**문제1**

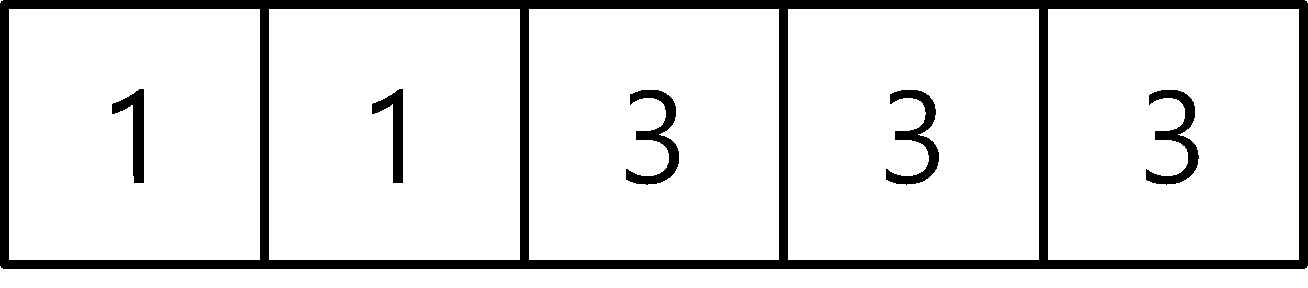
N명의 사람들이 좌우로 서 있다. 사람들은 왼쪽에서 순서대로 1번부터 번호가 붙어 있다. 사람들 사이의 친구 관계는 사람들이 들고 있는 자연수를 이용하여 아래 규칙으로 알아낼 수 있다. 아래 규칙에 따라 만들어지지 않는 친구 관계는 없다.

- 번호 i인 사람은 자연수 Di를 가지고 있다. ( 0<Di≤N )

- 번호 i인 사람은 번호 i+Di인 사람과 친구 관계이다. 친구 관계는 대칭적이다. 즉, 한쪽만 상대방이 친구라고 생각하는 경우는 없다. 만약 번호 i+Di인 사람이 존재하지 않는다면, 이 Di는 무시된다.

- 사람 A와 B가 친구 관계이고 B와 C가 친구 관계이면, A와 C도 친구 관계이다.

친구 관계인 극대 그룹은 사람들의 모임인데, 그룹 내의 모든 사람들이 친구 관계이고, 그 조건을 유지하면서 더 이상 사람을 추가할 수 없는 경우를 말한다. 예를 들어 아래와 같이 5명이 각자 가지고 있는 Di를 표시했다고 하자.



1번 사람과 2번 사람, 2번 사람과 3번 사람이 친구 관계라는 것은 직접 알 수 있다.

규칙에 따라 1번 사람과 3번 사람도 친구 관계이다.

3, 4, 5번 사람이 가지고 있는 수는 무시된다.

그룹 {1번 사람, 3번 사람}을 생각해 보자. 이 그룹 안의 모든 사람은 친구 관계이다.

하지만 2번 사람을 추가해도 모두 친구 관계가 되므로 {1번 사람, 3번 사람}은 친구 관계인 극대 그룹이 아니다.

그룹 {1번 사람, 2번 사람, 3번 사람}은 친구 관계인 극대 그룹이다.

그룹 {1번 사람, 4번 사람, 5번 사람}은 친구 관계가 아닌 쌍이 있으므로 친구 관계인 극대 그룹이 아니다.

친구 관계인 극대 그룹은 {1번 사람, 2번 사람, 3번 사람}, {4번 사람}, {5번 사람}의 3개가 있음을 알 수 있다.

각 사람이 들고 있는 Di들을 입력 받아 친구 관계인 극대 그룹의 개수를 출력하는 프로그램을 작성하시오.

- 제한시간: 전체 테스트 케이스는 50개 이하이며, 전체 수행 시간은 1초 이내. (Java 2초 이내)

제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,

그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,

이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.

