Game Party Manager

Time Limit 3000 ms

게임 파티(이하 파티)들을 관리하는 프로그램을 구현하라.

(게임 파티: 목적 달성을 위한 게임 내 모임. 혼자 처치하기 어려운 보스 또는 몬스터를 처치하기 위해 모인 이용자 모임을 지칭한다. 출처 네이버)

각 게임 파티는 최소 1명에서 4명까지의 파티원으로 구성되어 있다.

특정 게임 id를 대상으로 몇 개의 파티에 포함되어 있는지 확인하고자 한다.

그리고 2명의 id를 입력받아 두 명 중 한 명이라도 참여하고 있는 모든 파티의 수를 확인하거나

두 명이 모두 참여하고 있는 파티의 수를 확인하려고 한다.

부정행위로 인해 특정 id가 정지될 수 있다.

정지된 파티원이 포함된 파티 역시 게임을 진행할 수 없으므로

파티원의 정지가 풀릴때까지 파티도 임시 정지가 된다.

임시 정지된 파티의 경우 파티 수 계산에서 배제된다.

정지된 id의 정지가 풀릴 경우 파티원들 중 다른 id에 문제가 없다면 파티의 정지도 풀리게 된다.

하지만 다른 id 역시 정지가 되어 있다면 파티의 정지는 유지된다.

파티원 정지와 무관하게 파티 자체가 삭제될 수 있다.

삭제된 파티는 복구되지 않는다.

다음 함수들을 구현해야 한다.

**void** init(int n);  
user code에서 필요한 초기화를 수행한다.  
매 TC 초기에 한 번 불린다.  
n: TC에서 사용되는 파티들의 최대 수이다. (4~50,000)

**void** addParty(int index, int m, **char** members[4][11]);  
새로운 파티 정보를 추가한다.  
init() 함수에서 사용되는 최대 n 회수만큼 호출된다.(호출 회수는 n보다 적을 수 있다)  
index: 해당 파티의 index이다. 0부터 시작되며 순차적으로 주어지며 closeParty()에서도 사용된다.  
m: 파티에 포함된 파티원 수를 나타낸다. (1~4)  
member: 각 파티원의 id. 각 id는 4~10자이며 소문자 알파벳과 숫자로 구성되어 있다. 한 파티에 동일한 id가 두 번 주어지지 않는다.

**void** closeParty(int index);  
파티를 제거한다. 한 번 제거된 파티는 다시 복구되지 않는다.  
index: addParty()를 통해서 전달한 파티의 index이다. 존재하지 않는 파티 id는 주어지지 않는다.

**void** suspendMember(**char** member[11]);  
특정 게임 id를 계정 중지시킨다.  
아직 파티에 포함되어 있지 않더라도 계정을 중지 시킬 수 있다.  
id가 중지 되기 전에 구성된 파티이거나 혹은 중지된 이후에 구성된 파티 모두 정지상태가 된다.  
member: 계정 중지할 id

**void** recoveryMember(**char** member[11]);  
중지된 id를 복구시킨다.  
정지되지 않은 id를 대상으로 호출되지는 않는다.(모든 id는 기본적으로 정지상태가 아니다)  
해당 id가 포함된 파티는 정지 상태가 풀린다. 단 그 파티의 모든 파티원이 계정 중지되지 않은 경우에만 풀린다.  
주어진 id가 복구되더라도 이 id를 포함한 파티의 다른 파티원이 계정 중지 상태라면 그 파티는 여전히 정지상태가 된다.

int search(**char** members[2][11], int mode);  
주어진 id를 포함하고 정지상태가 아닌 파티 수를 확인한다.  
member: 검색할 한 개 또는 두 개의 id가 주어진다. 이 id는 모두 suspend 되어 있지 않으며 둘은 반드시 다른 id이다.  
mode(0): 0번째 member가 포함되고 정지상태가 아닌 파티들의 수를 계산한다.  
mode(1): 주어진 두 member가 모두 속해 있고 정지상태가 아닌 파티들의 수를 계산한다.(AND 연산)  
mode(2): 주어진 두 member들 중 한 명이라도 속해있는 파티들의 수를 계산한다. 둘 다 포함된 경우 하나로 계산한다.(OR 연산)

제약사항:

주어지는 member id는 최대 10,000개를 넘지 않으며 각 TC별 최대 파티 수를 넘지 않는다.

main에서 호출하는 user 함수들의 호출 회수는 최대 100,000을 넘지 않는다. (init() 제외)

입력:

처음 30개 TC는 검증용으로 각 TC의 search() 결과 중 10개까지만 결과를 비교하고 나머지는 조합로 비교한다.

만약 local에서 디버깅을 하고자 한다면 입력중 앞 30개를 제거하면 뒤에 나오는 20개 TC를 사용하면 된다.

뒤에 나오는 20개 TC는 모든 search()에 대해서 결과값을 비교할 수 있다.

그리고 main.cpp의 FULL\_CHECK를 true로 define 해야 한다.

#define FULL\_CHECK true

false일 경우에는 10개만 비교하고 나머지는 sum으로 비교하는데 앞쪽 30개 TC에 해당된다.