 item, data, user를 넣었다. 실행시 각각 u.user, u.data, u.user 파일을 받게 된다.

**(1) 영화 ID값에 해당되는 영화 데이터를 출력하기**

|  |
| --- |
| cat $item | sed -n "$findId p" |
| cat $item | awk -v findId=$findId -F '|' '{if($1 == findId) print}' |

해당 코드는 처음에 표 1행과 같이 작성을 하였지만 해당 코드는 스크립트에 들어간 데이터가 순서대로 들어가지 않으면 원하는 데이터가 출력되지 않아 2행과 같이 수정하였다. findId는 read 명령어로 사용자로부터 입력 받은 값이며, awk 명령어로 해당 값을 찾아서 그 행을 출력한다.

**(2) 액션 장르인 영화 데이터를 출력하기**

|  |
| --- |
| cat $item | awk -F '|' '{if($7 == 1) printf("%s %s\n", $1, $2)}'  | head -n 10 |

해당 코드는 u.user파일의 7열(액션장르)이 1이면 1열과 2열, 즉 Movie ID와 제목을 출력한다.

**(3) 영화 ID값에 해당되는 영화 평점 평균 출력하기**

|  |
| --- |
| Total=$(cat $data | awk -v findId="$findId" '{if($2 == findId) sum+=$3} END {print sum}')  cnt=$(cat $data | awk -v findId="$findId" '{if($2 == findId) print $3}' | wc -l)    if [ $cnt == 0 ]  then  echo "No ratings"  else  echo $(printf "%.5f" $(echo "scale=6; $Total / $cnt" | bc))  fi  echo |

awk명령어를 사용하여 영화 ID값에 해당되는 값을 u.data의 2열에서 찾는다. 찾은 행의 3열을 sum에 모두 더하여 Total변수에 저장한다. 또한 찾은 행의 개수를 cnt변수에 저장한다.

이후 cnt값이 0이라면 그 영화는 평가한 사람이 없으므로 No ratings를 출력한다. 값이 존재하면 소수점 6번째까지 가져온 다음 printf로 5번째까지로 형식을 바꾸어 반올림을 한다.

**(4) IMDb URL를 삭제한 영화 데이터를 출력하기**

|  |
| --- |
| cat $item | sed -E 's/http:\/\/us.imdb.com[^|]\*//g' | head -n 10 |

IMDB URL은 모두 <https://www.imdb.com>로 시작하므로 해당 값부터 ‘|’까지 삭제해주면 된다. 따라서 Regex는 [^|]\*이다. 이 값을 앞에서 10개를 출력하기 위해 head -n 10을 파이프라인으로 연결해 써준다.

**(5) 유저 데이터를 출력하기**

|  |
| --- |
| cat $user | awk -F '|' '{printf("user %s is %s years old ", $1, $2); if($3=="M") printf("male %s\n", $4); else printf("female %s\n", $4)}'  | head -n 10 |

awk명령어를 사용해서 user {1열} is {2열} years old male/female{3열로 결정} {4열} 형식으로 출력하였다. 1열은 user ID값, 2열은 유저나이, 3열은 M/F로 되어있지만 조건문을 사용하여 male/female로 출력을 하였다. 4열은 직업이며 모두 u.use파일에서 가져와 출력하였다.

**(6) 출시 날짜 형식 바꿔서 출력하기**

|  |
| --- |
| IFS=$'\n'  for movie in $(cat $item | tail -n 10)  do  dateFormat=$(date -d "$(echo $movie | cut -d\| -f3 | head -n 1)" +%Y%m%d)  echo $movie | sed -E "s/([^|]\*\|[^|]\*\|)[0-9]{2}-[A-Za-z]{3}-[0-9]{4}/\1$dateFormat/g"  done |

for문을 사용하여 u.item의 10개의 행을 가져왔다. cut명령을 사용하여 날짜를 date명령어로 형식을 바꾸어 저장하였고, 그 값을 sed명령어를 사용하여 기존 출시날짜 대신 넣었다. 기존 출시날짜는 ‘|’를 두 번 지나고나서 01-Jan-1994로 나오므로 Regex는 ([^|]\*\|[^|]\*\|)[0-9]{2}-[A-Za-z]{3}-[0-9]{4} 이다. 그리고 앞 내용을 출력 후 변수를 출력하기 위해 \1$dateFormat으로 변경하였다.

