

제 1 회
전국 대학생 프로그래밍 대회 연합 동아리 대회

Sponsored by



2011년 8월 13일

문제 A 부터 J 까지, 총 18 페이지

참가자를 위한 도움말

주의 사항

- 모든 입력은 **표준 입력**으로 주어지며, 모든 출력은 **표준 출력**으로 합니다. 표준 입/출력의 사용 방법에 대해서는 아래의 각 언어별 도움말을 참조해 주십시오.
- 모든 테스트 케이스에 대한 출력을 모아서 하실 필요 없이, 각 테스트 케이스를 처리할 때마다 출력해도 괜찮습니다. 역시 아래의 각 언어별 도움말을 참조해 주십시오.
- **리턴 코드**에 주의하십시오. 프로세스가 0이 아닌 다른 리턴 코드를 되돌리는 경우 **No - Runtime Error**를 받게 됩니다.
- 파일 이름을 영문자와 숫자로만 구성하십시오. 특히 **공백, 특수 문자, 한글**의 경우는 정상적인 채점을 보장할 수 없으며, 채점이 정상적으로 이루어지지 않은 경우 **“No - Compilation Error”**를 받게 됩니다.
- 대회 도중에 ACM-ICPC에서 정하는 규정에 따라 **인터넷 검색** 등의 행위를 삼가해 주시기 바랍니다.
- 문제에 대한 질의 사항은 채점 시스템인 PC²의 **Clarification** 기능을 사용해 주시기 바랍니다. 이 때 대답해 드리기 어려운 질문에 대해서는 **“No response, read problem statement”**로 대답될 수 있으므로 유의하십시오. 대답해 드리기 어려운 질문의 예는 아래와 같습니다:
 - 테스트 케이스의 수는 몇 개나 되나요: 출제자가 의도한 풀이로 넉넉한 시간 안에 나올 수 있을 정도로 주어집니다.
 - 시간제한은 얼마나 되나요: 위 질문을 참조하십시오.
 - 제 솔루션이 왜 WA 인가요: 제출하신 솔루션이 잘못된 답을 내었기 때문입니다. 테스트 케이스는 여러 차례에 걸쳐 검증되었으므로 틀릴 확률은 극히 낮습니다.
 - 문제에서 이리이러한 사항을 임의로 가정해도 되나요: 가정은 자유롭게 하시되 올바른 답만 내어 놓으시면 됩니다.
 - 제 시스템에서는 컴파일이 잘 되는데 왜 **No - Compilation Error**를 받나요: 대회에서 지정한 컴파일러 버전과 일치하지 않기 때문입니다. 특히 Visual C++ 6.0을 사용하시는 경우 빈번히 문제가 됩니다. 아래 언어별 도움말을 참조하십시오.

채점 결과에 대하여

Yes 제출하신 답안이 모든 테스트 데이터를 정해진 시간 안에 통과하여 정답으로 인정되었음을 의미합니다.

No - Compilation Error 제출하신 답안 프로그램을 컴파일하는 도중 오류가 발생하였음을 의미합니다.

No - Run-time Error 제출하신 답안 프로그램을 실행하는 도중 프로세스가 비정상적으로 종료되었음을 의미합니다.

No - Time-limit Exceeded 제출하신 답안 프로그램이 정해진 시간 안에 종료되지 않았음을 의미합니다. 시간 제한은 공개되지 않습니다.

No - Wrong Answer 제출하신 답안 프로그램이 테스트 데이터에 대해 생성한 출력이 출제자의 정답과 일치하지 않음을 의미합니다.

No – Excessive Output 제출하신 답안 프로그램이 지나치게 많은 양의 출력물을 생성하여 강제로 종료되었음을 의미합니다.

No – Output Format Error 제출하신 답안 프로그램이 정해진 출력 형식을 따르지 않았음을 의미합니다.

No – Other – Contact Staff 위에서 나열한 이유 이외의 문제로 채점이 거부되었음을 의미하며, Clarification 을 통해 대회 본부에 문의해 주십시오.