C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,

제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다. ※ 언어별 기본 제공 소스코드 내용 참고

만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.

- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB

- 제출 제한 : 최대 10회

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.

파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 T가 주어지고,

이후 차례로 T개의 테스트 케이스가 주어진다. ( 1≤T≤50 )

각 테스트 케이스의 사람들의 수 N이 주어진다. ( N≤100,000 )

다음 줄에는 왼쪽부터 각 사람들이 들고 있는 자연수가 주어진다.

- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 80 점)

주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,

각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.

ㆍ 그룹 1 (30 점) : N≤1,000

ㆍ 그룹 2 (50 점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

\* 모든 테스트 케이스를 풀지 않고 일부분의 그룹에 속하는 테스트 케이스만을 푸는 경우에도 입력 받은 모든 케이스에 대해 (답이 틀릴지라도) 출력 양식에는 맞는 출력을 생성해야 점수가 반영되는 것이 보장된다.

\* 제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며, 그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.입력 파일에는 여러 테스트 케이스가 포함될 수 있다.

파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 T가 주어지고,

이후 차례로 T개의 테스트 케이스가 주어진다. ( 1≤T≤50 )

각 테스트 케이스의 사람들의 수 N이 주어진다. ( N≤100,000 )

다음 줄에는 왼쪽부터 각 사람들이 들고 있는 자연수가 주어진다.

- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 80 점)

주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,

각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.

ㆍ 그룹 1 (30 점) : N≤1,000

ㆍ 그룹 2 (50 점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

\* 모든 테스트 케이스를 풀지 않고 일부분의 그룹에 속하는 테스트 케이스만을 푸는 경우에도 입력 받은 모든 케이스에 대해 (답이 틀릴지라도) 출력 양식에는 맞는 출력을 생성해야 점수가 반영되는 것이 보장된다.

\* 제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며, 그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,

각 테스트 케이스마다 첫 줄에는 “Case #C”를 출력하여야 한다. 이때 C는 테스트 케이스의 번호이다.

그 다음 줄에, 친구 관계인 그룹의 수를 출력하시오.

**입출력예**

**<입력>**

2

5

1 1 3 3 3

10

8 10 5 4 2 5 1 3 1 9

**<출력>**

Case #1

3

Case #2

4

**문제2**

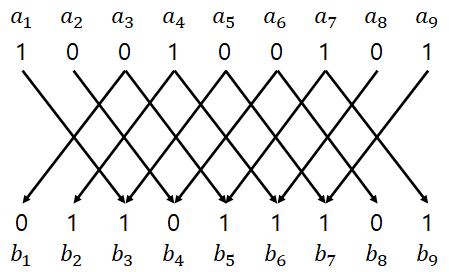
n 비트로 구성된 이진수 A=a1a2…an (ai는 0 또는 1)와 자연수 t 가 주어질 때, A로부터 아래와 같은 연산을 통해 새로운 이진수 B=b1b2…bn 를 만든다.

1. 초기에 bi (1≤i≤n) 는 모두 0으로 리셋한다.

2. 만약 i>t 이고 ai−t=1 이면 bi는 1로 둔다.

3. 만약 i≤n−t 이고 ai+t=1 이면 bi 는 1로 둔다.

예를 들어, 아래 그림에서 보인 것처럼 9 비트로 구성된 이진수 A=100100101 이고 t=2 인 경우, 변환을 통해 새로운 이진수 B=011011101 을 얻을 수 있다.



변환된 이진수 B와 자연수 t가 주어질 때, 이 정보로부터 역으로 A를 유추하는 프로그램을 작성하고자 한다.

예를 들어, 그림에서 보인 것처럼 B=011011101 이고 t=2 라면 가능한 A의 값은 100100101, 100110101, 000110100 등이다.

