HOW TO USE

이용방법

리눅스 `bash` 쉘에서 동작합니다. 리눅스 환경을 우선 준비해둬야 합니다.

1. 필수 요건 설치

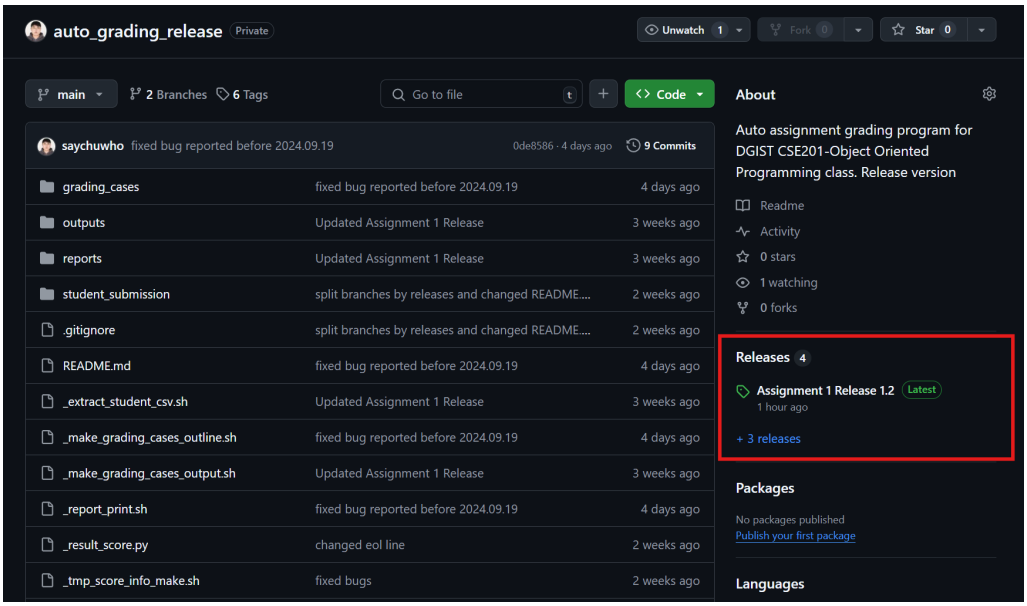
`git`, `zip`, `unzip`, `dos2unix`, `perl`을 설치합니다.

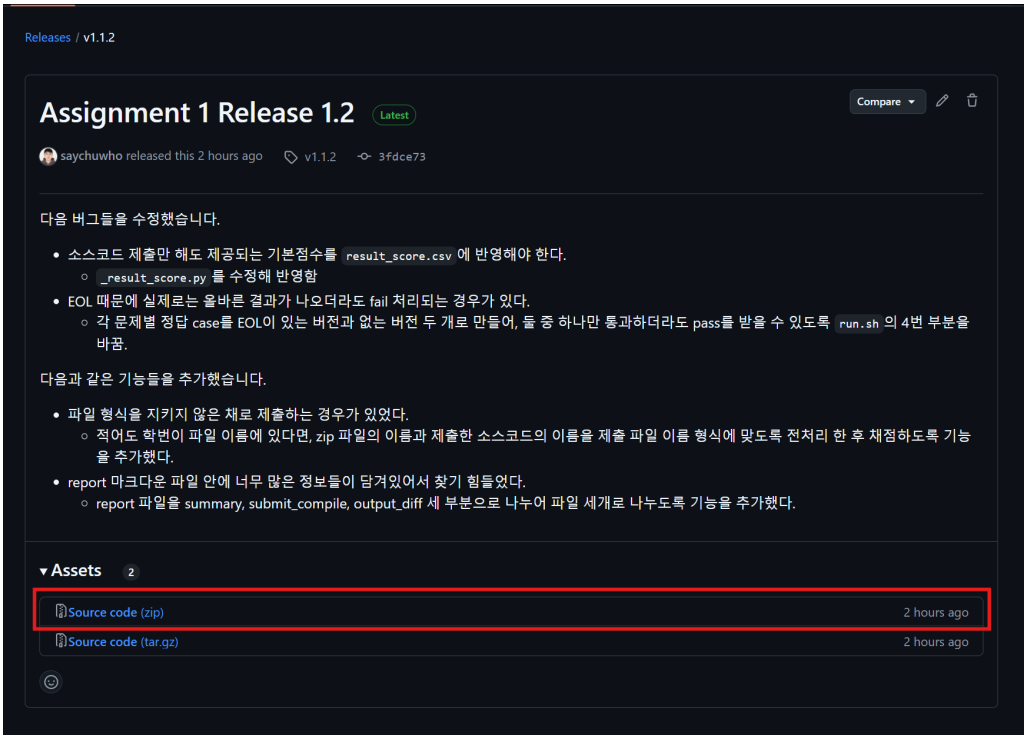
```
sudo apt install git zip unzip dos2unix perl
```

2. 프로그램 설치

각 assignment 버전에 맞는 최신 release zip 파일을 다운받아 압축해제 합니다.

예를 들어, assignment 1 채점 프로그램을 얻는다고 하면, assignment 1이 붙은 release 중 가장 최신 버전을 다운받으면 됩니다.

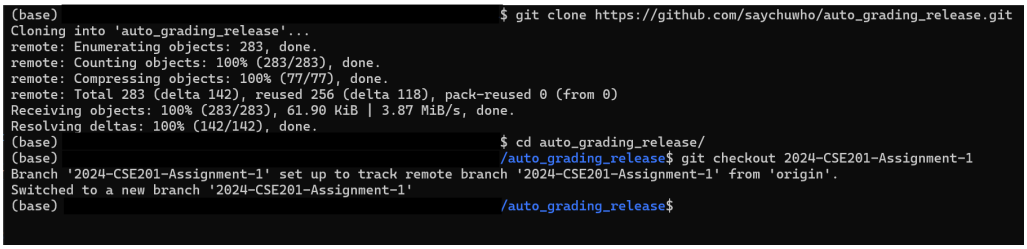




또는, 이 repository를 clone 한 후, assignment 버전에 해당하는 branch로 checkout합니다.

