**[제한 사항]**

|  |  |
| --- | --- |
| 시간 | 25개 테스트케이스를 합쳐서 C++ 의 경우 3초 / Java 의 경우 3초 |
| 메모리 | 힙, 정적 메모리 합쳐서 256MB 이내, 스택 메모리 1MB 이내 |

|  |
| --- |
| **①   C 또는 C++로 답안을 작성하시는 응시자께서는 검정시스템에 제출 시, Language 에서 C++ 를 선택하신 후 제출하시기 바랍니다.**  **②   Main 과 User Code 부분으로 구성되어 있습니다.**  **ㅇ   A.**Main         : 수정할 수 없는 코드이며, 채점 시 비 정상적인 답안 검출 등 평가를 위한 로직이 추가 될 수 있습니다.  **ㅇ   B.**User Code  : 실제 응시자가 작성해야 하는 코드이며, 제출 시에는 표준 입출력 함수가 포함되어 있으면 안 됩니다.  **③   Local PC 에서 프로그래밍 시 유의 사항**  **ㅇ   A.**2개의 파일을 생성하셔야 합니다. ( main.cpp / solution.cpp 또는 Solution.java / UserSolution.java )  **ㅇ   B.**Main 부분의 코드를 main.cpp 또는 Solution.java에 복사해서 사용하시기 바랍니다.  **ㅇ   C.**sample\_input.txt 를 사용하시기 위해서는 Main 부분의 코드 내에  **ㅇ   C.**표준 입력을 파일로 전환하는 코드 ( 주석처리 되어 있음 ) 의 주석을 풀어서 사용하시면 됩니다.  **ㅇ   D.**User Code 부분의 코드를 작성하신 후 서버에 제출하실 때,  **ㅇ   D.**디버깅을 위한 표준 입출력 함수를 모두 삭제 또는 주석 처리해 주셔야 합니다.  **④   문제 내에 제약조건을 모두 명시하지 않으므로 주어지는 코드를 분석하셔야 합니다.**  **⑤   코드는 개발 언어에 따라 상이할 수 있으므로, 작성할 언어를 기준으로 분석하셔야 합니다.** |

**※** 본 문제는 임직원의 S/W 문제해결 역량 향상을 위한 **Professional 실전형 연습문제**이며

함께 제공되는 **Testcase 또한 학습을 위한 연습용**입니다.

**[문제 설명]**

조별 경기에서 N명의 선수들의 점수를 관리하는 프로그램을 작성한다. (4 ≤ N ≤ 100,000)

각 선수들은 서로 다른 ID를 가지며 ID는 1에서 N까지의 수 중 하나이다.

편의상, ID가 a인 선수를 a번 선수라고 한다.

처음에는 모든 선수들이 1인 1조로 출전한다. 따라서 모든 선수들은 서로 다른 조에 속해있다.

조별 경기 시작 전 모든 선수들의 점수는 0점이다.

조별 경기가 진행되면서 서로 다른 두 조가 하나의 조로 합쳐질 수 있다.

하나의 조로 합쳐질 때 선수들 개개인의 기존 점수는 유지된다.

그리고 합쳐진 조로 경기를 진행한 결과는 해당 조에 속한 모든 선수들에게 동일하게 반영된다.

조별 경기는 여러 조들 중 서로 다른 두 조를 선정하여 경기를 진행한다.

경기의 결과가 판정되어 점수가 결정되면 해당 점수를 각 조에 속한 선수들의 점수에 반영한다.

점수는 1 이상 100 이하이며 승리한 조가 얻는 점수와 패배한 조가 잃는 점수는 같다.

예를 들어, A조와 B조가 경기를 하여 A조가 3점을 얻는 경우 B조는 3점을 잃는다.

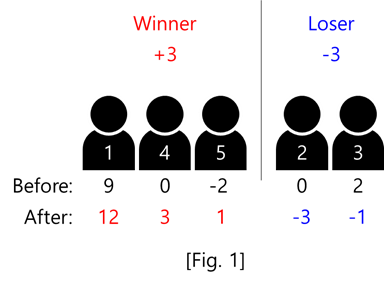
또한 A조에 속한 선수들은 모두 3점을 얻으며 B조에 속한 선수들은 모두 3점을 잃는다.

**[Fig. 1]** 은 두 조의 경기 결과를 나타낸 그림이다.

Before 에 표시된 점수는 경기전 각 선수들의 점수이며 After에 표시된 점수는 경기후 각 선수들의 점수이다.

승리한 조에는 1, 4, 5번 선수들이 속해있으며 모두 3점의 점수를 얻는다.

패배한 조에는 2, 3번 선수들이 속해있으며 모두 3점의 점수를 잃는다.



**아래 API 설명을 참조하여 각 함수를 구현하라.**

※ 아래 함수 signature는 C/C++에 대한 것으로 다른 언어에 대해서는 제공되는 Main과 User Code를 참고하라.

