**[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 |

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

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

텍스트이(가) 표시된 사진

자동 생성된 설명

**[문제 설명]**

병사들을 관리하는 프로그램을 작성하고자 한다.

병사들은 각각 고유번호, 소속팀, 평판 점수를 가지고 있다.

프로그램은 아래의 기능들을 수행할 수 있어야 한다.

1. 병사 고용
2. 병사 해고
3. 병사의 평판 점수 변경
4. 특정 팀에 속한 병사들의 평판 점수를 일괄 변경
5. 특정 팀 안에서 가장 평판 점수가 높은 병사를 검색

아래 API 설명을 참조하여 각 함수를 구현하라.

※ 아래 함수 signature는 C/C++에 대한 것으로 Java에 대해서는 제공되는 Solution.java와 UserSolution.java를 참고하라.

아래는 User Code 부분에 작성해야 하는 API 의 설명이다.

|  |
| --- |
| **void init()** |
| 각 테스트 케이스의 처음에 호출된다.  테스트 케이스의 시작 시 고용된 병사는 없다. |
| **void hire(int mID, int mTeam, int mScore)** |
| 고유번호가 mID, 소속팀이 mTeam, 평판 점수가 mScore인 병사를 고용한다.  한 테스트 케이스 내에서 동일한 mID를 가진 병사가 여러 번 고용되는 경우는 없음이 보장된다.    ***Parameters***  mID : 고유번호 (1 ≤ mID ≤ 100,000)  mTeam : 소속팀 (1 ≤ mTeam ≤ 5)  mScore : 평판 점수 (1 ≤ mScore ≤ 5) |
| **void fire(int mID)** |
| 고유번호가 mID인 병사를 해고한다.  fire() 함수 호출 시, 고유번호가 mID인 병사가 고용되어 있음이 보장된다.    ***Parameters***  mID : 고유번호 (1 ≤ mID ≤ 100,000) |
| **void updateSoldier(int mID, int mScore)** |
| 고유번호가 mID인 병사의 평판 점수를 mScore로 변경한다.  고유번호가 mID인 병사가 고용되어 있음이 보장된다.    ***Parameters***  mID : 고유번호 (1 ≤ mID ≤ 100,000)  mScore : 평판 점수 (1 ≤ mScore ≤ 5) |
| **void updateTeam(int mTeam, int mChangeScore)** |
| 소속팀이 mTeam인 병사들의 평판 점수를 모두 변경한다.  소속팀이 mTeam인 병사가 한 명 이상 고용되어 있음이 보장된다.    updateTeam() 함수에서의 평판 점수 변경은 아래의 규칙에 따른다.  ‘변경 전 평판 점수 + mChangeScore’가 5보다 클 경우, 평판 점수를 5로 변경한다.  ‘변경 전 평판 점수 + mChangeScore’가 1보다 작을 경우, 평판 점수를 1로 변경한다.  그 외의 경우, 평판 점수를 ‘변경 전 평판 점수 + mChangeScore’로 변경한다.    ***Parameters***  mTeam : 소속팀 (1 ≤ mTeam ≤ 5)  mChangeScore : 평판 점수의 변화량 (-4 ≤ mChangeScore ≤ 4) |
| **int bestSoldier(int mTeam)** |
| 소속팀이 mTeam인 병사들 중 평판 점수가 가장 높은 병사의 고유번호를 반환한다.  평판 점수가 가장 높은 병사가 여러 명일 경우, 고유번호가 가장 큰 병사의 고유번호를 반환한다.  소속팀이 mTeam인 병사가 한 명 이상 고용되어 있음이 보장된다.    ***Parameters***  mTeam : 소속팀 (1 ≤ mTeam ≤ 5)    ***Returns***  평판 점수가 가장 높은 병사의 고유번호 |

**[예제]**

아래 표에서 [a, b, c]는 고유번호가 a, 소속팀이 b, 평판 점수가 c인 병사를 의미한다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Order** | **Function** | **Status** | **Return** |
| 1 | init() |  |  |
| 2 | hire(16, 1, 5) | [16, 1, 5] |  |
| 3 | hire(21, 1, 5) | [16, 1, 5], [21, 1, 5] |  |
| 4 | bestSoldier(1) | [16, 1, 5], [21, 1, 5] | 21 |
| 5 | fire(21) | [16, 1, 5] |  |
| 6 | bestSoldier(1) | [16, 1, 5] | 16 |
| 7 | hire(25, 1, 4) | [16, 1, 5], [25, 1, 4] |  |
| 8 | hire(30, 1, 2) | [16, 1, 5], [25, 1, 4], [30, 1, 2] |  |
| 9 | updateTeam(1, 1) | [16, 1, 5], [25, 1, 5], [30, 1, 3] |  |
| 10 | bestSoldier(1) | [16, 1, 5], [25, 1, 5], [30, 1, 3] | 25 |
| 11 | updateTeam(1, 2) | [16, 1, 5], [25, 1, 5], [30, 1, 5] |  |
| 12 | bestSoldier(1) | [16, 1, 5], [25, 1, 5], [30, 1, 5] | 30 |
| 13 | updateSoldier(30, 2) | [16, 1, 5], [25, 1, 5], [30, 1, 2] |  |
| 14 | bestSoldier(1) | [16, 1, 5], [25, 1, 5], [30, 1, 2] | 25 |
| 15 | updateTeam(1, -4) | [16, 1, 1], [25, 1, 1], [30, 1, 1] |  |
| 16 | hire(1, 1, 3) | [16, 1, 1], [25, 1, 1], [30, 1, 1], [1, 1, 3] |  |
| 17 | updateTeam(1, -1) | [16, 1, 1], [25, 1, 1], [30, 1, 1], [1, 1, 2] |  |
| 18 | bestSoldier(1) | [16, 1, 1], [25, 1, 1], [30, 1, 1], [1, 1, 2] | 1 |
| 19 | hire(100000, 5, 1) | [16, 1, 1], [25, 1, 1], [30, 1, 1], [1, 1, 2]  [100000, 5, 1] |  |
| 20 | bestSoldier(5) | [16, 1, 1], [25, 1, 1], [30, 1, 1], [1, 1, 2]  [100000, 5, 1] | 100000 |

**[제약사항]**

1. 각 테스트 케이스 시작 시 init() 함수가 호출된다.

2. 각 테스트 케이스에서 hire() 함수의 호출 횟수는 100,000 이하이다.

3. 각 테스트 케이스에서 fire() 함수의 호출 횟수는 100,000 이하이다.

4. 각 테스트 케이스에서 updateSoldier() 함수의 호출 횟수는 100,000 이하이다.

5. 각 테스트 케이스에서 updateTeam() 함수의 호출 횟수는 100,000 이하이다.

6. 각 테스트 케이스에서 bestSoldier() 함수의 호출 횟수는 100 이하이다.

**[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.