예를 들어, assignment 1 채점 프로그램을 얻는다고 하면, 이 repository를 clone한 후, assignment 1에 해당하는 branch인 `2024-CSE201-Assignment-1`로 checkout하면 됩니다.

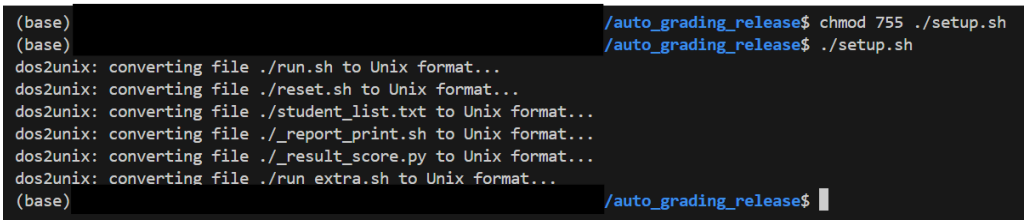
```
git clone https://github.com/saychuwho/auto_grading_release.git
cd auto_grading_release/
git checkout 2024-CSE201-Assignment-1
```



3. 스크립트에 실행 권한 주기

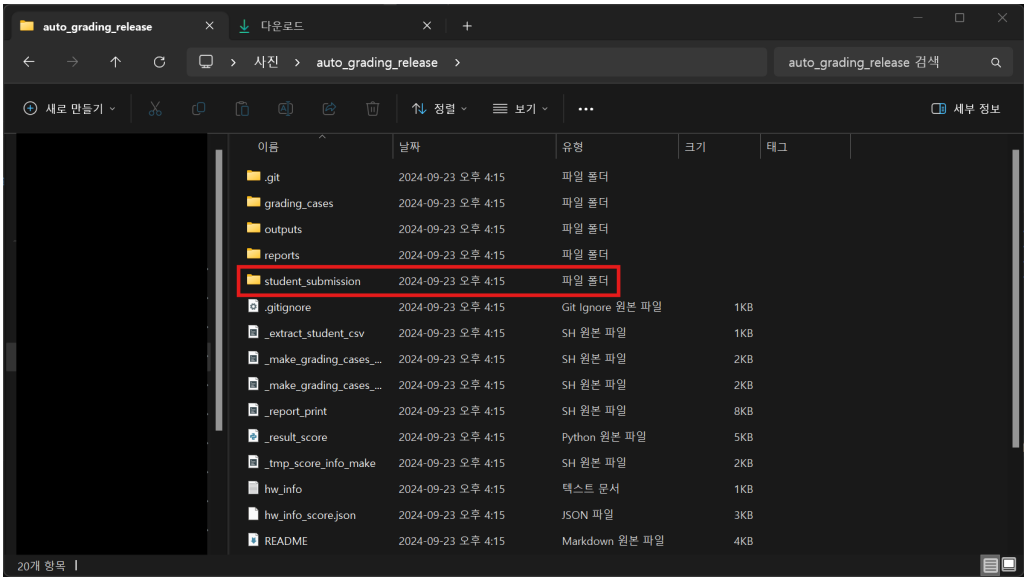
다음 명령어들을 실행합니다.

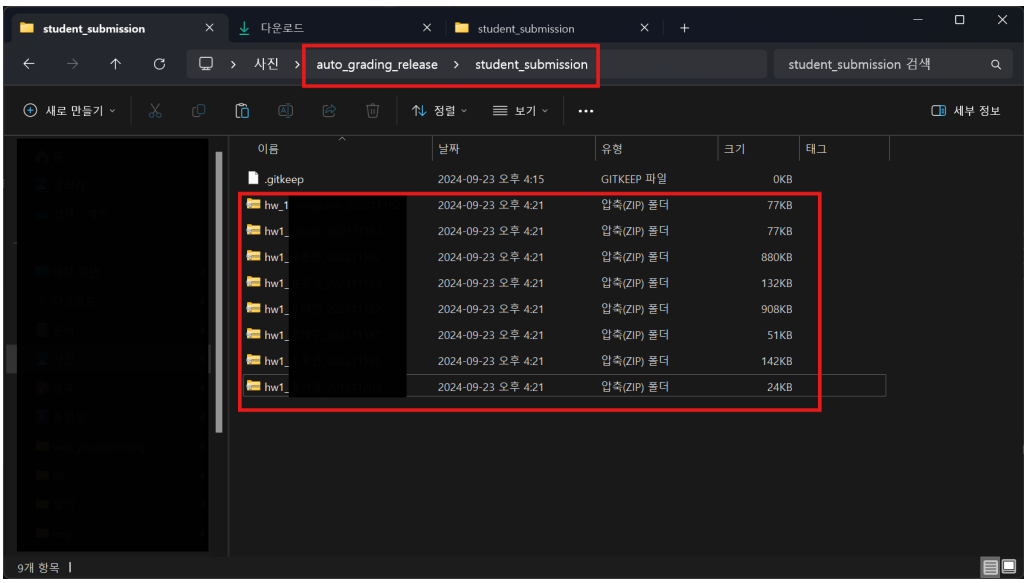
```
chmod 755 ./setup.sh
./setup.sh
```



4. 채점 파일 준비하기

채점하고자 하는 학생들의 `.zip` 파일을 `./student_submission` 폴더에 넣습니다.