만약 여러 가지의 원인으로 인해 “Yes” 가 아닌 다른 결과를 얻으셨다면, 그 중 어떤 것도 결과가 될 수 있습니다. 예를 들어 답도 잘못되었고 출력 형식도 잘못된 코드를 제출하신 경우 대부분 “No – Output Format Error” 를 받으시게 되지만, 경우에 따라서 “No – Wrong Answer” 를 받을 수도 있습니다.

언어별 도움말

C/C++

다음 두 예시 코드는 먼저 테스트 케이스의 개수를 입력받고, 각 테스트 케이스에 대해 두 정수를 입력받아 더한 값을 출력합니다.

```
1 // Written in C
  #include <stdio.h>

  int main()
  {
6     int t;
      scanf("%d", &t);
      for (int i = 0; i < t; i++)
      {
11         int a, b;
          scanf("%d %d", &a, &b);
          printf("%d\n", a + b);
      }
      return 0;
  }
```

```
// Written in C++
#include <iostream>

using namespace std;
5
int main()
{
    int t;
    cin >> t;
10    for (int i = 0; i < t; i++)
    {
```

```
15         int a, b;  
            cin >> a >> b;  
            cout << a + b << endl;  
        }  
        return 0;  
    }
```

Java


다음 예시 코드는 먼저 테스트 케이스의 개수를 입력받고, 각 테스트 케이스에 대해 두 정수를 입력받아 더한 값을 출력합니다.

```
import java.util.*;  
  
3 public class Adder  
    {  
        public static void main(String[] args)  
        {  
            Scanner sc = new Scanner(System.in);  
8            int t = sc.nextInt();  
            for (int i = 0; i < t; i++)  
            {  
                int a = sc.nextInt();  
                int b = sc.nextInt();  
13                System.out.println(a + b);  
            }  
        }  
    }
```

언어의 규칙 상, 소스 파일의 이름과 클래스 이름이 **일치해야** 함에 유의하십시오.

다음 장부터 문제가 주어집니다.

Problem A. 사각형 그리기

 AdbyMe, Inc. 의 인턴인 A.I. 는 웹 브라우저에 직사각형을 그리는 코드를 작성해야 한다. 웹 브라우저는 직사각형 모양의 뷰포트를 가지고 있고, 그려지는 직사각형의 네 변은 반드시 그 뷰포트의 두 축에 평행해야 한다.

한편, A.I. 는 코드를 작성하던 중 그릴 직사각형의 네 꼭지점 중 어느 것이든 세 개의 좌표를 알고 있다면 나머지 점의 위치는 유일하게 결정됨을 알아내었다 (네 점 중 어떤 두 개의 좌표를 알아낸 경우는 때에 따라 직사각형을 결정하지 못할 수도 있다.)

A.I. 는 LiBe에게 이를 이번 대회 문제로 출제할 것을 제안하였다.

직사각형을 이루는 네 점 중 임의의 세 점의 좌표가 주어졌을 때, 나머지 한 개의 점의 좌표를 찾는 프로그램을 작성하라.

Input

입력은 T 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 T 가 주어진다.

각 테스트 케이스는 공백 하나로 구분되는 두 개씩의 정수로 구성된 세 행으로 이뤄지며, 각각 임의의 세 점의 x 와 y 좌표이다. 브라우저 뷰포트의 맨 왼쪽 위 픽셀의 좌표는 $(1, 1)$ 이고, 맨 오른쪽 아래 픽셀의 좌표는 $(1000, 1000)$ 이다. 모든 좌표는 뷰포트 안에 위치하며, 각 점의 위치는 모두 다르다.

Output

각 테스트 케이스에 대해 한 행에 하나씩 좌표가 주어지지 않은 나머지 한 점의 x 와 y 좌표를 공백 하나로 구분하여 출력한다.

Sample input and output

Standard Input	Standard Output
2	7 7
5 5	30 10
5 7	
7 5	
30 20	
10 10	
10 20	

Problem B. Wooksin-ness of A Graph

At the headquarters of algospot.com, Toivoa “the chairman” has been pestering Astein “the slave” for a new graph problem for so long. So Astein came up with the following problem.

