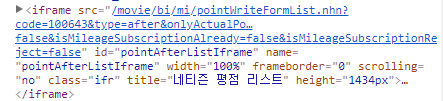
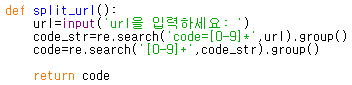


* 크롤링에 필요한 모듈을 불러옴.

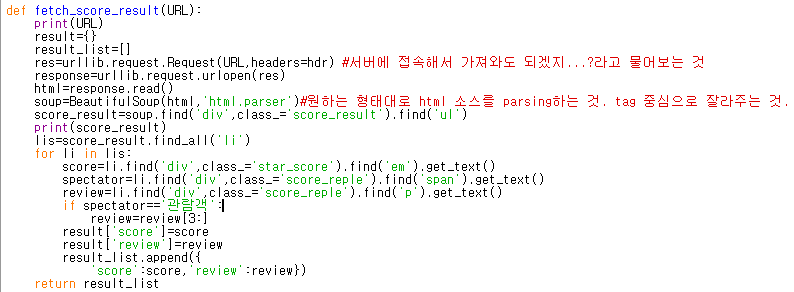
url소스는 홈페이지를 봤을 때



이 부분에서 가져옴.



가져오고자 하는 url을 입력하고, 그 url을 정규식을 이용해서 코드 뒷부분의 숫자만 가져오는 ‘split\_url’을 정의.

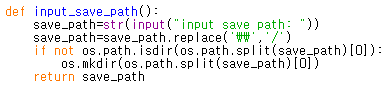


* Fetch\_score\_result(URL)

URL 소스를 서버에 접속해서 가져올 수 있도록 요청한 후, 그 후 소스를 열고 읽어옴. 이를 beautiful soup이 이를 parsing해서 tag중심으로 잘라줌.

그 후에, score\_result부분을 찾아온다.

다음에는 list안에 li태그를 다 찾음. li에서 평점 부분과 평가 주체를 가져옴. 이를 result에 딕셔너리로 저장하고(이때 여러 개의 li를 각각의 딕셔너리에 집어넣게 됨), 이를 result\_list라는 리스트에 append시킴.



* 저장경로를 설정하기 위한 함수. Input 함수를 save\_path에 string 형태로 저장하고, /를 ‘\\’로 바꿔줌. 이때, save\_path를 쪼개서 내용을 담고자 하는 파일을 [1], 그 앞의 모든 경로를 [0]으로 해서, 해당 경로가 존재하면 그 폴더에 파일을 저장하고, 존재하지 않을 경우에는 폴더 생성 후 파일을 저장한다.



* 코드를 입력 받고, 이를 save path에 인코딩 후 저장을 한다. 페이지는 첫번째 페이지의 내용을 가져옴. 이때, 게시물 개수가 10단위로 입력되면 다음으로 넘어가고, 그렇지 않으면 게시물 개수의 입력이 계속될 것. 다음으로 넘어간다면, l\_count에 1을 저장하고 isLoop=True로 저장해놓음. 이후, isLoop가 True일 때, URL을 담아서 fetch\_score\_result를 호출하고 이를 result\_list에 저장함. 이때, result\_list는 점수와 리뷰가 적혀있는 count개의 딕셔너리로 구성된 리스트임. 이후, 리스트 안의 딕셔너리 요소를 각각 score와 review로 나누어서 write함과 동시에 이를 구분하는 ====를 같이 write함. Write한 후에 l\_count의 값은 하나씩 증가하게 되고 이게 count값을 넘는 순간 for문이 멈추고 페이지가 다음으로 넘어가면서 다음페이지를 읽어옴. 읽어오는 사이에, 다음에 4부터 10 사이까지 난수 생성을 한 뒤, 생성된 난수만큼의 초 동안 시간이 흐르고, 이를 출력. If 문의 조건에 맞게 된다면, while문은 break가 되고 끝.
* 페리카나 실행결과