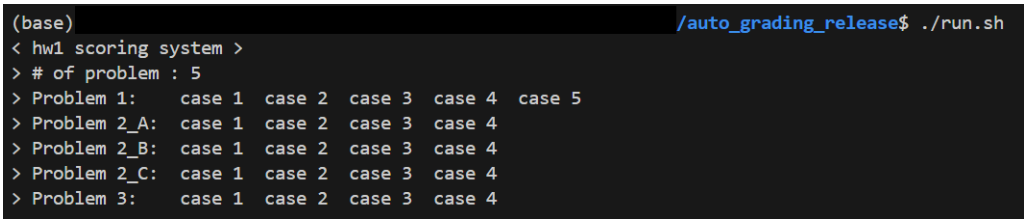


5. 채점 시작

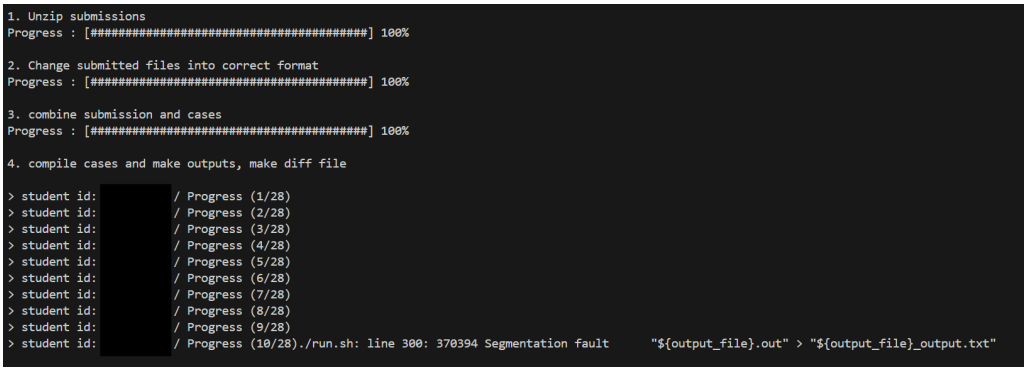
`./run.sh`를 실행합니다.

```
./run.sh
```

`run.sh`는 다음과 같이 실행될 때 현재 채점하는 Assignment의 정보를 보여줍니다. Assignment 정보에는 Assignment에 있는 문제, 문제 별 test case의 수가 포함됩니다. 다음 사진은 Assignment 1의 정보를 보여줍니다.



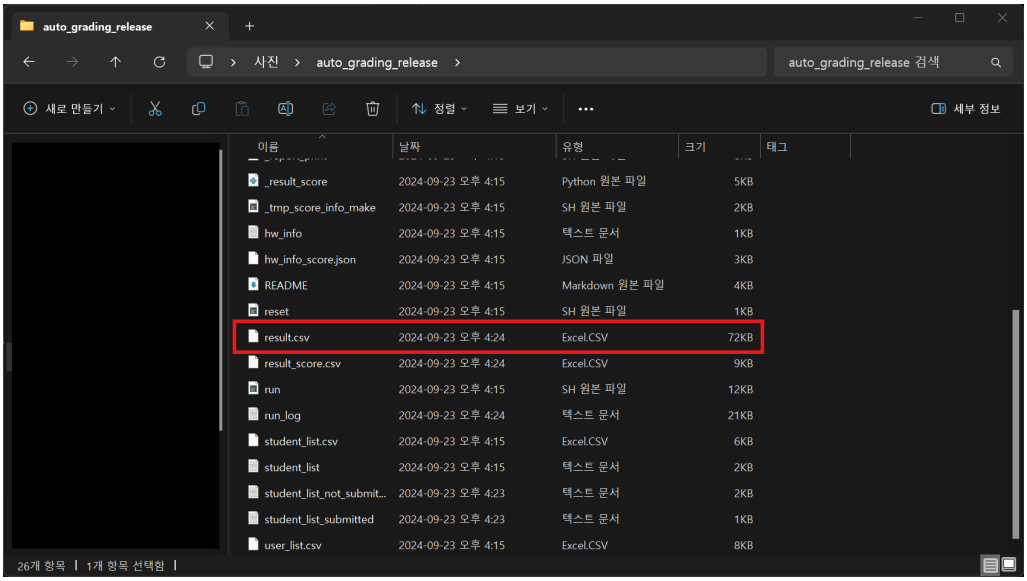
`run.sh`는 실행되는 동안 진행 상황을 다음과 같이 보여줍니다. 만약 컴파일 과정에서 segmentation fault가 일어났다면, 해당 학생이 제출한 코드 중 하나가 segmentation fault가 일어났다는 것을 의미합니다. 채점 시 참고하면 됩니다.



6. 결과 확인

`run.sh`는 다음 파일들을 생성합니다.

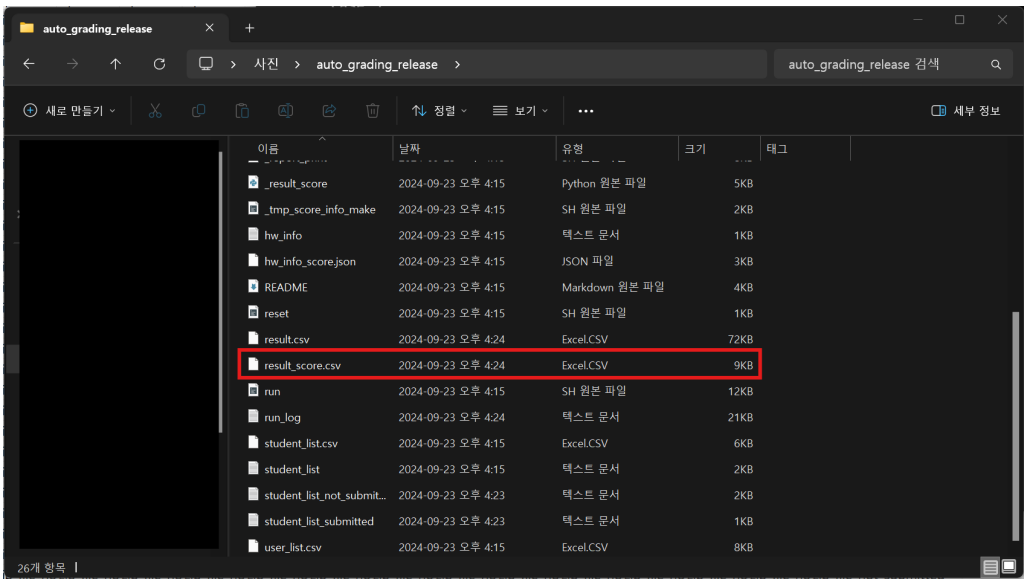
1. 모든 학생의 테스트 케이스 통과 여부를 담고 있는 `result.csv`