Wooksin-ness(옥신함) of an undirected graph $G(V, E)$ is defined as the minimum number of additional edges in order to have a cycle in the graph. Stated formally, you can write it as:

$$\min |E' - E| \text{ s.t. } E' \supseteq E \text{ and } G(V, E') \text{ has at least one cycle}$$

However, loops (edges that start and end at the same vertex) and multiple edges between a single pair of vertices are not allowed.

Write a program that calculates Wooksin-ness of given graph.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case contains two integers V and E ($1 \leq V \leq 100, 0 \leq E \leq 1,000$), where V represents the number of vertices in the graph, and E represents the number of edges. The vertices are numbered from 0 to $V - 1$. The following E lines will each contain two integers, which are the number of two vertices connected by an edge.

There will be no loops in the input data. There will be at most one edge between a pair of vertices.

Output

Print exactly one line for each test case. The line should contain an integer indicating the minimum number of additional edges we need to add to the graph to get a cycle. If this is impossible, print -1 instead.

Sample input and output

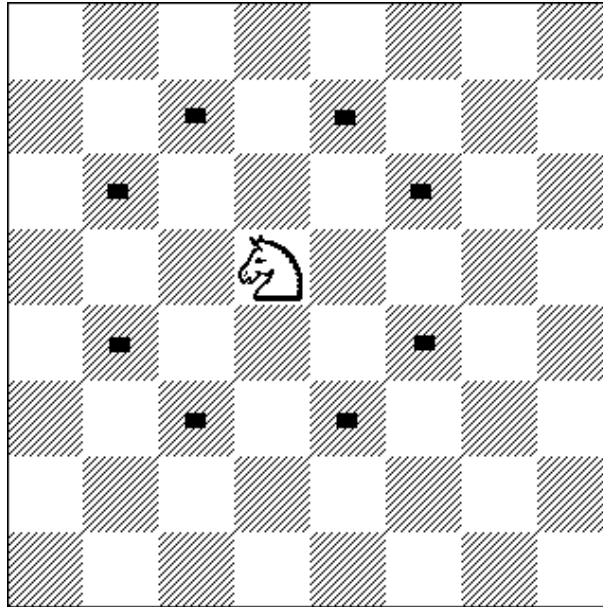
Standard Input	Standard Output
2	1
3 2	0
0 1	
2 0	
3 3	
0 1	
1 2	
2 0	

Problem C. Run Run Run

In the world of Altertania, there is a large chessboard with N^2 cells. The board has N rows and N columns; rows are numbered from 1 to N starting from the top, and columns are numbered from 1 to N starting from the left. Each cell (r, c) are located by its row number r and column number c .

With this ample real estate, chess pieces do not fight anymore and live peacefully together.

A knight wants to visit his friend rook for his birthday. He still has his horse with him, so he can jump around like he used to. See the following diagram for the jumps he can make.



However, he has a problem: his horse is old and lazy, so it will refuse to make more than a single move each day. Alas! At this rate, he cannot make it before rook's birthday.

Don't despair, all is not lost. Some cells in this chessboard are converted to vegetable patches. At the start of each day, the knight steps into a cell by a single move. If that cell has vegetable patch, he can feed his horse some carrots and the horse will agree to make a few more moves that day. The number of additional moves the horse can make is equal to the nutrition value of the carrot, which is an integer between 1 and 5 (inclusive). The horse will not eat carrots twice in a single day.

Write a program to calculate the minimal days the knight can reach his friend.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case contains two integers N ($4 \leq N \leq 1,000$) and V ($0 \leq V \leq 10,000$). The second line contains four integers R_K, C_K, R_R, C_R ($1 \leq R_K, C_K, R_R, C_R \leq N$). The knight is currently at (R_K, C_K) , and the rook lives at (R_R, C_R) . V lines will follow: each line will contain three integers R_i, C_i and V_i ($1 \leq R_i, C_i \leq N, 1 \leq V_i \leq 5$). V_i is the nutrition value of carrots from vegetable patch at (R_i, C_i) .

There will be at most one vegetable patch in a single cell. The knight's starting position will not be a vegetable patch nor the rook's position.