주어진 이진수 B와 자연수 t로부터 유추 가능한 A가 둘 이상일 경우, A의 값을 이진수로 보았을 때, 가장 작은 값을 출력하는 프로그램을 작성하시오. 주어진 이진수 B와 자연수 t 로부터 유추 가능한 A가 존재하지 않는 경우는 주어지지 않는다.

- 제한시간: 전체 테스트 케이스는 64개 이하이며, 전체 수행 시간은 1초 이내. (Java 2초 이내)

제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,

그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,

이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.

C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,

제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다. ※ 언어별 기본 제공 소스코드 내용 참고

만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.

- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB

- 제출 제한 : 최대 10회

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.  
파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 T 가 주어지고,  
이후 차례로  T개의 테스트 케이스가 주어진다.  (1≤T≤64)  
각 테스트 케이스의 첫 줄에는 변환된 이진수 B의 비트 수를 나타내는 정수 n과 자연수 t가 주어진다.  (2≤n≤50,000,1≤t<n)  
다음 줄에는 길이가 n 인 이진수가 주어진다.  
주어진 이진수 B와 자연수 t로부터 유추 가능한 A가 존재하지 않는 경우는 주어지지 않는다.  
  
- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 150 점)  
   주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,  
 각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.  
  
ㆍ 그룹 1 (31 점) : 이 그룹의 테스트 케이스에서는 n≤15이다.  
ㆍ 그룹 2 (38 점) : 이 그룹의 테스트 케이스에서는 n≤1,000이다.  
ㆍ 그룹 3 (81 점) : 이 그룹의 테스트 케이스에서는 n≤50,000이다.  
  
\* 모든 테스트 케이스를 풀지 않고 일부분의 그룹에 속하는 테스트 케이스만을 푸는 경우에도 입력 받은 모든 케이스에 대해 (답이 틀릴지라도) 출력 양식에는 맞는 출력을 생성해야 점수가 반영되는 것이 보장된다.  
  
\* 제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며, 그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #C”를 출력하여야 한다. 이때 C는 테스트 케이스의 번호이다.  
그 다음 줄에, 주어진 이진수 B와 자연수 t로부터 유추 가능한 A중, A의 값을 이진수로 보았을 때 가장 작은 값을 출력하시오.

**입출력예**

**<입력>**

2

5 1

00111

10 2

1111111000

**<출력>**

Case #1

00011

Case #2

0111100000

**문제3**

정점이 N개인 그래프가 있다. 이 그래프의 간선들 중 일부는 방향성이 있고 일부는 방향성이 정해지지 않았다.

이 그래프에서 방향성이 있는 간선들만 고려했을 때 사이클이 존재하지 않는다.

방향성이 정해지지 않은 간선들의 방향을 모두 정하려고 하는데, 모두 정하고 나서도 그래프에 사이클이 존재하지 않아야 한다.

아래 예 들에서 화살표로 표시된 간선은 방향성이 있는 간선, 점선으로 표시된 간선은 방향성이 정해지지 않은 것이다. 정점 안의 수는 정점의 번호이다.

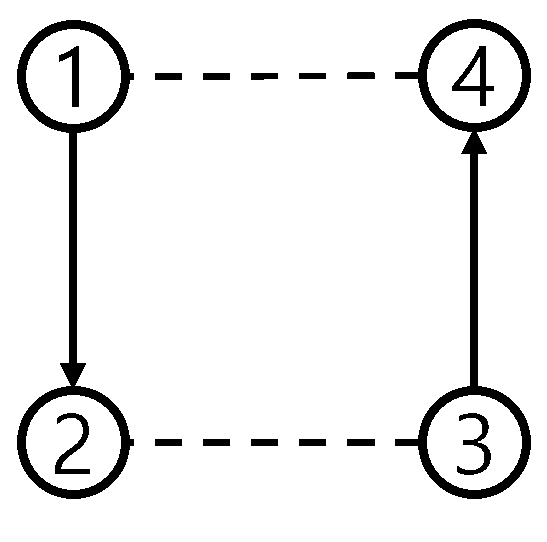


왼쪽의 경우 점선으로 표시된 간선의 방향성을 3번에서 2번으로 가는 것으로 정하면 사이클이 생기게 된다.

