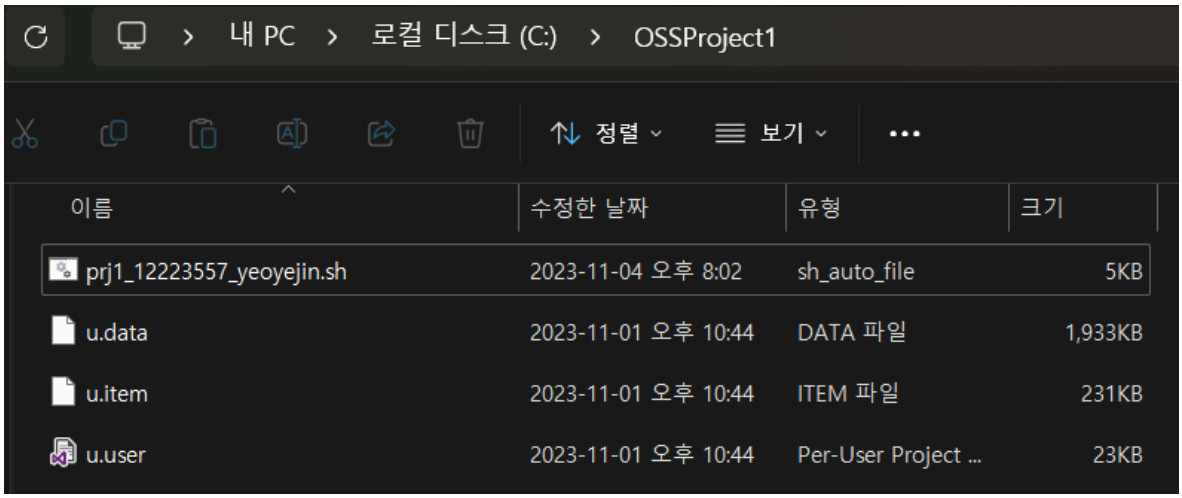
OSS Project1 README

인공지능공학과 12223557 여예진

script file 작성 및 실행 과정

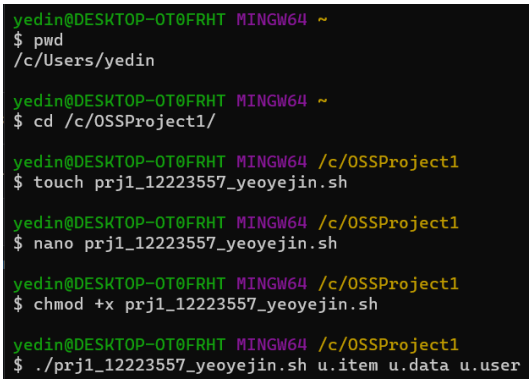


* 강의의 리눅스 개발환경을 재설치 하는 과정에서 오류가 생겨 git bash로 스크립트 작성, 파일 생성 및 실행을 하였다.

1. 로컬 디스크에 OSSProject1 폴더 생성 및 데이터 파일 저장

로컬 디스크 (C:)에 OSSProject1 폴더를 생성하여 u.data, u.item, u.user 세 데이터 파일을 저장한다.

2. 스크립트 파일 생성 및 실행



item file(ITEM_F), data file(DATA_F), user file(USER_F) 세 input file을 인자로 전달받는 스크립트를 작성한 후, 우측과 같이 작성된 명령어들을 통해 '/c/OSSProject1/' 디렉토리에 'prj1_12223557_yeoyejin.sh' 스크립트 파일을 생성하여 스크립트를 세 개의 인자와 함께 실행한다. 세부 사항은 아래와 같다.

touch 명령어로 'prj1_12223557_yeoyejin.sh' 이름으로 새 파일을 현재 디렉토리(OSSProject1)에 생성하고, nano 명령어를 통해 프로그램 코드를 파일에 작성했다. chmod +x 를 통해 실행하고자 하는 파일에 실행 권한을 부여하여 쉘 스크립트로 실행할 수 있도록 설정했다.

세부 구현 내용

1. 스크립트 인자로 전달된 파일 경로 할당

u.item, u.data, u.user 세 개의 input file을 test.sh의 인자로 전달하기 위해 ITEM_F, DATA_F, USER_F 변수에 각 인자들을 할당한다.

```
ITEM_F=$1
DATA_F=$2
USER_F=$3
```

2. 메뉴 출력

```
echo "------------------------"
echo "User Name: Yeo Yejin"
echo "Student Number: 12223557"
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 date' 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 "------------------------"
```

3. 메뉴 선택에 따른 실행 반복 처리

사용자로부터 메뉴 선택을 계속해서 받아 해당 선택에 따른 작업을 수행한다. "9. Exit"에 해당하는 메뉴 선택을 하면 반복을 종료한다.