엑셀로 열면 다음과 같이 학생별로 테스트 케이스의 결과 정보들을 담고 있습니다.

student_id	1	1-case-1	1-case-2	1-case-3	1-case-4	1-case-5	2_A	2_A-case-	2_A-case-	2_A-case-	2_A-case-	2_B	2_B-case-	2_B-case-	2_B-case-	2_B-case-	2_C	2_C-case-	2_C-case-	2_C-case-	2_C-case-	3	3-case-1	3-case-2	3-case-3	3-case-4	total-score
201811183	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-not-submitted
201811014	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-not-submitted
201811034	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-not-submitted
201811203	file-submi	pass	pass	pass	pass	pass	file-submi	pass	pass	pass	pass	file-submi	fail	fail	fail	fail	file-submi	fail	pass	pass	pass	file-submi	pass	pass	pass	pass	
201911090	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-not-submitted
201911024	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-not-submitted
202011144	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-no	zip-file-not-submitted

2. 모든 학생의 테스트 케이스 별 점수 여부와 총 점을 담고 있는 `result_score.csv`

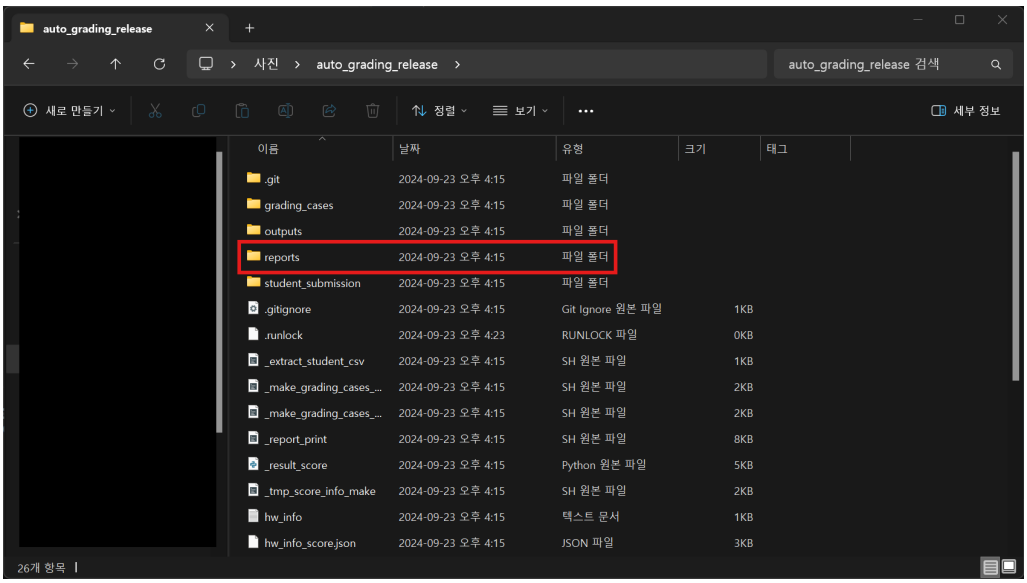


엑셀로 열면 다음과 같이 학생별로 테스트 케이스에서 얻은 점수와 총점을 담고 있습니다.

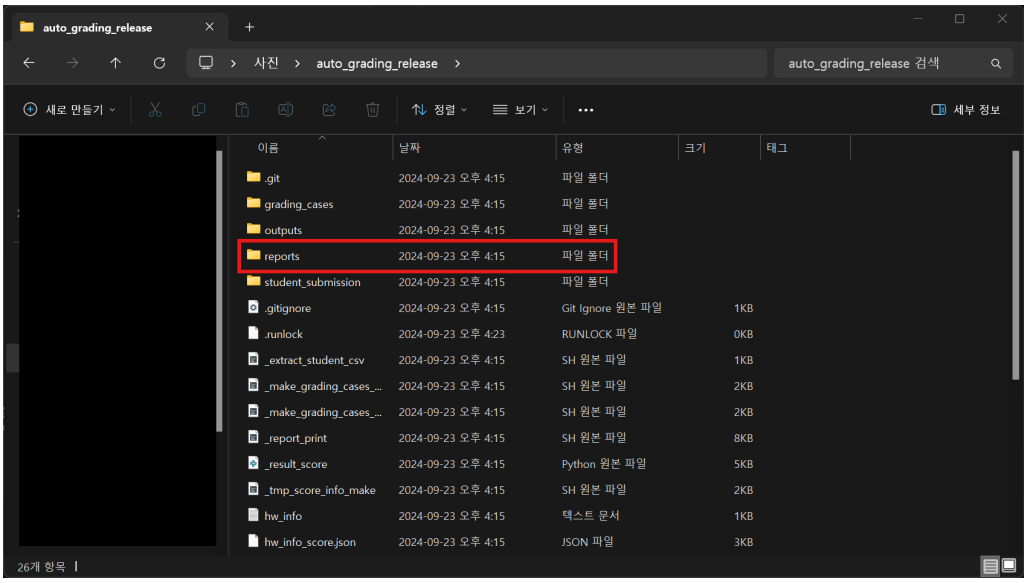
student_id	1	1-case-1	1-case-2	1-case-3	1-case-4	1-case-5	2_A	2_A-case-	2_A-case-	2_A-case-	2_A-case-	2_B	2_B-case-	2_B-case-	2_B-case-	2_B-case-	2_C	2_C-case-	2_C-case-	2_C-case-	2_C-case-	3	3-case-1	3-case-2	3-case-3	3-case-4	total-score
201811183	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
201811014	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
201811034	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
201811203	1	1	1	1	1	1	0	0.5	0.5	0.5	0.5	0	0	0	0	0	0	0	0.5	0.5	0.5	4	1	1	1	1	17.5
201911090	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
201911024	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
202011144	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