**(7) 유저 ID값에 해당되는 유저가 평가한 영화 데이터 출력하기**

|  |
| --- |
| movieSet=$(cat $data | awk -v findId="$findUserId" '{if($1 == findId) print $2}' | sort -n | sed -z -E 's/\n/|/g')  echo ${movieSet:0:-1} |

먼저 특정 유저가 평가한 영화 ID값들을 ‘|’를 기준으로 출력한다. 그러기 위해 유저 ID값으로 평가한 영화 ID값을 모두 가져온다. 그 다음 파이프라인으로 sed명령어를 이어서 newline을 ‘|’로 변경한다. 이 경우 마지막에도 ‘|’가 붙으므로 슬라이스하여 마지막 ‘|’만 제거하여 출력한다.

|  |
| --- |
| IFS=$'|'  for i in $movieSet  do  cat $item | awk -v i="$i" -F '|' '{if($1 == i) printf("%s|%s\n", i, $2)}'  done  echo  IFS=$'\n' |

movieSet 변수를 그대로 사용하기위해 IFS에 $’|’를 입력한다. 이 경우 for문에서 행을 기준으로 값을 가져오지 않고 ‘|’를 기준으로 값을 가져온다. 만약 사용하지 않으려면 위에서 awk 명령어만 가져와 새로운 변수에 저장 후 for문을 사용하면 된다. 다시 본론으로 돌아와서 for문의 i에는 유저가 평가한 영화 ID값이 반복하여 들어오고 이 값을 u.item파일의 Movie ID값과 비교하여 같으면 ‘|’를 기준으로 ID값과 제목을 출력한다.

**(8) 20대 프로그래머 유저가 평가한 영화 평점 평균 출력하기**

|  |
| --- |
| prog20To29s=""  for userId in $(cat $user | awk -v occupation="programmer" -F '|' '{if($4 == occupation && $2 < 30 && $2 >= 20) print $1}')  do  for movieIdWithRating in $(cat $data | awk -v userId="$userId" '{if($1 == userId) printf("%s,%s\n", $2, $3)}')  do  prog20To29s+=${movieIdWithRating}$'\n'  done  done    movieId=0  while true  do  movieId=$(($movieId+1))  if [ $movieId -gt $(cat $item | wc -l) ] # $(cat $item | wc -l)  then  break  fi  Total=$(echo "$prog20To29s" | awk -v movieId="$movieId" -F ',' '{if($1 == movieId) sum+=$2} END {print sum}')  cnt=$(echo "$prog20To29s" | awk -v movieId="$movieId" -F ',' '{if($1 == movieId) print $2}' | wc -l)  if [ $cnt != 0 ]  then  echo -n "$movieId "  echo $(printf "%.5f" $(echo "scale=6; $Total / $cnt" | bc)) | sed -e 's/[0]\*$//g' | sed -e 's/\.$//g'  fi  done  echo |

먼저 20대 프로그래머 유저가 평가한 영화 ID값을 찾아 변수에 모두 저장하기 위해 u.user에서 20대이고, 직업이 프로그래머인 유저들을 awk명령어를 이용해 가져온다. 해당 유저들의 ID를 for문 userId에 반복하여 입력하고 각 유저들이 평가한 영화의 ID와 평점을 가져오기 위해 awk명령어를 이용하여 u.data에서부터 모든 데이터를 movieIdWithRating변수에 저장한다.   
 movieId 개수가 변할 수 있으므로 wc -l 명령어로 u.item의 개수를 찾고, 해당 개수만큼 while문을 반복한다. 그 후 Movie ID 1부터 1682까지 awk명령어 즉 3번 문제에서 사용한 코드를 활용하여 각 Moive ID에 20대 프로그래머 유저가 평가한 평균 점수를 출력한다.   
 이때 다른 점은 평가가 안되어 있으면 출력하지 않아서 조건문을 달리하였고, 출력시 소수점을 줄일 수 있으면 줄여야하므로 sed를 이용하여 0을 뒤에서부터 지웠으며 만약 정수로 출력할 수 있으면 ‘.’까지 지웠다.

**(9) 스크립트 종료**

|  |
| --- |
| if [ $choice == 9 ]  then  break  fi |

해당 코드는 while문안에 넣었으며 choice변수 즉 메뉴입력시 9이면 “Bye!”를 출력한 뒤, break하여 while문을 빠져나와 프로그램을 종료한다.  
 “Bye!”출력은 case문에 넣었다.