Output

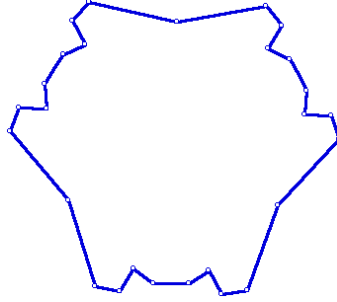
Print exactly one line for each test case. The line should contain an integer indicating the number of days the knight needs to reach his friend.

Sample input and output

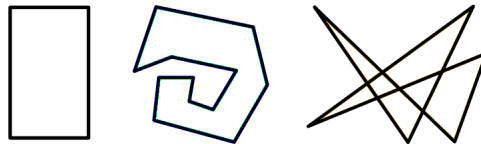
Standard Input	Standard Output
3	2
4 3	2
1 1 4 3	5
1 4 2	
4 2 3	
4 4 2	
5 4	
1 1 5 5	
2 4 2	
3 3 1	
3 4 2	
4 4 2	
4 1	
1 1 1 4	
1 4 5	

Problem D. 재하의 장난감

얼마 전에 태어난 깜짝 스타 재하는 순식간에 무럭무럭 자랐다. 누구를 닮았는지 호기심이 유달리 강해 주위 사물들에 손을 뻗어서 만지작거리곤 하는데, 재하의 아버지와 어머니는 이러한 재하를 위해서 장난감을 한 개 사다 주었다. 이 장난감은 선분과 선분이 환형으로 연결된 모양이라, 아이가 입에 넣는 것이 쉽지 않도록 되어 있다.



재하는 장난기가 발동하여 다음과 같은 다양한 모양으로 장난감을 이리저리 흐트러트리기 시작했다.



엄밀하게 말하자면, 이 장난감은 $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n), (x_1, y_1)$ 의 인접한 점들을 선분으로 이은 교차점이나 교차선이 있을 수 있는 다각형으로 정의된다.

이 장난감이 만들어낸 넓이가 0이 넘는 영역의 수를 구하라. 단, 이 장난감 가장 바깥에 원래부터 존재하고 있었던 무한대 넓이의 영역은 세지 않는 것으로 한다.

Input

입력은 T 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 T 가 주어진다.

각 테스트 케이스의 첫 행에는 장난감을 이루는 꼭지점의 수 N ($3 \leq N \leq 100$) 이 주어진다. 그 후 N 줄에 각 꼭지점의 좌표 x_i, y_i ($0 \leq x_i, y_i \leq 100$) 가 주어진다. 각 테스트 케이스 안에서, 같은 좌표의 꼭지점은 입력으로 들어오지 않는다.

Output




각 테스트 케이스마다 한 행에 하나씩 문제에서 설명한 영역의 수를 출력한다.


Sample input and output

Standard Input	Standard Output
2	2
4	1
0 1	
1 0	
1 1	
0 0	
6	
0 0	
1 0	
1 1	
2 1	
2 0	
3 0	

Problem E. 화이트 칼라

전미 최고의 사기꾼. 안 해본 도둑질, 안 해본 사기가 없는 닐 카프리는 오늘 저녁 세계 최고의 미술품 중 하나인 “뮤직박스”를 훔칠 예정이다.

오늘 아침, 이 정보를 입수한  AdbyMe, Inc. 는 그를 검거하기 위한 작전을 세우고 있다.  AdbyMe, Inc. 는 그가 현재 어느 도시에 있는지, 그리고 뮤직박스가 어느 도시에 있는지 파악했고, 그를 잡기 위해 직원들을 배치할 것이다.  AdbyMe, Inc. 가 입수한 정보에 의하면 닐 카프리는 매우 급한 성격(?)이고, 그의 성격으로 볼 때, 현재 위치에서 뮤직박스가 있는 곳까지 최단 경로로 이동할 것이다.

그래서  AdbyMe, Inc. 는 현재 닐 카프리가 있는 도시와 뮤직박스가 있는 도시, 그리고 그가 이동할 때 거쳐 갈 가능성이 있는 모든 도시에 직원들을 배치하려고 한다. 지도를 보고 직원들을 배치해야 되는 도시를 모두 골라내자.