3. 학생 별 평가 개요, 제출한 코드와 컴파일 결과, 출력 결과를 보여주는 마크다운 리포트

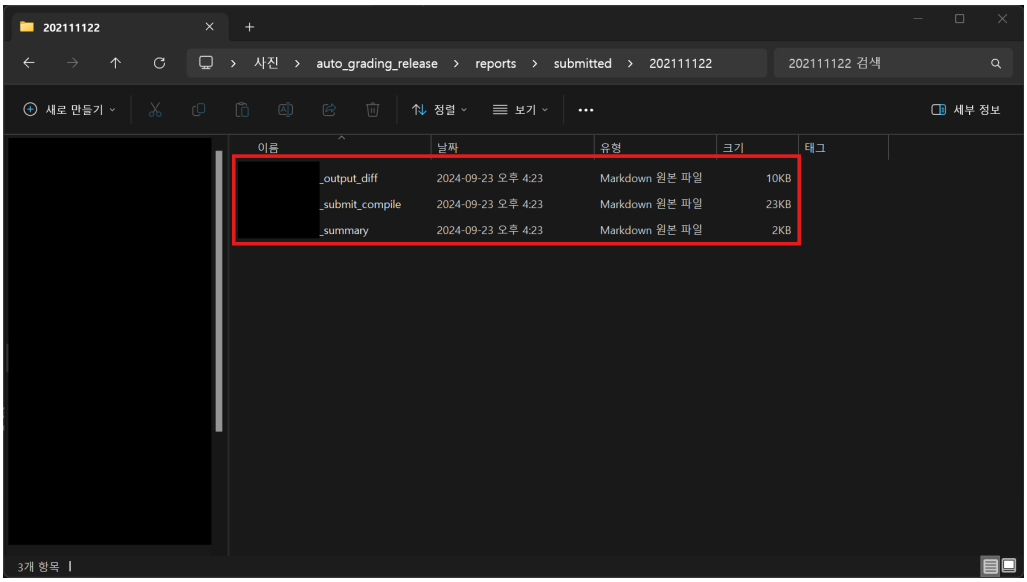
다음 폴더에 마크다운 리포트가 저장되어 있습니다.



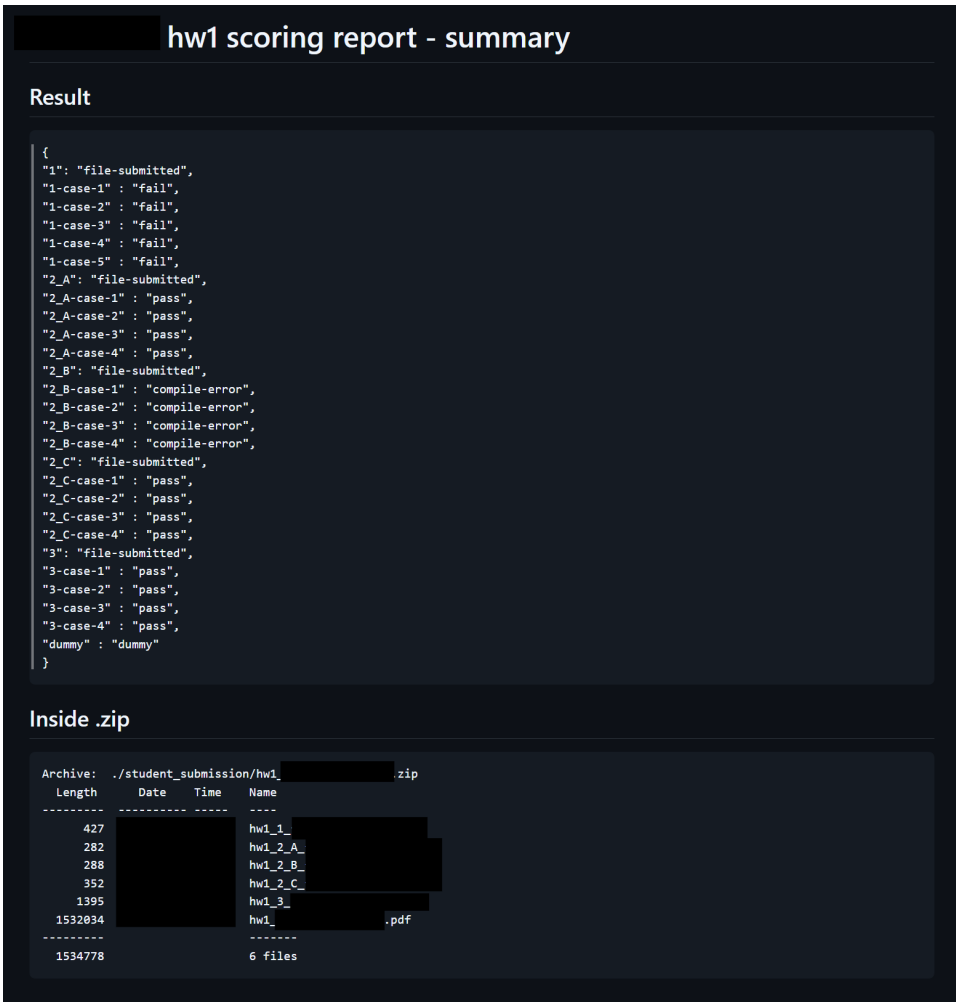
`not_submitted`에는 zip 파일을 제출 안 한 학생들의 리포트가, `submitted` 폴더에는 zip 파일을 제출 한 학생들의 리포트가 있습니다.



