**[기출P-0058] 탱크**

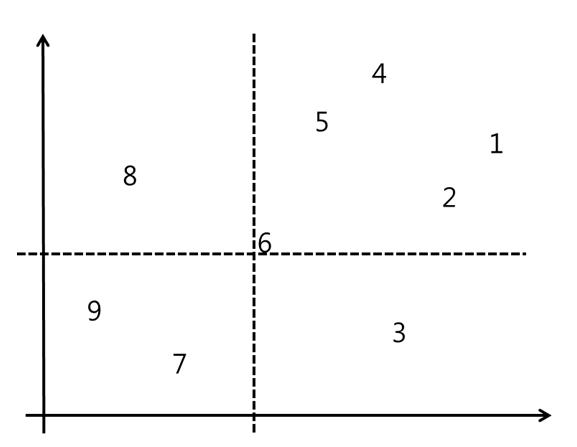
|  |  |
| --- | --- |
| 시간 제한 | **20** 개의 Test Input 입력시 C/C++ **1**초, Java **1.5**초 |
| 메모리 제한 | Stack : **5** Mbytes  /  Total : **256** Mbytes |
| 코드길이 제한 | **128** Kbytes |
| 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려준다.  모든 케이스에 대해 정답일 경우 **Pass**, 그 외의 경우 **Fail** 이다. |

2차원 평면에 N개의 탱크가 위치하고 있다. 각 탱크에는 점수가 부여되어 있다.

임의의 탱크와 그 부여된 점수 S 에 대해서, **그 탱크의 왼쪽 아래 방향에 위치하는 탱크에 부여된 점수는 S 보다 크다는 것이 보장**된다.

점수가 S 인 한 탱크를 공격하면, **그 탱크의 오른쪽 위 방향에 위치하는 모든 탱크들 중 점수가 S 보다 작은 모든 탱크들에 부여된 점수가 합산되어 점수**로 주어진다. **탱크를 공격한 이후에도 탱크들에 어떤 변화가 생기지는 않는다.**

예를 들어 아래 그림에서 점수가 6 인 탱크를 공격하면 오른쪽 위 방향에 위치하는 4 개의 탱크 중 점수가 6보다 작은 1, 2, 4, 5의 점수가 합산된 12점이 점수로 주어진다.



탱크들의 위치와 점수를 입력으로 받아 **모든 탱크를 공격했을 때 받게 되는 점수들의 총합을 계산하는 프로그램**을 작성하라.

**[입력]**   
입력 파일의 제일 첫째 줄에는 파일에 포함된 케이스의 수 T가 주어진다.각 케이스의 첫째 줄에 탱크의 개수 N (1 ≤ N ≤ 100,000) 이 주어진다.

다음 N 개의 줄 각각에 탱크의 위치와 점수가 주어진다. 이들 중 첫 두 자연수는 탱크의 x와 y좌표이며, 세번째 자연수는 탱크에 부여된 점수이다.

모든 좌표 값은 1 이상 1,000,000이하이다. 탱크에 부여된 점수는 1 이상 200,000 이하인 자연수이다.

**두 탱크가 동일한 x좌표에 위치하는 경우는 없으며, 두 탱크가 동일한 y좌표에 위치하는 경우도 없다. 또한, 탱크에 부여된 점수는 모두 다르다.**

**[출력]**   
각 테스트 케이스의 답을 순서대로 표준출력으로 출력하며, 각 케이스마다 줄의 시작에 “#x”를 출력하여야 한다. 이때 x는 케이스의 번호이다.

같은 줄에, 모든 탱크를 공격했을 때 얻게 되는 점수의 합을 출력한다. **출력해야 하는 값이 4 바이트 정수형 변수의 표현 범위를 넘어갈 수 있음에 주의**하라.

**[입출력 예]**   
(입력)   
2    
5    
1 1 5   
2 2 4   
3 3 3   
4 4 2   
5 5 1   
9    
3 1 7   
4 4 6   
1 3 9   
7 2 3   
8 5 2   
2 6 8   
9 7 1   
5 8 5   
6 9 4

(출력)

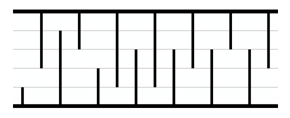
#1 20   
#2 77   
  
(sample\_input.txt 에 대한 출력)   
#1 167436   
#2 166417   
#3 182761   
#4 669232868   
#5 1425634765276   
#6 443318195116   
#7 1787841655257   
#8 47548858537543   
#9 25678408034773   
#10 34362071934741