4. 메뉴1 : 특정 영화 데이터 출력

```
1)
    echo
    echo -n "Please enter the 'movie id'(1~1682): "
    read movie_id
    echo
    awk -F"|" '$1 == "'$movie_id'" { print $0 }' $ITEM_F
    echo
    ;;
```

사용자의 입력을 movie_id 변수에 저장하고, awk 명령어를 사용하여 u.item 파일($ITEM_F)에서 입력받은 movie id에 해당하는 데이터를 찾아 출력한다. 필드 구분자를 "|"로 설정하고 awk 스크립트 내에서, 첫 번째 필드($1)가 사용자가 입력한 movie_id와 일치하는지 확인하여 일치할 시 해당 행($0)을 출력한다. 변수를 awk 스크립트 내에 넣기 위해 '$movie_id' 형식으로 쉘 변수를 문자열 안에 삽입했다.

5. 메뉴2 : action 장르 영화 데이터 출력

```
2)
    echo
    echo -n "Do you want to get the data of ‘action’ genre movies from 'u.item’?(y/n): "
    read response
    echo
    if [ "$response" == "y" ]; then
        awk -F"|" '$7 == "1" {print $1, $2}' $ITEM_F | sort -n | head -10
        echo
    fi
    ;;
```

액션 장르 영화 데이터를 가져오고 싶은지에 대한 여부 메시지를 출력하고 사용자의 응답을 response 변수에 저장한다. 사용자의 응답이 ‘y’인 경우에는 awk 명령어를 사용하여 u.item 파일($ITEM_F)의 7번째 필드가 ‘1’인 영화로 표시된 부분을 필터링하고, sort -n, head -10 명령어를 movie id 기준 오름차순으로 10개의 결과만 주어진 형식에 맞게 출력했다.

6. 메뉴3 : 특정 영화의 평균 평점 출력

```
3)
    echo
    echo -n "Please enter the 'movie id’(1~1682): "
    read movie_id
    echo
    echo -n "average rating of $movie_id: "
    awk -F"\t" '$2 == "'$movie_id'" { sum+=$3; count++ } END { if (count > 0) printf "%.5f", sum/count; else print 0 }' $DATA_F
    echo
    echo
    ;;
```

사용자의 입력을 movie_id 변수에 저장하고, awk 명령어를 사용하여 입력받은 movie id에 해당하는 평균 평점을 계산한다. 필드 구분자를 u.data 파일($DATA_F)에 맞게 탭으로 설정하고 2번째 필드(movie id)가 사용자로부터 입력받은 movie_id와 일치하는지 확인한 후, 평점의 누적합과 평점 받은 횟수를 계산하여 각각 sum, count 변수에 저장한다. 모든 입력을 처리한 후 END를 통해 평균 평점을 계산하고, count 변수의 값이 0인 경우에 대한 validation을 처리했다.

7. 메뉴4 : IMDb URL 부분 삭제

```
4)
    echo
    echo -n "Do you want to delete the ‘IMDb URL’ from ‘u.item’?(y/n): "
    read response
    echo
    if [ "$response" == "y" ]; then
        awk -F"|" 'BEGIN {OFS="|"} { $5=""; print $0 }' $ITEM_F | head -10
        echo
    fi
    ;;
```

사용자에게 'u.item' 파일에서 IMDb URL을 삭제하고 싶은지 여부를 입력 받고 ‘y’인 경우 awk 명령어를 이용해 5번째 필드(IMDb URL)를 빈 문자열로 바꾼 후 해당 줄을 10개만 출력한다.

8. 메뉴5 : user 데이터 출력

```
5)
    echo
    echo -n "Do you want to get the data about users from ‘u.user’?(y/n): "
    read response
    echo
    if [ "$response" == "y" ]; then
        head -10 $USER_F | awk -F"|" '{ gender = ($3 == "M") ? "male" : "female"; print "user", $1, "is", $2, "years old", gender, $4 }'
        echo
    fi
    ;;
```

사용자의 응답이 ‘y’인 경우 파일의 처음 10줄을 awk 명령어의 입력 데이터로 하여, 현재 줄의 세 번째 필드(성별) 기반으로 gender 변수를 설정한다. M은 male로 그렇지 않으면 female로 바꿔서 출력한다.