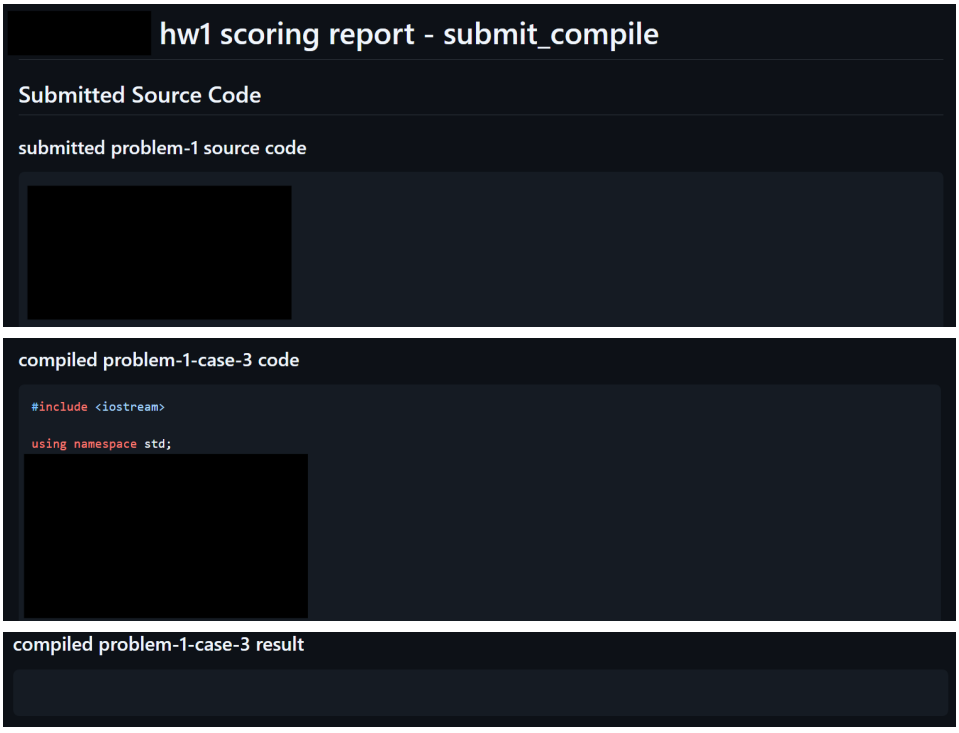
`not_submitted`, `submitted` 폴더에는 학생들의 학번별로 폴더가 생성되어 있는데, 폴더 속에 학생별 리포트 파일이 있습니다. 마크다운 파일들은 vscode를 이용해 열면 렌더링 된 상태로 볼 수 있습니다.



`_summary.md`는 학생의 채점 결과와 학생이 제출한 zip 파일 속 내용물에 대한 정보를 볼 수 있습니다.

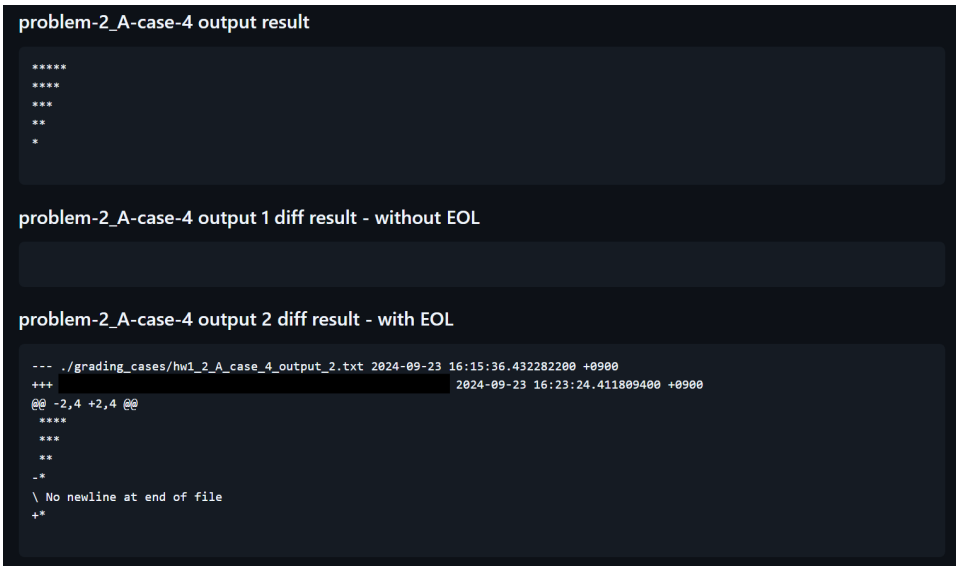


_submit_compile.md는 학생이 제출한 소스코드와 이를 바탕으로 컴파일 한 소스코드, 컴파일 결과를 볼 수 있습니다. 컴파일 결과에 아무것도 없으면 컴파일이 정상적으로 진행되었다는 의미이고, 컴파일 결과에 내용이 있다면 컴파일 에러가 일어났다는 의미입니다.



_output_diff.md는 테스트 케이스 별 학생 코드의 출력 결과, 정답 출력 결과와의 diff 결과를 보여줍니다. 정답 출력 결과는 두 종류로 파일의 맨 끝에 EOL 문자가 있는 정답과 EOL 문자가 없는 정답 두 가지로 나뉩니

다. 둘 중 하나라도 동일하다면 정답 처리 합니다.

