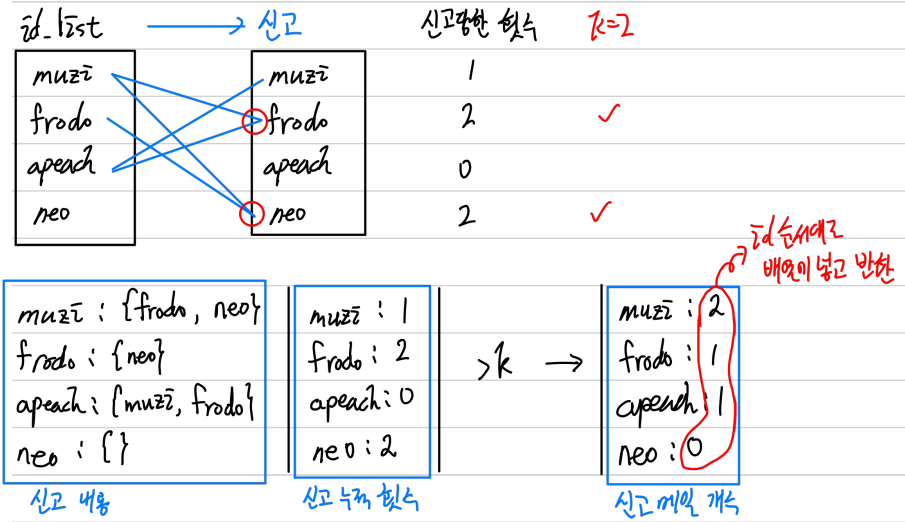
신고 결과 받기



신고 당한 사람 / 신고한 사람

String	HashSet
muzi	{apeach}
frodo	{muzi, apeach}
apeach	{}
neo	{muzi, frodo}

HashSet의 길이 값 ⇒ muzi +1, apeach +1, muzi +1, frodo +1

muzi : 2, frodo : 1, apeach : 1, neo : 0

1. 신고 당한 사람, 신고 메일 개수를 담을 Map 선언 및 id_list 순서로 초기화

```
Map<String, HashSet<String>> reportedMap = new HashMap<>();
Map<String, Integer> reportMailMap = new HashMap<>();

for (String id : id_list) {
    reportedMap.put(id, new HashSet<>());
    reportMailMap.put(id, 0);
}
```

2. report 배열을 활용해서 reportedMap에 데이터 삽입

```
for (String str : report) {
    2-1. 신고자, 피신고자 데이터 추출
    String[] content = str.split(" "); // 0: 신고자, 1: 피신고자
    String reporter = content[0];
    String reported = content[1];

    2-2. reportedMap에 넣기
    reportedMap.get(reported).add(reporter);
}
```

3. reportedMap의 entry들 각각 실행

```
for (Map.Entry<String, HashSet<String>> reportEntry : reportedMap) {
    Set<String> reporterSet = new HashSet<>(reportEntry.getValue());
```

3-1. reporterSet의 크기가 누적 신고 횟수이므로 k보다 작으면 다음 entry로 넘어가기

```
    if (reporterSet.size() < k) {
        continue;
    }
```

3-2. reporterSet 안의 reporter들의 메일 개수 +1

```
    for (int i=0; i < reporterSet.size(); i++) {
        String reporter = reporterSet.get(i);
        reportMail(Map.put(reporter, reportMail(Map.get(reporter)+1);
    }
}
```

4. 반환하기 위해 배열에 데이터 넣기

```
int[] answer = new int[id_list.length];

for (int i=0; i < id_list.length; i++){
    answer[i] = reportMail(Map.get(id_list[i]));
}

return answer;
```