따라서 이 경우는 2번에서 3번으로 가는 것으로 방향성을 정해야 한다.

오른쪽의 경우 점선으로 표시된 간선의 방향성을 어떻게 정하더라도 사이클이 생기지 않는다.

방향성을 정할 수 있는 방법이 여러가지임에 주의하라.



위의 경우 점선으로 표시된 간선들의 방향성을 4번에서 1번, 2번에서 3번으로 정하는 경우를 제외하면 다른 모든 경우에서 사이클이 생기지 않는다.

방향성이 정해지지 않은 간선들의 방향성을 모두 정하되, 그래프에 사이클이 없도록 하는 프로그램을 작성하라.

간선의 방향성을 정해서 출력하는 방법이 복잡할 수 있으므로 아래 [입력]과 [출력] 부분을 주의 깊게 확인하기 바란다.

또, 사이클이 없도록 방향성을 정하는 방법이 여러가지인 경우에도 주의해서 답을 출력해야 한다.

- 제한시간: 전체 테스트 케이스는 70개 이하이며, 전체 수행 시간은 1초 이내. (Java 2초 이내)

제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,

그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,

이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.

C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,

제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다. ※ 언어별 기본 제공 소스코드 내용 참고

만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.

- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB

- 제출 제한 : 최대 10회

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.

파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 T가 주어지고,

이후 차례로 T개의 테스트 케이스가 주어진다. (1≤T≤70)

각 테스트 케이스의 첫 줄에는 정점의 개수 N, 방향성 정해진 간선의 개수 M, 방향성이 정해지지 않은 간선의 개수 K가 주어진다. ( 3≤N≤500, 0≤M≤2,000, 1≤K≤2,000 )

정점들은 1번부터 N번까지 번호가 붙어 있다.

둘째 줄부터 M개의 각 줄에는 두 정점의 번호가 주어진다. 첫 정점에서 두번째 정점으로 방향이 있는 간선이 있다는 의미이다.

그 다음 K개의 각 줄에는 두 정점의 번호가 주어진다. 두 정점을 연결하는 방향이 정해지지 않은 간선이 있다는 의미이다. 이 간선들은 주어진 순서대로 번호가 붙어 있다.

하나의 정점을 간선이 연결하는 경우는 없으나, 동일한 쌍의 정점들에 대해 여러 개의 간선이 존재할 수 있다.

- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 180 점)

주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,

각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.

ㆍ 그룹 1 (41 점) : 이 그룹의 테스트 케이스에서는 3≤N≤10, 0≤M≤20, 1≤K≤10이다.

ㆍ 그룹 2 (52 점) : 간선들에 방향을 부여할 수 있는 방법이 유일하다.

ㆍ 그룹 3 (87 점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

\* 모든 테스트 케이스를 풀지 않고 일부분의 그룹에 속하는 테스트 케이스만을 푸는 경우에도 입력 받은 모든 케이스에 대해 (답이 틀릴지라도) 출력 양식에는 맞는 출력을 생성해야 점수가 반영되는 것이 보장된다.

\* 제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며, 그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #C”를 출력하여야 한다. 이때 C는 테스트 케이스의 번호이다.  
그 다음 줄에, 0과 1로 이루어진 문자열을 출력한다. 각 문자는 K개의 방향성이 정해지지 않은 간선들의 번호 순으로 0인 경우 입력에 주어진 순서 대로 방향을 부여했다는 의미이고 1은 그 반대로 방향을 부여했다는 의미이다. 가능한 답이 여러가지인 경우 사전순으로 가장 빠른 것을 출력하라.