Input

입력은 T 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 T 가 주어진다.

각 테스트 케이스의 첫 행에는 도시의 수 N ($2 \leq N \leq 1,000$), 도시 간에 연결된 길의 수 M ($1 \leq M \leq 50,000$) 이 주어진다. 그 다음 M 행에 연결된 도시의 번호 A_i 와 B_i 가 주어진다. 모든 길은 그 길이가 같은 A_i 에서 B_i 로 이동하는 일방통행 길이다. ($1 \leq A_i, B_i \leq N, A_i \neq B_i$)

닐 카프리는 현재 1번 도시에 위치해 있고, 뮤직박스는 N 번 도시에 위치해 있다. 1번 도시에서 N 번 도시로 이동 가능한 경로는 반드시 하나 이상 존재한다.

Output

각 테스트 케이스에 대해 한 행에 하나씩  AdbyMe, Inc. 가 직원들을 배치해야하는 도시의 번호를 오름차순으로 출력한다.

Sample input and output

Standard Input	Standard Output
2	1 3 4
4 5	1 2 3 5
1 2	
2 1	
1 3	
3 4	
4 3	
5 6	
1 2	
1 3	
2 5	
3 4	
3 5	
4 5	

Problem F. Jubeat

Recently, Nana is spending a lot of time playing “Jubeat”, the latest arcade music game from Konami. The game consists of a 4×4 grid. When a note pops up in some of the cells, the player must push that cell following the rhythm of the music.

For each note pushed, the player is awarded points. If a song contains N notes, a player will receive $900,000/N$ points if the timing of the push is perfect. For notes where the timing was less perfect, she can receive either 70%, 30%, or 0% of this score. After the song is over, the sum of all the points will be rounded down to the nearest integer.

Nana just played a song and received a score of S : however he doesn't know the number of notes N in the song. However, he suspects N lies in an interval $[a, b]$. Write a program that calculates the sum of all possible N s in that range.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input. Each test case consists of a single line with three integers: S ($0 \leq S \leq 900,000$), a and b ($1 \leq a \leq b \leq 10^9$).

Output

Print exactly one line for each test case. The line should contain an integer indicating the sum of all possible N in the given range.

Sample input and output

Standard Input	Standard Output
2 128571 1 10 450000 11 20	7 155

Problem G. 올림픽

4년마다 열리는 올림픽에서는 순위에 따라 선수들에게 금, 은, 동메달을 수여한다. 따라서 선수들의 국적을 바탕으로 각 국가가 획득한 메달의 수를 셀 수 있다. 공식적으로는 이를 통해 각 국가의 순위를 가리는 방법이 없지만, 본질적으로 순위를 매기는 편이 더 재미있고 열기를 고취시킬 수 있기 때문에 순위를 매기는 방법들이 고안되었다. 이 문제에서는 널리 쓰이는 두 가지 방법을 생각해 본다.

- 국가 x 의 등수는 (국가 x 보다 좋은 성적을 거둔 국가의 수 + 1)로 정의한다.
- 방법 1. 메달의 수를 종류에 관계 없이 세어 많은 국가가 적은 국가보다 좋은 성적이라 정의한다.
- 방법 2. 금메달이 많은 국가가 적은 국가보다 좋은 성적이라 정의한다. 단, 같을 경우는 은메달이 많은 국가를 좋은 성적으로 정의하며, 은메달도 같은 경우는 동메달이 많은 국가를 좋은 성적으로 정의한다.

N 개의 국가가 경쟁 중이며, 앞으로 K 개의 경기가 남았다. 앞선 경기는 반드시 그렇지 않았지만 남은 경기 모두는 금, 은, 동메달이 서로 다른 국가에 하나씩 주어지는 종목이다. 지금까지 그들이 획득한 메달의 수가 주어질 때, 앞으로 남은 K 경기의 결과에 따른 최선의 등수를 두 방법 각각에 대해 구하는 프로그램을 작성하라.

Input

입력은 T 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 T 가 주어진다.

