제 2회
전국 대학생 프로그래밍 대회 연합 동아리 대회

후원

출제 및 채점

2012년 8월 18일

문제 A 부터 J 까지, 총 20 페이지

Problem A. 이상한 드래프트

2150년, Kureyo Baseball Organization(이하 KBO)은 100번째 구단의 창단을 승인하였다. 그 전까지는 지난 시즌의 순위에 따라 지그재그식으로 한 선수씩 선발하는 식이었지만, 구단이 매우 많아지면서 드래프트가 너무 오랜 시간이 걸리게 되자 KBO는 다음과 같은 새로운 드래프트 룰을 발표하였다.

- 드래프트를 신청한 선수 N 명은 출신 학교, 대회 입상 성적, 신체 조건 등의 다양한 요소를 고려한 임의의 순서로 정렬된다.
- 각 선수는 자신의 고유 수비 위치를 나타내는 정수인 P_i $(1 \leq P_i \leq 9)$ 와 수비 능력을 나타내는 정수인 A_i 로 표현된다.
- 각 구단은 자신의 드래프트 차례가 오면 선수들의 목록에서 연속된 K 명을 뽑고 해당 선수들을 목록에서 지운다.
- 이 때 선택된 K 명 안에는 각 아홉 가지의 수비 위치에 대해 해당 위치를 수비할 수 있는 선수가 반드시 적어도 한 명씩 포함되어야 한다.

당신은 전년도에 꼴찌를 한 '쌍둥이들' 의 감독으로 이번 드래프트에서 가장 먼저 K 명의 선수를 선택하게 되었다. 쌍둥이들의 고질적 수비 불안을 해소하기 위해 다음 시즌에는 수비 능력 위주로 팀을 리빌딩할 예정이기 때문에, 이번 드래프트는 매우 중요하다.

N 명의 선수들의 수비 위치와 수비 능력이 드래프트 신청자 목록에 주어진 순서대로 주어질 때, 각 수비 위치에서 가장 뛰어난 선수들의 능력의 총 합이 최대가 되도록 K 명을 뽑는 프로그램을 작성하라. 단, 드래프트가 불가능한 경우는 없다고 가정한다.

Input

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

각 테스트 케이스 첫 줄에는 드래프트를 신청한 선수의 수 N 과 드래프트로 한 번에 뽑을 수 있는 사람의 수 K 가 공백으로 구분되어 주어진다. $(9 \leq K \leq N \leq 100,000)$ 그 후 N 줄에 각 선수들의 수비 위치 P_i $(1 \leq P_i \leq 9)$ 와 수비 능력 A_i $(1 \leq A_i \leq 100)$ 가 공백으로 구분되어 주어진다.

Output

각 테스트 케이스마다 한 줄에 하나씩 이번 드래프트에서 얻을 수 있는 각 수비 위치별로 가장 뛰어난 선수들의 수비 능력의 총 합의 최대값을 출력한다.

Sample input and output

Standard Input	Standard Output
2 10 9 1 1 2 1 3 1 4 1 5 1 6 1 7 1 8 1 9 1 1 100 17 12 1 1 2 1 3 1 4 1 5 1 6 1 7 1 8 1 9 1 8 2 7 3 6 4 5 5 4 6 3 7 2 8 1 9	108 45

Problem B. K-ary Huffman Encoding

Huffman 나라의 문자는 N 개의 문자로 이루어져 있다. 0과 1로 이루어진 이진 인코딩이 발달한 대한민국과는 달리, Huffman 나라에서는 0부터 $K-1$까지의 숫자로 이루어진 K 진법 인코딩이 발달했다.

우리는 Huffman 나라의 언어를 K 진법으로 인코딩하려 한다. 이 때 다음 조건을 만족해야 한다.

1. N 개의 각 문자에 K 진법 문자열을 하나씩 배정해야 한다.
2. 배정된 K 진법 문자열들이 서로의 접두사(prefix)일 수 없다.

예를 들어 아래의 표는 `!`, `@`, `#`, `+` 4개의 문자를 3진법으로 올바르게 인코딩한 예이다.