**입출력예**

**<입력>**

2

3 3 1

1 2

2 3

1 3

3 1

4 2 2

1 2

3 4

2 3

4 1

**<출력>**

Case #1

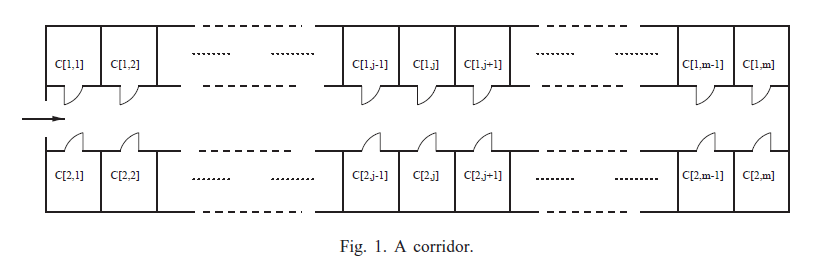
1

Case #2

01

**문제4**

호텔의 2m개의 방들이 아래 그림과 같은 복도 모양으로 배치되어 있다.



이 호텔 방들은 배열 2행 m열 배열 C로 나타낼 수 있다. 여기서 각 방은 C[i][j] (1≤i≤2, 1≤j≤m)로 나타낸다.

(i=i′ 이고 |j−j′|=1) 또는 (i≠i′ 이고 j=j′) 이면, 두 방 C[i][j] 와 C[i′][j′] 은 인접하다고 한다.

이 호텔에 묵을 계획인 예약자들이 예약 시스템을 통해서 방을 배정받으려고 한다.

예약자들의 총 수는 2m 이고, 예약자들의 전체 집합 S에 대해서, S는 집합 S1,S2,⋯,Sn 으로 나뉜다. 다시 말해서, S=S1∪⋯∪Sn.

또한 임의의 두 집합 Sh, Sk에 대해서, Sh∩Sk=∅ 이고, 모든 집합 Si 는 적어도 5명의 예약자를 포함한다. 다시 말해서, |Si|≥5,∀i.

예약 시스템은 방 하나에 한 사람씩 배정한다. 다시 말해서, 임의의 배정 A 는 예약자들의 집합에서 방들의 집합으로 일대일 대응이다.

또, 한 집합에 속한 예약자들은 모두 한 덩어리의 방들을 배정 받아야 한다. 한 덩어리의 방들이란 덩어리에 속한 어떤 방 두개에 대해서도, 덩어리에 속하고 인접한 방들을 통해서 이동이 가능하다는 의미이다.

각 예약자 a 는 스트레스 지수 wa를 가진다.

예약 시스템의 배정 A 에서, 인접한 두 방에 각각 배정된 예약자 a와 b가 서로 다른 집합에 속하면, 충돌 c 가 발생한다고 하고,

c의 페널티는 p(c)=wa+wb로 주어진다.

그러면 배정 A의 페널티 p(A)는 A에서 발생하는 모든 충돌들의 페널티 합으로 정의한다.

예약 시스템은 페널티 p(A)를 최소화하는 배정 A 를 찾아야 한다.

예를 들어, 호텔에 12개의 방이 있고, 12명의 예약자는 각각 6명씩 2개의 집합 S1과 S2로 나뉜다. 예약자들의 스트레스 지수는 모두 1이다.

방들을 2행 6열 배열 C로 나타낼 때, C의 홀수 열에 S1에 속한 예약자들을 짝수 열에 S2에 속한 예약자들을 배정한다면,

같은 행의 인접한 두 방 사이에 충돌이 발생하고, 각 충돌에 2의 페널티가 주어진다. 이 경우 모두 10개의 충돌이 생기고 총 20의 페널티가 주어진다.

(물론, 이 경우는 한 덩어리 조건도 만족하지 못한다.)

이 예에서는 2개의 충돌에서 4의 페널티가 생기는 배정이 최적이다.