각 테스트 케이스의 첫 행에는 두 정수 N ($3 \leq N \leq 100$), K ($0 \leq K \leq 1,000$)가 공백으로 구분되어 주어진다. 다음 N 행에 걸쳐 한 행에 하나씩 각 국가가 획득한 금, 은, 동메달의 수가 공백으로 구분되어 주어진다. 각 메달의 수는 1,000을 넘지 않는다.

Output

각 테스트 케이스에 대해 두 행에 나누어 입력에서 주어진 순서에 따라 각 국가의 방법 1에 따른 최선의 등수를 공백으로 구분하여 출력하고, 다음 행에 마찬가지로 방법 2에 따른 최선의 등수를 출력한다.

Sample input and output

Standard Input	Standard Output
2	1 2 2 2
4 1	1 2 2 2
1 1 1	3 1 1 1
0 1 0	1 1 1 1
0 0 1	
0 0 1	
4 1	
0 0 0	
1 0 0	
0 1 0	
0 0 1	

Notes

- 첫 번째 테스트 케이스: 첫 번째 국가가 남은 경기에서 메달을 획득하지 못하더라도 다른 국가가 어떤 방법으로도 더 잘 할 수 없다.

- 두 번째 테스트 케이스: 방법 1에 의하면, 첫 번째 국가를 제외한 모든 국가는 적어도 공동 1위가 될 수 있다. 그러나 첫 번째 국가의 경우 세 국가 중 적어도 둘은 메달을 획득하므로 첫 번째 국가보다 앞설 수밖에 없다. 방법 2에 의하면 어느 국가도 자신이 금메달을 획득하고 나머지 국가들이 적당히 은, 동메달을 나누어 가지면 1위가 될 수 있다.

Problem H. 맑은 하늘 프로젝트

2차원 세계를 지배하는 J. 왕국에 왕세자 H. 가 태어났습니다! 국왕인 M. 은 이를 기념하여 2주일간의 축제 기간을 선포했습니다. 탄생 축제기간의 마지막 날에는 거대한 축포를 쏘아올려 J. 왕국의 모든 국민들이 그 축포를 볼 수 있도록 할 것입니다.

하지만 정작 축포를 발사할 전일이 되자, 국왕인 M. 은 고민이 생겼습니다. 그 고민의 원인은 다름이 아니라 J. 왕국의 위에 떠있는 수많은 구름들 때문입니다. 축포가 구름에 가려서 안 보이더라도 하면 불운의 조짐으로 여겨질 것이며 축제를 위해 수도를 찾은 사람들이 크게 실망할 것입니다.

고민 끝에 M. 왕은 대학생 시절에 자취방 제습제 대응으로 만들었던 “Nerdull-Nerdull-Han 레이저포(이하 너덜포)”를 꺼냈습니다. 희귀 금속인 Nerdull로 만든 너덜포는 고출력의 매우 뜨거운 광선을 발사하기 때문에 수증기 덩어리인 구름따위는 스치기만 해도 사라져 버립니다 (하지만 맞지 않으면 아무 것도 아니라는 사실에 주의하세요.) M. 은 이 문제를 다음과 같이 단순화할 수 있다는 것을 직감했습니다:

1. 하늘에 존재하는 N 개의 구름은 x 축에 평행한 선분이다. 모든 구름은 $x > 0$ 인 영역에 존재한다.
2. 지면은 $y = 0$ 인 직선으로 생각하며, 너덜포는 지면 위의 한 지점으로부터 연직 상방으로 발사한다.
3. 내구도의 문제 때문에 너덜포는 최대 K 회까지만 발사가 가능하다.
4. 너덜포를 발사하면 고출력 광선을 따라 그 위에 존재하는 모든 구름이 사라진다. 이 때의 필요한 전기 비용은 다음과 같다:

$$[\text{너덜포의 } x \text{ 좌표값}(= \text{발전소로부터 전력을 끌어오는 비용})] \times [\text{사라지는 구름의 수}]$$

M. 은 이 문제를 쉽게 풀수 있다는 것을 직감했지만, 암산 도중에 H. 가 울기 시작했기 때문에 당신에게 계산을 맡겼습니다. J. 왕국은 축제 기간동안 예산을 많이 사용하였기 때문에 구름을 제거하는 비용은 가능한 최소로 줄이고 싶어합니다. 가능한 적은 예산을 계산해서 M. 국왕 전하의 신임을 얻으세요.