9. 메뉴6 : release data 형식 변경

```
6)
    echo
    echo -n "Do you want to Modify the format of ‘release data’ in ‘u.item’?(y/n): "
    read response
    echo
    if [ "$response" == "y" ]; then
        tail -n 10 $ITEM_F | awk -F"|" '{
            split($3, date, "-")
            newDate = sprintf("%4d%02d%02d", date[3], (index("JanFebMarAprMayJunJulAugSepOctNovDec", date[2]) + 2) / 3, date[1])
            $3=newDate;
            print
        }' OFS="|"
        echo
    fi
    ;;
```

파일의 마지막 10줄을 awk 명령어의 입력 데이터로 설정한다. “|”를 구분자로 설정하고 세 번째 필드($3)에서 'release date'를 '-'로 분할하여 date 배열에 저장한다. sprintf 함수를 사용하여 새로운 날짜 형식을 만드는데, 월 형식 변환에 대한 자세한 로직은 아래와 같다.

예를 들어, "Jan"의 경우 index 값은 1이다. 1에 2를 더하면 3이 되며, 이를 3으로 나누면 1이 된다. "Feb"의 경우 index 값은 4이다. 4에 2를 더하면 6이 되며, 이를 3으로 나누면 2가 된다. 이러한 연산을 위해 data[2] 변수에 저장된 월에 해당하는 index 값에 2를 더하고 3으로 나누어 숫자 형식으로 변환했다.

10. 메뉴7 : 특정 user에 대한 영화 출력

```
7)
    echo
    echo -n "Please enter the ‘user id’(1~943): "
    read user_id
    echo
    awk -v id=$user_id -F"\t" '$1 == id {print $2}' $DATA_F | sort -n | tr '\n' '|' | sed 's/|$//'
    echo
    echo

    awk -v id=$user_id -F"\t" '$1 == id {print $2}' $DATA_F | sort -n | head -10 | while read movie_id; do
        awk -v m_id=$movie_id -F"|" '$1 == m_id { print m_id "|" $2 }' $ITEM_F
    done
    echo
    ;;
```

특정 사용자가 평가한 movie id 출력은 아래와 같다.

awk를 사용하여 $DATA_F 파일에서 입력받은 user_id와 일치하는 사용자의 모든 movie id를 가져와 오름차순으로 정렬하고 tr '₩n' '|'을 사용하여 각 movie id 사이에 '|' 문자를 삽입하여 하나의 문자열로 만든다. 마지막에 있는 '|' 문자는 sed 's/|$//'을 사용하여 제거했다.

상위 10개의 movie id와 title 출력은 아래와 같다.

awk 명령어로 user_id와 일치하는 첫 번째 필드($1)를 가진 레코드를 찾고, 해당 레코드의 두 번째 필드($2)를 오름차순으로 정렬한 후 상위 10개만 선택한다. while 명령어를 통해 파일에서 해당 movie_id와 일치하는 영화 제목을 찾아 출력한다.

11. 메뉴8 : 특정 조건(20대 프로그래머)을 만족하는 사용자들의 평균 평점 계산

```
8)
    echo
    echo -n "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): "
    read response
    echo

    if [[ "$response" == "y" ]]; then
        awk -F'|' '$2 >= 20 && $2 <= 29 && $4 == "programmer" {print $1}' $USER_F |
        awk -F'\t' -v user_file=- 'BEGIN { while ((getline uid < user_file) > 0) user_arr[uid]=1; }
            { if ($1 in user_arr) { sum[$2] += $3; count[$2]++; } }
            END {
                for (movie in sum) {
                    avg = sum[movie]/count[movie];
                    if (avg == int(avg))
                        printf "%s %d\n", movie, avg;
                    else
                        printf "%s %.5f\n", movie, avg;
                }
            }' $DATA_F | sort -k1,1n | awk '{ printf "%s ", $1; if ($2 ~ /\.0+$/) { print int($2); } else { gsub(/0+$/, "", $2); print $2; } }'
        echo
    fi
    ;;
```

awk 명령어로 $USER_F 파일에서 나이가 20에서 29세 사이이며 직업이 'programmer'인 user id를 다음 awk 명령어의 입력 데이터로 하여 배열 user_arr에 저장한다. if 명령어를 통해 $DATA_F 파일에서 해당 사용자의 영화 평점을 가져와 각 영화에 대한 평점 누적합과 평가 횟수를 계산하여 각각 sum, count에 저장한다. END 부분에서는 각 영화의 평균 평점을 계산하고 오름차순으로 정렬하여 출력한다. 이때 소수점이 .0으로 끝나면 정수로 출력하고, 그렇지 않으면 불필요한 0을 제거하고 출력하도록 설정하였다.

12. 메뉴9 : 프로그램 종료

사용자의 입력이 9인 경우 터미널 화면에 "Bye!"라는 메시지를 출력하고 현재 실행 중인 스크립트를 종료한다. 이때 1-9 사이의 정수가 아닌 입력을 받았을 경우에 대한 validation을 추가로 설정했다.

```
9)
    echo "Bye!"
    exit 0
    ;;

*)
    echo "Invalid number."
    echo
    ;;
```