- 제한시간: 전체 테스트 케이스는 67개 이하이며, 전체 수행 시간은 3초 이내. (Java 6초 이내)

제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,

그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,

이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.

C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,

제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다. ※ 언어별 기본 제공 소스코드 내용 참고

만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.

- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB

- 제출 제한 : 최대 20회

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.

파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 T 가 주어지고,

이후 차례로 T 개의 테스트 케이스가 주어진다. (1≤T≤67)

각 테스트 케이스의 첫 줄에는 두 정수 n과 m이 주어진다 (2≤n≤20,000, 5≤m≤50,000).

여기서, n은 예약자들의 집합의 개수이고, 호텔에 2m개 방이 있음을 나타낸다.

다음 n개 줄의 i번째 줄에는 집합 Si에 속한 예약자들의 수를 나타내는 정수 li 와 각각 Si의 예약자들의 스트레스 지수를 나타내는 li개의 정수 wij (j=1,…,li)가 주어진다 (5≤li≤100,000, 1≤wij≤10,000,000, ∑ni=1li=2m).

- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 190점)

주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,

각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.

ㆍ 그룹 1 ( 37점) : 이 그룹의 테스트 케이스에서는 모든 li 가 짝수이다.

ㆍ 그룹 2 ( 51점) : 이 그룹의 테스트 케이스에서는 모든 li 가 홀수이다

ㆍ 그룹 3 ( 102점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

\* 모든 테스트 케이스를 풀지 않고 일부분의 그룹에 속하는 테스트 케이스만을 푸는 경우에도 입력 받은 모든 케이스에 대해 (답이 틀릴지라도) 출력 양식에는 맞는 출력을 생성해야 점수가 반영되는 것이 보장된다.

\* 제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며, 그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며, 각 테스트 케이스마다 첫 줄에는 “Case #C”를 출력하여야 한다. 이때 C는 테스트 케이스의 번호이다.

예약시스템의 가능한 모든 배정 중 페널티가 가장 작은 배정을 찾아서 그 최소 페널티를 출력한다.

**입출력예**

**<입력>**

2

2 6

6 1 2 1 3 1 1

6 3 2 2 1 4 1

2 5

5 1 2 3 4 5

5 1 2 3 4 5

**<출력>**

Case #1

4

Case #2

8

**문제5**

N개의 변수 X1.,X2,…,XN이 있다. 처음에는 Xi들의 값에 대한 어떤 정보도 가지고 있지 않다. K개의 명령이 주어지는데, 명령은 두가지 종류이다. 첫번째 종류는 두 변수의 차이를 알려주는 것이다. 즉, Xi−Xj=d라는 정보를 준다. Xj−Xi=−d라는 정보도 주어진 것으로 간주한다. 두번째 종류는 두 변수의 차이를 묻는 것이다. 명령들은 하나하나 순서대로 주어지는 것으로 생각한다.

두번째 종류의 명령에 대한 대답은 다음의 3가지가 있다.

- 그 시점까지 주어진 모든 정보에 따라 두 변수의 차이가 유일하게 정해지면 그 값을 출력해야 한다.

- 그 시점까지 주어진 모든 정보에 따라 두 변수의 차이를 유추할 방법이 전혀 없다면 NC를 출력한다. NC의 의미는 Not Connected이다.

- 그 시점까지 주어진 모든 정보에 따라 두 변수의 차이를 유추할 수 있지만 하나로 정하는 것이 불가능하다면 CF를 출력한다. CF의 의미는 Conflict이다.