모든 구름을 제거하는 일이 불가능한 경우는 없습니다. 왕세자 H. 는 신의 축복 아래서 태어났거든요!

Input

입력의 첫 행에는 평행 세계의 갯수 T 가 주어집니다.

각 평행 세계마다 왕국 위에 떠 있는 구름의 숫자와 너덜포의 내구도를 나타내는 두 정수 N ($1 \leq N \leq 500$) 과 K ($1 \leq K \leq N$)가 공백으로 구분되어 주어집니다. 그 다음 N 개의 행은 구름의 위치를 나타냅니다. 각 행마다 i 번째 구름의 양 끝 x 좌표인 두 정수 $Left_i$ 와 $Right_i$ 가 공백으로 구분되어 주어집니다. ($1 \leq Left_i \leq Right_i \leq 10,000$)

Output

각 평행세계의 J. 왕국의 최소 구름 제거 비용을 한 행에 하나씩 출력합니다.

Sample input and output

Standard Input	Standard Output
3	9
3 1	7
1 3	6
2 5	
3 5	
3 2	
1 3	
2 5	
3 5	
3 3	
1 3	
2 5	
3 5	

Problem I. Turn off the Lights

Eunjin has a rectangular-shaped board of light bulbs. It is consisted of R rows and C columns, therefore there are RC bulbs in total. A light bulb can have either of two states: turned on (often denoted by “1”) or turned off (often denoted by “0”).

Wonha, a mischievous boyfriend, messed up her light board by turning on some bulbs. The other bulbs are currently turned off. After Eunjin noticed his mischief, she decided to turn all the lights off using minimum number of operations. The only operation she can do is to reverse the state of a consecutive sequence of light bulbs in a row or column.

You must write a program that computes the smallest possible number of operations needed for her, to make all the bulbs turned off. It is allowed that some light bulbs to be turned on during the whole process, but after a sequence of appropriate operations, all the bulbs must be turned off.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test cases contains two integers R and C , which denotes the number of rows and the number of columns, respectively, in the light board. Each of the following R rows contains a string of C digits “1” or “0”. Each digit denotes the initial state of each light bulb after Wonha changed the state of some light bulbs: “1” means the bulb is turned on, and “0” means the bulb is turned off. You may assume that both R and C are not greater than 15.

Output

Print exactly one line for each test case. The line should contain an integer indicating the smallest number of operations needed.

Sample input and output

Standard Input	Standard Output
1 5 8 00100000 00100000 11011000 00100000 00111100	3

Problem J. Lottery Games

You live in a lively town named Lottery Vegas where lots of different kinds of lottery games are available for you to play. Next to your house, you found an interesting lottery game that is called Double ticket winner ainu7 for the win, named after a really famous Miku-admirer.

The ainu7 lottery game consists of P different lottery tickets. The i -th ticket contains numbers between 1 and N_i , inclusive, and you are to pick M_i numbers out of them. The ainu7 lottery game seller also picks M_i numbers while you are picking. For each lottery ticket, you win if you and the seller have at least K_i numbers in common with the seller. You can assume that the seller picks the numbers at random, regardless of what you pick.

You are curious which of the P tickets gives you the highest winning odds. If there are multiple such tickets with the same highest winning odds, you want to know them all.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case contains a single integer P ($2 \leq P \leq 100$), the number of lottery tickets. Following P lines contains three numbers each: N_i , M_i , and K_i where $3 \leq N_i \leq 50$, $1 \leq M_i \leq N_i$, and $1 \leq K_i \leq M_i$.

Output

For each test case, you must output a single line of integer(s). It must contain the lottery game number(s) with highest winning odds. If there are multiple, you must sort them, and the game number is 1-based.

Sample input and output

Standard Input	Standard Output
2	4
4	1 2 3 4 5 6
3 1 1	
8 2 1	
8 4 2	
8 3 1	
7	
8 7 1	
8 7 2	
8 7 3	
8 7 4	
8 7 5	
8 7 6	
8 7 7	