문자	인코딩
!	012
@	120
#	201
+	210

4개의 문자 각각에 서로의 접두사가 아닌 3진법 문자열을 할당했음을 확인해 볼 수 있다. 반면,

문자	인코딩
!	013
@	120
#	201
+	210

은 `!`에 `013`을 배정해서 첫 번째 조건을 만족하지 않고,

문자	인코딩
!	012
@	120
#	201
+	20

은 `+`이 `#`의 접두사이기 때문에 두 번째 조건을 만족하지 않는다.

Huffman 나라 언어 문자열이 하나 주어졌을 때, 이 문자열을 K 진법으로 인코딩한 결과를 가장 짧게 만들면 길이가 어떻게 될까? 예를 들어 "`!!!@@@@#####++++++`" 과 같이 `!`가 3번, `@`가 4번, `#`가 5번, `+`가 6번 문자열에 나타났다면, 첫 번째 표에서 제시한 인코딩을 적용한 경우 총 길이는 54가 된다 ($3\times3+4\times3+5\times3+6\times3=54$).

Input

입력은 T 개의 테스트 케이스로 구성된다. 입력의 첫 줄에는 T가 주어진다.

각 테스트 케이스 첫 줄에는 두 정수 N ($2 \le N \le 10000$), K ($2 \le K \le 10000$)가 공백으로 구분되어 주어진다. N은 Huffman 나라의 문자의 수이고 K는 인코딩할 진법을 나타낸다. 다음 줄에는 각 문자가 문자열에 몇 번이나 나타나는지를 의미하는 N 개의 정수 C_i ($0 \le C_i \le 100000$) 가 공백으로 구분되어 주어진다.

Output

각 테스트 케이스마다 한 줄에 하나씩 주어진 문자열의 가능한 최소 K 진법 인코딩의 길이를 출력한다.

Sample input and output

Standard Input	Standard Output
2 4 2 0 1 2 3 4 2 0 1 2 2	10 9

Notes

두 테스트 케이스 모두

문자	인코딩
3	0
2	10
1	110
0	111

로 할당하면 최소의 길이를 얻을 수 있다. 첫 번째 케이스의 경우 $3 + 4 + 3 = 10$ 길이로 문자열을 표현할 수 있으며, 두 번째 케이스의 경우 $3 + 4 + 2 = 9$ 길이로 문자열을 표현할 수 있다.

Problem C. Talk jail

How have you been?

I want to play a game.

This game will be played on a smartphone, on which you spend most of your time.

On the 'Talk.'

Are you happy because you got unlimited 3G?

Tonight, we shall see how far you go in order not to turn your phone off.

Will you get a push or turn it off?

Your choice.

— Jigsaw

Jigsaw invited N people to the "Talk jail" room. Here is a brief description of how Talk room works. Everyone can send messages, and all messages are delivered to everyone in the room, including the sender. However, a new message arrived on someone's smartphone is not displayed immediately if the chat room window is not open. Instead, it is kept in the **unread** status. As soon as one **opens** the Talk jail room window, the status of all messages that have been arrived so far changes to the **read** status. Also, while the chat room window is open, newly arrived messages will be displayed and marked as read immediately. One can close the chat room window, but the messages will still arrive. One can send a message only when he/she has the chat room window open.

Moreover, when one opens the chat room window, one can see 3 pieces of information on each message: the sent time, the sender, and the number of people who have not read the message yet.

This simple chatting service can actually serve as a jail as follows. When a lot of people send messages in the Talk jail room, everyone's phone will keep beeping continuously, which makes it impossible for him/her to manage ordinary life. Even if someone tries to leave the room, Jigsaw invites him/her back immediately: so it is impossible to leave the room. As a result, people will complain in the room which yields more messages and more beeps…

Jigsaw will never send any message to the room but will keep inviting leavers back to the room. Being so frustrated, people eventually choose one or the other: (A) just suck it up and live with it or (B) ignore the smartphone entirely and live free.

Jigsaw just opened the chat room window and checked out the M messages that have been sent ever since the room was created. Based on this, he can infer that certain people have not read certain messages with certainty. That is, for each message, he may be able to infer whether someone has read the message or not (sometimes, such inference may not be possible). He calls those who have not read the message for sure, "the wretched addict."

For each message, your job is to figure out how many people Jigsaw can identify as "the wretched addicts" because you can infer that they have not read the message.

Input

The input consists of T test cases. The first line of the input contains T.