위의 조건을 자세히 설명하면 다음과 같다: 두번째 종류의 명령에서 주어진 두 변수를 Xi와 Xj라고 하자. Xi에서 Xj로의 연결은 변수의 열인데, 처음이 Xi이고 마지막이 Xj이며 열에서 인접한 모든 쌍이 그 시점까지 첫번째 종류의 명령으로 주어진 것이라야 한다. 인접한 쌍이 첫번째 종류의 명령에서 등장한 순서대로일 필요는 없다. Xi에서 Xj로의 연결은 여러 개가 존재할 수 있음에 주의하라. Xi에서 Xj로의 연결을 따라 계산한다는 것은 다음과 같이 정의된다. [Xi,Xk,…,Xj]가 연결이라고 하자. Xi−Xk의 값에 대한 정보가 여러 번 주어진 경우 어떤 것이든 사용할 수 있다. 동일한 연결에 대해서도 계산의 결과는 여러가지일 수가 있다. 선택한 정보가 Xi−Xk=d라면 d의 차이가 최초로 누적된다. Xk이후 Xj에 도달할 때까지 동일한 방식으로 차이를 누적한다. 동일한 연결에 대해서도 계산의 결과는 여러가지일 수가 있음에 주의하라.

Xi에서 Xj로의 가능한 모든 연결에 따라 Xi와 Xj의 차이의 모든 가능한 경우를 계산했을 때 전부 동일한 값이 나온다면 위의 첫번째 경우에 해당한다. Xi와 Xj의 가능한 연결이 존재하지 않는 경우 위의 두번째 경우에 해당한다. Xi와 Xj의 가능한 연결들을 이용해 Xi와 Xj의 차이를 계산했을 때 그 결과가 여러가지라면 위의 세번째 경우에 해당한다.

아래 입출력 예의 경우를 보자 입력은 아래와 같다.

1

5 9

1 2 3 2

1 3 2 -2

2 2 3

1 3 2 -3

2 2 3

1 1 4 7

1 4 5 8

2 5 2

2 5 1

첫 줄의 “1”은 테스트 케이스의 개수이다. 테스트케이스의 첫번째 줄의 “5 9”는 변수가 5개 있으며 명령이 9개 있다는 뜻이다. 다음 줄의 “1 2 3 2”는 첫번째 종류의 명령이며 X2−X3=2라는 정보를 준다. 그 다음 줄은 X3−X2=−2라는 정보를 준다. 두 정보는 동일한 것임을 알 수 있다. 다음 줄의 “2 2 3”은 X2−X3에 대한 명령이다. X2에서 X3로의 연결의 예는 [X2,X3],[X2,X3,X2,X3] 등이 있다. 모든 연결에서 계산한 값이 2로 동일함을 알 수 있다. 따라서 이 명령에 대한 답은 2이다. 다음 줄은 X3−X2=−3이라는 정보를 준다. 다음 명령은 “2 2 3”이다. X2에서 X3로의 가능한 연결인 [X2,X3]에 대해 X2−X3=2라는 정보를 사용한 계산과 X3−X2=−3(동등하게 X2−X3=3)이라는 정보를 사용한 계산의 결과가 서로 다르므로 이 명령에 대한 답은 CF이다. 다음 명령인 “2 5 2”의 경우 X5에서 X2로의 가능한 연결이 없으므로 답은 NC이다. 마지막 명령의 답은 -15이다.

위와 같이 정해진 명령들을 처리하는 프로그램을 작성하시오.

- 제한시간: 전체 테스트 케이스는 35개 이하이며, 전체 수행 시간은 5초 이내. (Java 8초 이내)

제한 시간을 초과하면 제출한 소스코드의 프로그램이 즉시 종료되며,

그때까지 출력한 내용이 파일에 저장되지 않아 점수가 제대로 반영되지 않을 수 있습니다.

그러나, 제한 시간을 초과하더라도 테스트 케이스를 1개 그룹 이상 통과하였다면 '부분 점수(0< 점수< 만점)'를 받을 수 있으며,

이를 위해서는, C / C++ 에서 "printf 함수" 사용할 경우, 프로그램 시작부분에서 "setbuf(stdout, NULL);"를 한번만 사용하십시오.