**[교육P-0016] 동굴**

|  |  |
| --- | --- |
| 시간 제한 | **10**개의 Test Input 입력시 C/C++ **1**초 / Java **1.5**초 |
| 메모리 제한 | Stack : **1** Mbytes  /  Total : **256** Mbytes |
| 제출횟수 제한 | **제한 없음** |
| 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려주며  그 의미는 다음과 같다.  **Pass : Test Input에 대하여 모두 정답 처리된 경우**  **Fail : Test Input에 대하여 일부 혹은 모두 오답 처리된 경우**  ※ Test Input : 검정 시스템에서 제출된 코드 실행 時 자동으로 입력되는 Input Data |
| 평가기준 | 정답 처리된 Test Input의 개수 |

**※ 본 문제는 교육에 활용되는 연습 문제로 별도의 힌트가 제공되지 않습니다.**

**[문제]**  
벌레 한 마리가 동굴을 지나려고 한다. 모두가 알다시피 동굴은 석순과 종유석이 굉장히 많은 공간이다. 이 벌레는 이렇게 장애물이 많은 동굴을 지날 것이다. 동굴의 길이는 N미터이고, 높이는 H미터다. N은 항상 짝수이고, 장애물은 석순과 종유석이 번갈아 등장하고, 첫 장애물은 항상 석순이다. 아래 그림은 N=14, H=5의 예이다.



이 벌레는 장애물을 굳이 피하지 않고, 처음 정한 높이에서 일직선으로 장애물을 부수면서 지나간다. 벌레가 아래에서 4번째 구간으로 지나가면, 아래 그림과 같이 8개의 장애물을 부순다. 하지만 첫 번째나 다섯 번째 구간으로 날아간다면 벌레는 7개의 장애물만 부수면 된다.



동굴의 크기  N과 장애물 배치가 주어졌을 때, 최소 개수의 장애물을 부수면서 지나갈 수 있는 구간의 수와 그 때 부수는 장애물의 수를 구하는 프로그램을 작성하라.

**[입력]**  
첫 번째 줄에 테스트 케이스의 갯수 T가 주어진다. 이어서 차례로 T개 테스트 케이스가 주어진다. (1 ≤ T ≤ 10) 각 케이스의 첫 번째 줄에는 장애물의 수 N과 동굴의 높이 H가 주어진다. (2 ≤ N ≤ 200,000, 2 ≤ H ≤ 500,000) 이후 N개의 줄에는 장애물의 크기가 순서대로 주어진다. 장애물의 크기는 H보다 작은 양의 정수이다.

**[출력]**  
각각의 테스트 케이스에 대하여 #x(x는 테스트 케이스 번호를 의미)를 출력하고 공백을 하나 둔 최소 개수의 장애물을 지날 때 지나는 장애물의 수와 그러한 구간의 수를 공백으로 구분하여 출력한다.

**[입출력 예]**  
(입력)  
2                                                   ← 2 test cases in total  
6 6                                                ← 1st case  
3  
3  
3  
3  
3  
3  
14 5                                              ← 2nd case  
1  
3  
4  
2  
2  
4  
3  
4  
3  
3  
3  
2  
3  
3

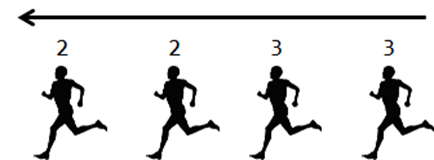
(출력)  
#1 3 6  
#2 7 2

**[교육P-0020] 달리기**

|  |  |
| --- | --- |
| 시간 제한 | **15**개의 Test Input 입력시 C/C++ **1**초 / Java **1.5**초 |
| 메모리 제한 | Stack : **1** Mbytes  /  Total : **256** Mbytes |
| 제출횟수 제한 | **제한 없음** |
| 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려주며  그 의미는 다음과 같다.  **Pass : Test Input에 대하여 모두 정답 처리된 경우**  **Fail : Test Input에 대하여 일부 혹은 모두 오답 처리된 경우**  ※ Test Input : 검정 시스템에서 제출된 코드 실행 時 자동으로 입력되는 Input Data |
| 평가기준 | 정답 처리된 Test Input의 개수 |

**※ 본 문제는 교육에 활용되는 연습 문제로 별도의 힌트가 제공되지 않습니다.**

**[문제]**  
N명의 달리기 선수가 달리고 있다. 현재 i등으로 달리고 있는 선수의 달리기 실력은 A[i]이다. 만약 현재 자신보다 앞에서 달리고 있으면서 자신보다 실력이 낮으면 역전 가능성이 있다고 말할 수 있다. 자신이랑 실력이 같거나 실력이 더 좋은 사람을 역전할 가능성은 아예 없다.