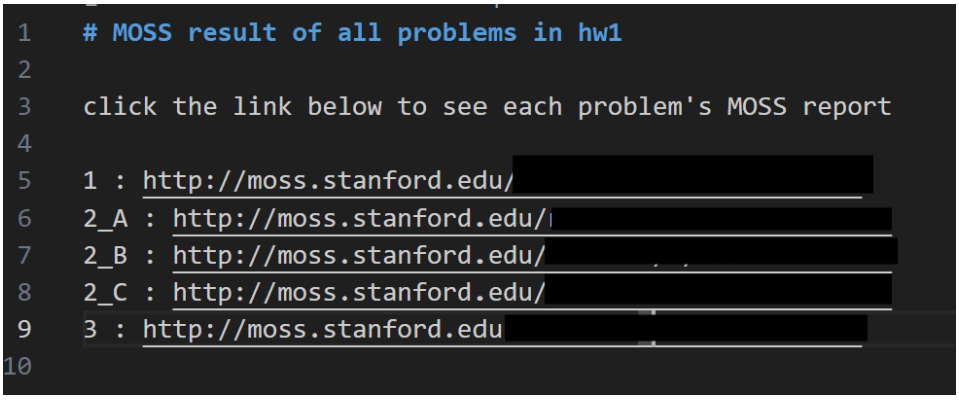


출력 형식을 제대로 지킨 경우에는 제대로 채점이 되지만, 간혹 의미적으로는 맞지만 출력 형식을 지키지 않았거나, 제출 파일 이름 형식을 지키지 않아 채점이 되지 않을 수 있습니다. 이 경우에는 결과 파일을 바탕으로 직접 채점을 해야 할 수 있습니다.

4. 학생들간의 코드 유사도를 측정한 결과를 담고 있는 `result_moss.md`

유사도는 MOSS를 이용해 측정한 결과입니다. 파일 안에 각 문제 별 유사도 측정 결과를 볼 수 있는 링크가 있습니다. 간단한 과제의 경우 유사도가 의미 없을 수 있지만, 텀 프로젝트 같이 복잡한 과제의 경우는 유사도가 중요하게 작용할 수 있습니다.

다음은 assignment 1의 경우를 표시한 경우입니다.



각 링크로 들어가면, 어떤 파일이 얼마나 유사한지를 퍼센트로 표시하고 있습니다. 정렬은 유사도가 높은 순서대로 입니다.

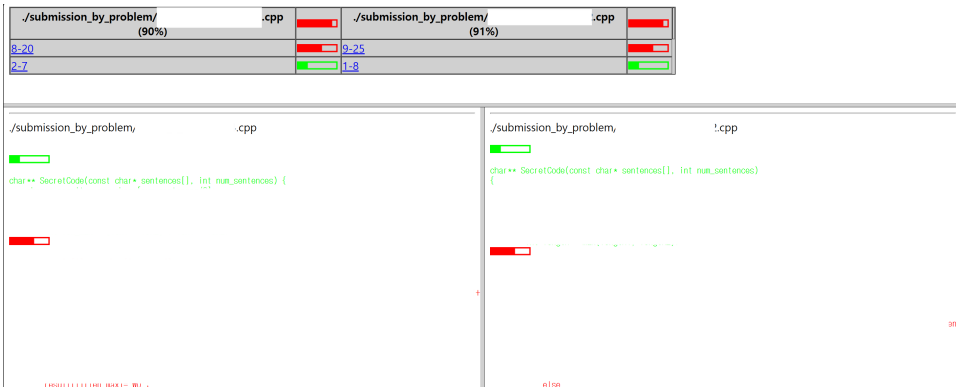
Moss Results

Options -l cc -m 10

[How to Read the Results | Tips | FAQ | Contact | Submission Scripts | Credits]

File 1		File 2		Lines Matched
./submission_by_problem/	.cpp (90%)	./submission_by_problem	.cpp (91%)	25
./submission_by_problem/	.cpp (82%)	./submission_by_problem	.cpp (83%)	28
./submission_by_problem/	.cpp (64%)	./submission_by_problem	.cpp (83%)	28
./submission_by_problem/	.cpp (64%)	./submission_by_problem	.cpp (82%)	23
./submission_by_problem/	.cpp (81%)	./submission_by_problem	.cpp (90%)	26
./submission_by_problem/	.cpp (64%)	./submission_by_problem	.cpp (90%)	28
./submission_by_problem/	.cpp (78%)	./submission_by_problem	.cpp (78%)	27
./submission_by_problem/	.cpp (80%)	./submission_by_problem	.cpp (80%)	21
./submission_by_problem/	.cpp (59%)	./submission_by_problem	.cpp (70%)	19
./submission_by_problem/	.cpp (73%)	./submission_by_problem	.cpp (73%)	24
./submission_by_problem/	.cpp (72%)	./submission_by_problem	.cpp (72%)	25
./submission_by_problem/	.cpp (65%)	./submission_by_problem	.cpp (71%)	28
./submission_by_problem/	.cpp (71%)	./submission_by_problem	.cpp (65%)	28
./submission_by_problem/	.cpp (64%)	./submission_by_problem	.cpp (71%)	20
./submission_by_problem/	.cpp (50%)	./submission_by_problem	.cpp (71%)	22
./submission_by_problem/	.cpp (68%)	./submission_by_problem	.cpp (68%)	24
./submission_by_problem/	.cpp (46%)	./submission_by_problem	.cpp (46%)	20
./submission_by_problem/	.cpp (50%)	./submission_by_problem	.cpp (52%)	14
./submission_by_problem/	.cpp (65%)	./submission_by_problem	.cpp (72%)	14
./submission_by_problem/	.cpp (72%)	./submission_by_problem	.cpp (64%)	21
./submission_by_problem/	.cpp (62%)	./submission_by_problem	.cpp (62%)	21