C++에서는 "setbuf(stdout, NULL);"와 "printf 함수" 대신 "cout"를 사용하고, Java에서는 "System.out.printIn"을 사용하시면,

제한 시간을 초과하더라도 '부분 점수'를 받을 수 있습니다. ※ 언어별 기본 제공 소스코드 내용 참고

만약, 제한 시간을 초과하지 않았는데도 '부분 점수'를 받았다면, 일부 테스트 케이스를 통과하지 못한 경우 입니다.

- 메모리 사용 제한 : heap, global, static 총계 256MB, stack 100MB

- 제출 제한 : 최대 10회 (참고: 동점자의 경우 최고 점수에 도달한 시간에 제출 횟수 한번마다 7분의 지연을 부가하여 시간을 계산하고 이 시간이 빠른 순서대로 등수를 결정함)

**메모리 사용 제한**

heap, global, static (총계) : 256MB  
stack : 100MB

**입력**

입력 파일에는 여러 테스트 케이스가 포함될 수 있다.

파일의 첫째 줄에 테스트 케이스의 개수를 나타내는 자연수 T 가 주어지고,

이후 차례로 T 개의 테스트 케이스가 주어진다. (1≤T≤35)

각 테스트 케이스의 첫 줄에는 변수의 개수 N과 명령의 개수 K가 주어진다. (1≤N≤100,000,1≤K≤200,000)

둘째 줄부터 K개의 각 줄에는 명령이 하나씩 주어진다. 명령의 첫 수가 1인 경우 차이를 알려주는 것이다. “1 i j d”와 같이 주어진 경우 Xi−Xj=d라는 의미이다. 동일한 쌍의 변수에 대해 여러 개의 1번 종류 명령이 주어질 수 있다. 명령의 첫 수가 2인 경우 차이를 묻는 것이다. 즉, “2 i j”와 같이 주어진 경우 Xi−Xj를 묻는 것이다. 두 종류 모두 i와 j가 같을 수도 있다. d의 절대값은 10,000 이하이다.

- 점수 : 각 제출에서 취득한 점수 중에서 최대 점수 (만점 200 점)

주어지는 테스트 케이스 데이터들의 그룹은 아래와 같으며,

각 그룹의 테스트 케이스를 모두 맞추었을 때 해당되는 부분 점수를 받을 수 있다.

ㆍ 그룹 1 (31 점) : 이 그룹의 테스트 케이스에서는 1≤N≤1,000,1≤K≤2,000이다. 모든 1번 종류 쿼리에서 차이 값은 0이다.

ㆍ 그룹 2 (28 점) : 이 그룹의 테스트 케이스에서는 1≤N≤1,000,1≤K≤2,000이다.

ㆍ 그룹 3 (20 점) : 이 그룹의 테스트 케이스에서는 모든 1번 종류 쿼리에서 차이 값은 0이다.

ㆍ 그룹 4 (121 점) : 이 그룹의 테스트 케이스에서는 원래의 조건 외에는 다른 제약조건이 없다.

\* 모든 테스트 케이스를 풀지 않고 일부분의 그룹에 속하는 테스트 케이스만을 푸는 경우에도 입력 받은 모든 케이스에 대해 (답이 틀릴지라도) 출력 양식에는 맞는 출력을 생성해야 점수가 반영되는 것이 보장된다.

**출력**

각 테스트 케이스의 답을 순서대로 표준출력으로 출력하여야 하며,  
각 테스트 케이스마다 첫 줄에는 “Case #C”를 출력하여야 한다. 이때 C는 테스트 케이스의 번호이다.  
그 다음 줄부터 각 쿼리에 대한 답을 한 줄에 출력한다. 

**입출력예**

**<입력>**

1

5 9

1 2 3 2

1 3 2 -2

2 2 3

1 3 2 -3

2 2 3

1 1 4 7

1 4 5 8

2 5 2

2 5 1

**<출력>**

Case #1

2

CF

NC

-15