달리기에 경주에서 역전이 나오면 관중에 환호가 커지므로 앞으로 역전이 최대 몇 번 나올지 궁금해졌다. 현재 달리기 상황이 주어졌을 때, 앞으로 역전이 최대 몇 번 나올지 계산하는 프로그램을 작성하시오.

**[입력]**  
입력 파일에는 여러 테스트 케이스가 포함될 수 있다. 파일의 첫째 줄에 케이스의 개수 T가 주어지고, 이후 차례로 T개 테스트 케이스가 주어진다. (1 ≤ T ≤ 15) 각 케이스의 첫 번째 줄에는 달리기 선수의 수 N 이 주어진다. (2 ≤ N ≤ 100,000) 둘째 줄에 N개의 자연수가 공백으로 구분되어 주어진다. 이 때 i번째로 주어지는 수는 A[i]를 의미하며, 이는 109 보다 크지 않다.

**[출력]**  
각 테스트 케이스의 답을 순서대로 표준출력으로 출력하며, 각 케이스마다 줄의 시작에 “#x”를 출력하여야 한다. 이때 x는 케이스의 번호이다. 같은 줄에, 앞으로 역전이 최대 몇 번 나올지 나타내는 수를 출력한다.

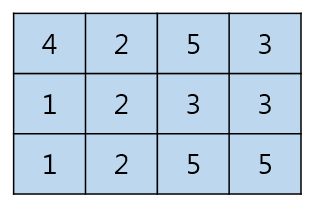
**[입출력 예]**  
(입력)  
2  
4  
2 2 3 3  
4  
3 1 2 4

(출력)  
#1 4  
#2 4

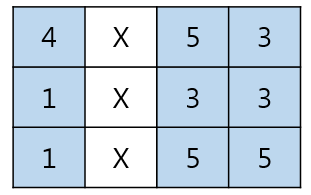
**[기출P-0037] 섬나라**

|  |  |
| --- | --- |
| 시간 제한 | **10** 개의 Test Input 입력시 C/C++ **1**초, Java **1.5**초 |
| 메모리 제한 | Stack : **5** Mbytes  /  Total : **256** Mbytes |
| 코드길이 제한 | **128** Kbytes |
| 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려준다.  모든 케이스에 대해 정답일 경우 **Pass**, 그 외의 경우 **Fail** 이다. |

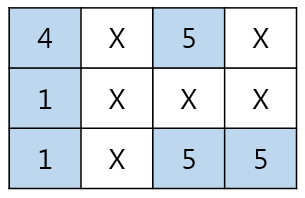
NxM 크기의 격자판으로 구성된 섬나라가 있다. 초기에 각 칸에는 자연수가 적혀있다. 아래는 N=3, M=4일 때 3x4 크기의 섬나라를 나타낸 표이다.



시간이 지남에 따라 특정 수가 적혀있는 칸들이 파괴된다. 예를 들어, 2가 적혀있는 칸을 모두 파괴하면 다음과 같이 된다.



어떤 칸과 상하좌우로 인접한 다른 칸은 서로 연결되어있다고 말한다. 서로 연결되어 있는 덩어리를 하나의 섬이라고 한다. 2 가 적혀있는 칸이 파괴되기 이전에는 섬나라는 하나의 섬으로 구성되었지만, 파괴된 이후에는 2개의 섬으로 구성되어있다. 이후에 3 이 적혀있는 칸을 모두 파괴하면 다음과 같이 된다.



파괴된 이후 3개의 섬이 존재한다.

처음 격자판에 적혀있는 수가 주어지고, 파괴 명령이 Q 개 주어진다. 이 때, 각 파괴 명령을 수행한 이후 존재하는 섬의 개수를 구하는 프로그램을 작성하시오.

**[입력]**

입력 파일의 제일 첫째 줄에는 케이스의 수 T가 주어진다. 각 케이스의 첫째 줄에 격자판 크기를 나타내는 두 자연수 N, M이 공백으로 구분되어 주어진다. (1 ≤ N, M ≤ 500) 둘째 줄부터 N개의 줄에 걸쳐 격자판에 적힌 수를 나타내는 M개의 자연수가 공백으로 구분되어 주어진다. 이 때, 주어지는 수는 100,000보다 크지 않다. N+2번째 줄에 파괴 명령의 개수 Q가 주어진다. (1 ≤ Q ≤ 100,000) N+3번째 줄에는 파괴 명령을 나타내는 Q개의 자연수가 순서대로 공백으로 구분되어 주어진다. 주어지는 자연수는 100,000보다 크지 않고, i번째로 주어지는 수가 k라는 것은 i번째 파괴 명령이 “k가 적혀있는 모든 격자칸을 파괴한다”라는 것을 의미한다.