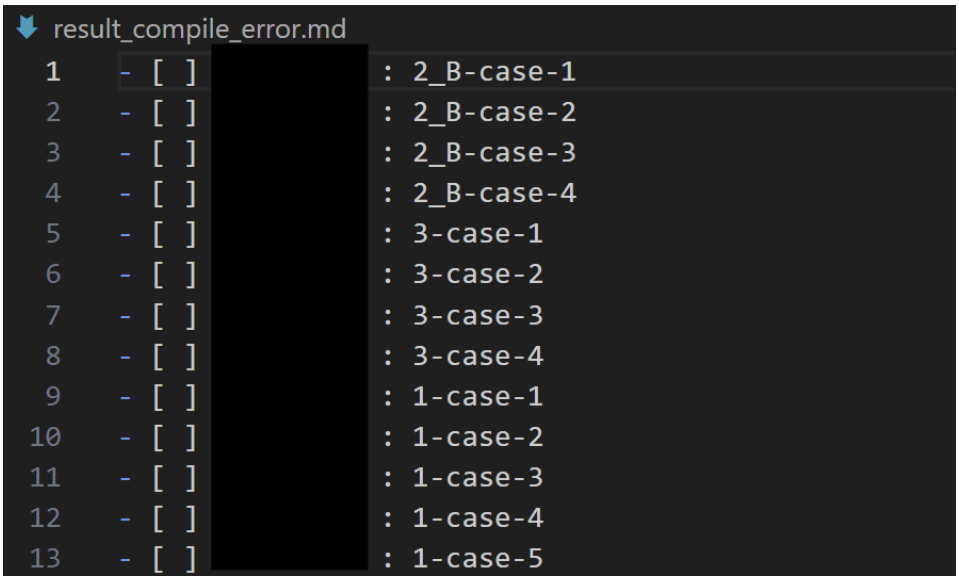
여기서 파일을 클릭하면, 두 소스코드가 어떤 부분에서 유사한지를 보여줍니다. 이 정보들을 종합해서 채점 시 판단하면 됩니다.



5. 학생들의 complie error, fail 여부 등을 담고 있는 `result_compile_error.md`, `result_fail_list.md`, `result_file_not_submitted_list.md`, `result_zip_file_not_submitted.md`

채점 시 확인하기 편하도록 이번에 추가했습니다. 각 목록을 보고 채점시 참고하면 됩니다.

다음은 `result_compile_error.md`의 예시로, 학생의 학번과 컴파일 에러가 난 문제 및 케이스를 보여줍니다.



7. 재실행

채점 프로그램을 다시 실행하기 위해서는 `./reset.sh`를 실행 한 후 `./run.sh`를 실행해야 합니다.

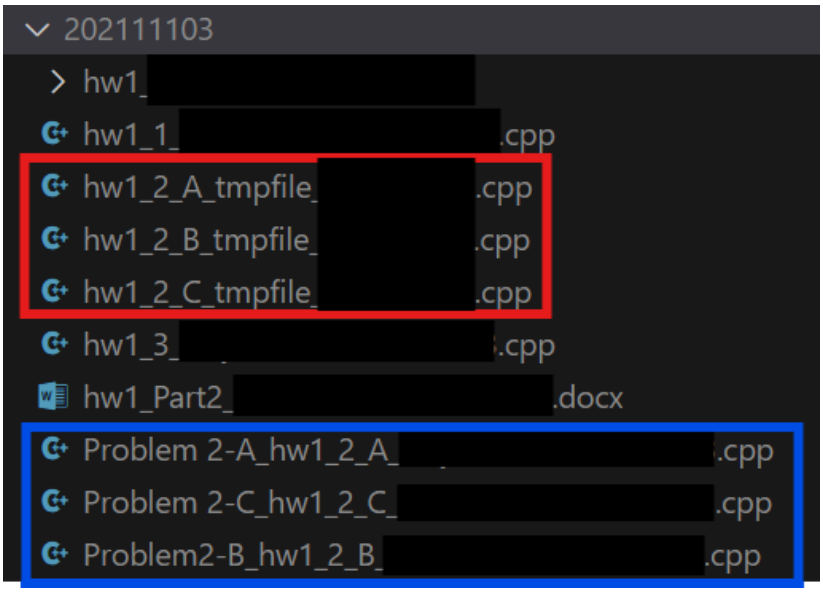
```
./reset.sh
```

또는, `./run.sh`를 실행한 이후, 특정 학생이 제출한 코드를 `./student_submission/학생학번`에서 수정한 뒤 다시 채점하고자 한다면, `./student_submission/학생학번/수정하고자하는파일.cpp`에서 소스 코드를 수정한 다음 `./run_extra 학생학번`을 실행하면 됩니다. 이러면 새로 수정한 소스코드를 바탕으로 해당 학생만 새로 채점합니다.

```
./run_extra.sh 학생학번
```

이때, 소스코드 중 `_tmpfile_`이라 표시된 소스코드가 있다면, **해당 소스코드를 수정해야 합니다.**

아래 예시를 보면, 실제 학생이 제출한 코드는 **파란색 사각형** 안 코드입니다. 만약 해당 문제의 소스코드를 수정하고자 한다면, **파란색 사각형** 속 소스코드를 수정하는 것이 아니라 **빨간색 사각형** 안 소스코드를 수정해야 합니다.



Troubleshooting

만약 /bin/bash^M을 찾을 수 없다는 오류 메시지가 `./setup.sh`를 실행할 때 나온다면, 다음 명령어를 입력해 주세요.

```
dos2unix ./setup.sh
```