아래는 User Code 부분에 작성해야 하는 API 의 설명이다.

|  |
| --- |
| **void init(int N)** |
| 각 테스트 케이스의 맨 처음에 호출된다.  N은 조별 경기에 참가한 선수들의 수이다.    모든 선수들은 1인 1조로 출전한다.  최초 모든 선수들의 점수는 0점이다.    ***Parameters***  N : 조별 경기에 참가한 선수들의 수 (4 ≤ N ≤ 100,000) |
| **void updateScore(int mWinnerID, int mLoserID, int mScore)** |
| 서로 다른 두 조가 선정되어 경기를 진행하고 경기의 결과를 갱신하는 함수이다.  ID가 mWinnerID인 선수가 속한 조가 승리하여 해당 조에 속한 선수들은 mScore의 점수를 얻는다.  ID가 mLoserID인 선수가 속한 조가 패배하여 해당 조에 속한 선수들은 mScore의 점수를 잃는다.    mWinnerID 선수와 mLoserID 선수의 조가 같은 경우는 주어지지 않는다.    ***Parameters***  mWinnerID : 승리한 조에 속한 선수의 ID (1 ≤ mWinnerID ≤ N)  mLoserID : 패배한 조에 속한 선수의 ID (1 ≤ mLoserID ≤ N)  mScore : 승리한 조가 얻는 점수 (1 ≤ mScore ≤ 100) |
| **void unionTeam(int mPlayerA, int mPlayerB)** |
| 서로 다른 두 개의 조를 선정하여 하나의 조로 합치는 함수이다.  ID가 mPlayerA인 선수가 속한 조와 ID가 mPlayerB인 선수가 속한 조를 하나의 조로 합친다.    mPlayerA 선수와 mPlayerB 선수의 조가 같은 경우는 주어지지 않는다.    ***Parameters***  mPlayerA : 선정된 선수의 ID (1 ≤ mPlayerA ≤ N)  mPlayerB : 선정된 선수의 ID (1 ≤ mPlayerB ≤ N) |
| **int getScore(int mID)** |
| ID가 mID인 선수의 점수를 확인한다.    ***Parameters***  mID : 점수를 확인할 선수의 ID (1 ≤ mID ≤ N)    ***Returns***     ID가 mID인 선수의 점수 |

**[예제]**

아래와 같이 함수가 호출되는 경우를 살펴보자.

|  |  |  |
| --- | --- | --- |
| **#** | **Function** | **Return** |
| 1 | init(6) |  |
| 2 | updateScore(4, 5, 25) |  |
| 3 | getScore(4) | 25 |
| 4 | getScore(5) | -25 |
| 5 | unionTeam(4, 5) |  |
| 6 | updateScore(1, 5, 50) |  |
| 7 | getScore(4) | -25 |
| 8 | updateScore(2, 3, 100) |  |
| 9 | unionTeam(2, 3) |  |
| 10 | unionTeam(5, 1) |  |
| 11 | unionTeam(6, 3) |  |
| 12 | updateScore(5, 6, 75) |  |
| 13 | getScore(1) | 125 |
| 14 | getScore(2) | 25 |

[#1] init(6)

조별 경기에 6명의 선수가 출전하며 최초 1인 1팀으로 모든 선수들의 점수는 0점이다.

텍스트, 무기이(가) 표시된 사진

자동 생성된 설명

[#2] updateScore(4, 5, 25)

[#3] getScore(4)

[#4] getScore(5)

4번 선수와 5번 선수가 경기를 하여 4번 선수가 25점을 얻고 5번 선수가 25점을 잃는다.



[#5] unionTeam(4, 5)

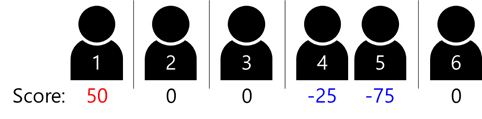
[#6] updateScore(1, 5, 50)

[#7] getScore(4)

4번 선수와 5번 선수가 하나의 조로 합치며 선수들의 점수는 그대로 유지된다.

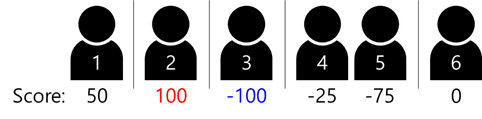
1번 선수가 속한 조와 5번 선수가 속한 조가 경기를 하여 1번 선수가 속한 조가 50점을 얻고 5번 선수가 속한 조가 50점을 잃는다.

4번 선수는 5번 선수와 같은 조이므로 50점을 잃어 현재까지 4번 선수의 점수는 -25점이다.



[#8] updateScore(2, 3, 100)

2번 선수와 3번 선수가 경기를 하여 2번 선수가 100점을 얻고 3번 선수가 100점을 잃는다.



[#9] unionTeam(2, 3)

[#10] unionTeam(5, 1)

[#11] unionTeam(6, 3)

[#12] updateScore(5, 6, 75)

[#13] getScore(1)

[#14] getScore(2)

#9, #10, #11 진행 후 1, 4, 5번 선수가 하나의 조가 되고 2, 3, 6번 선수가 하나의 조가 된다.

5번 선수가 속한 조와 6번 선수가 속한 조가 경기를 하여 5번 선수가 속한 조가 75점을 얻고 6번 선수가 속한 조가 75점을 잃는다.

1번 선수는 5번 선수와 같은 조이므로 75점을 얻어 현재까지 1번 선수의 점수는 125점이다.

2번 선수는 6번 선수와 같은 조이므로 75점을 잃어 현재까지 2번 선수의 점수는 25점이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

**[제약사항]**

1. 각 테스트 케이스 시작 시 init() 함수가 1회 호출된다.

2. 각 테스트 케이스에서 updateScore() 함수의 총 호출 수는 50,000회 이하이다.

3. 각 테스트 케이스에서 unionTeam() 함수의 총 호출 수는 N보다 작다.

4, 각 테스트 케이스에서 getScore() 함수의 총 호출 수는 50,000회 이하이다.

5. updateScore() 함수와 unionTeam() 함수에서 같은 조의 두 선수가 주어지는 경우는 없다.

**[입출력]**

입출력은 제공되는 Main 부분의 코드에서 처리하므로 User Code 부분의 코드에서는 별도로 입출력을 처리하지 않는다.

Sample input 에 대한 정답 출력 결과는 “#TC번호 결과” 의 포맷으로 보여지며 결과가 100 일 경우 정답, 0 일 경우 오답을 의미한다.