**[출력]**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하며, 각 케이스마다 줄의 시작에 “#x”를 출력하여야 한다. 이때 x는 케이스의 번호이다. 같은 줄에, 각 파괴 명령을 수행한 **이후** 섬의 개수를 공백으로 구분되어 순서대로 출력한다.

**[입출력 예]**

**입력**

2                                              ← 2 test cases in total   
3 4                                            ← 1st case   
4 2 5 3   
1 2 3 3   
1 2 5 5   
2   
2 3   
5 4                                            ← 2nd case   
1 2 1 2   
2 1 2 1   
1 2 1 2   
2 1 2 1   
1 2 1 2   
3   
1 1 2

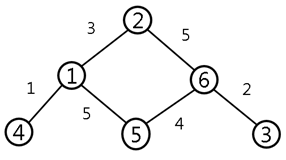
**출력**

#1 2 3   
#2 10 10 0   
  
(sample\_input.txt 에 대한 정답)   
=> sample\_input의 정답은 지면 관계상 sample\_input.txt 하단에 함께 작성하여 공유드리니, 풀이에 참고하여 주시기 바랍니다.

**[기출P-0039] 개미 굴 파기**

|  |  |
| --- | --- |
| 시간 제한 | **20** 개의 Test Input 입력시 C/C++ **1.5**초, Java **3**초 |
| 메모리 제한 | Stack : **20** Mbytes  /  Total : **256** Mbytes |
| 코드길이 제한 | **128** Kbytes |
| 채점 | 답안을 제출하면 Test Input에 대한 결과를 판정해서 실시간으로 알려준다.  모든 케이스에 대해 정답일 경우 **Pass**, 그 외의 경우 **Fail** 이다. |

여왕개미 한마리가 결혼 여행을 마치고 혼자 개미 집을 지으려고 한다. 여왕개미는 N개의 방이 있는 땅을 발견하였다. 방들은 1번부터 N번까지 번호가 붙어 있다. 방들을 연결하는 굴이 없어서 방들이 전혀 연결되지 않은 상태이다. N개의 서로 다른 방의 쌍들에 대해 그 쌍의 방을 연결하는 굴을 팔 수 있다는 것을 알고 있다. 각 쌍에 대해 굴을 파는 비용은 서로 다를 수 있다. 아래 그림은 6개의 방이 있는 상황을 보여 준다. 원 안에 써 있는 숫자는 방의 번호이며, 선은 굴을 팔 수 있는 쌍을 보여주고, 선 옆의 숫자는 그 굴을 파는 비용이다.



여왕개미는 **모든 방이 하나 이상의 굴을 통해서 연결될 수 있도록** 굴들을 파려고 한다. 물론 전체 비용을 최소화하고 싶다.

방의 개수, 굴로 연결이 가능한 방들의 쌍과, 그 비용을 입력으로 받아 최소 비용으로 굴을 파는 방법의 개수를 출력하는 프로그램을 작성하라.

**[입력]**   
입력 파일의 제일 첫째 줄에는 파일에 포함된 케이스의 수 T가 주어진다. 단, T ≤ 20이다. 케이스의 첫 줄에 방의 개수 N이 주어진다. (2 ≤ N ≤300,000)  다음 N개의 줄 각각에 굴을 파서 연결할 수 있는 방 번호의 쌍과 그 비용이 주어진다. 동일한 방 번호의 쌍이 두 번 이상 주어지는 경우는 없다. 비용은 1 이상 1,000,000이하의 자연수이다.

**[출력]**   
각 테스트 케이스의 답을 순서대로 표준출력으로 출력하며, 각 케이스마다 줄의 시작에 “#x”를 출력하여야 한다. 이때 x는 케이스의 번호이다. 같은 줄에 최소 비용으로 굴을 팔 수 있는 방법의 개수를 출력한다.

**[입출력 예]**   
입력   
2                                              ← 2 test cases in total   
6                                              ← 1st case   
2 1 3   
1 5 5   
3 6 2   
6 5 4   
2 6 5   
1 4 1   
3                                              ← 2nd case   
1 2 1   
3 2 1   
1 3 1

출력   
#1 2   
#2 3   
  
(sample\_input.txt에 대한 출력)   
#1 2   
#2 3   
#3 0   
#4 2   
#5 6