Each test case starts with two integers N ($1 \leq N \leq 60,000$) and M ($1 \leq M \leq 60,000$), separated by a whitespace. The following M lines contain three integers each: t_i, p_i, and c_i where t_i ($0 \leq t_i < 2^{31}$) denotes the sent time, p_i ($1 \leq p_i \leq N$) denotes the sender, and c_i ($0 \leq c_i < N$) denotes the number of people who have not read the message i. No two messages have the same value of t_i. The input is consistent and valid.

Output

For each test case, you must output "#Test case number" in the first line. You must output the number of people who are certain that have not read the i-th message yet in a separate line.

Sample input and output

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

Notes

Here comes the explanation of the first test case.

Everybody read the first message (from the input data). Hence, the answer is 0.

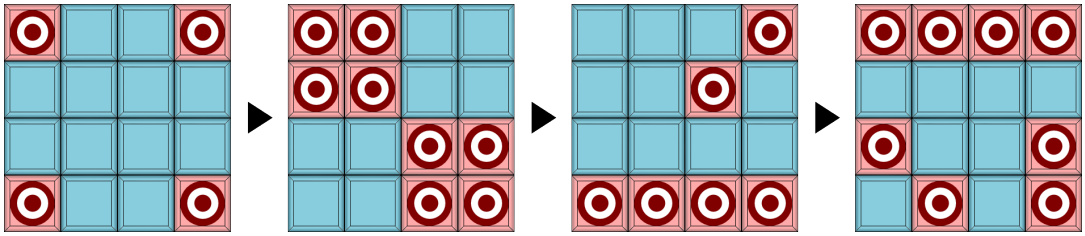
There is one person who has not read the second message yet. Person 2 must have read the message (as he sent it). Either Person 1 or 3 could have read the message, and Jigsaw cannot identify exactly who has not read it. The answer is 0.

The third message is again read by Person 2. However, both Person 1 and Person 3 have not read it yet, and thus Jigsaw can be certain that those two have not read. The answer is 2.

Problem D. N-beat

N-beat is a popular rhythm arcade game, in which you push buttons arranged in a $x \times y$ grid. The buttons light up when you have to push them; you have to push them with precise timing to get scores.

One game of N-beat consists of one or more screens. A **screen** is a configuration of lit buttons which you have to push.



For example, the picture above shows 4 successive **screen**s for a N-beat machine with 4×4 grid. Here, boxes with the target mark are the lit buttons you have to press.

A sequence of screens in a game is called a **transcription**.

Now, Jaeha Koo, a genius composer, just made a new song for N-beat. You're going to make transcription for this song.

There are B beats in this song, so this transcription will consist of B **screen**s. Therefore, without any restriction, there are a total of $(2^{xy})^B$ possible transcriptions.

But **transcription**s differ in difficulty. Generally, you can push one button with one finger, so it's harder when a single screen has more lit buttons. Especially, if a screen has more than 10 lit buttons, it's very difficult to push them successfully. Also, adjacent screens having a lot of lit buttons can also can make the game difficult. For example, two successive screens with 7 and 6 lit buttons each might be more difficult than two screens with 2 and 8 lit buttons each.

As a generous transcriptor, you decided to limit the number of lit buttons. The limit is given as three nonnegative integers: p_1, p_2, and p_3. p_1 is the maximum number of lit buttons you can have in any one screen. p_2 is the maximum number of lit buttons you can have in any two consecutive screens. Also, as you guessed, p_3 is the maximum number of lit buttons you can have in any three consecutive screens. In the picture above, 4 screens each have 4, 8, 6, 8 of turned-on buttons, so this transcription will only be valid if $p_1 \geq 8$, $p_2 \geq 14$, and $p_3 \geq 22$.

Your task is to calculate the number of possible transcriptions given x, y, B, p_1, p_2 and p_3. As the result can be quite huge, just calculate the answer mod $10^9 + 7$.

Input

The input consists of T test cases. The first line of the input contains T.

Each test case starts with 6 integers x, y ($1 \leq x, y \leq 1000$), B ($1 \leq B \leq 10^9$), p_1 ($0 \leq p_1 \leq 10$), p_2 ($0 \leq p_2 \leq 20$), p_3 ($0 \leq p_3 \leq 30$) separated by a whitespace.

Output

For each test case, print the number of possible transcriptions mod $10^9 + 7$, in a single line.

Sample input and output

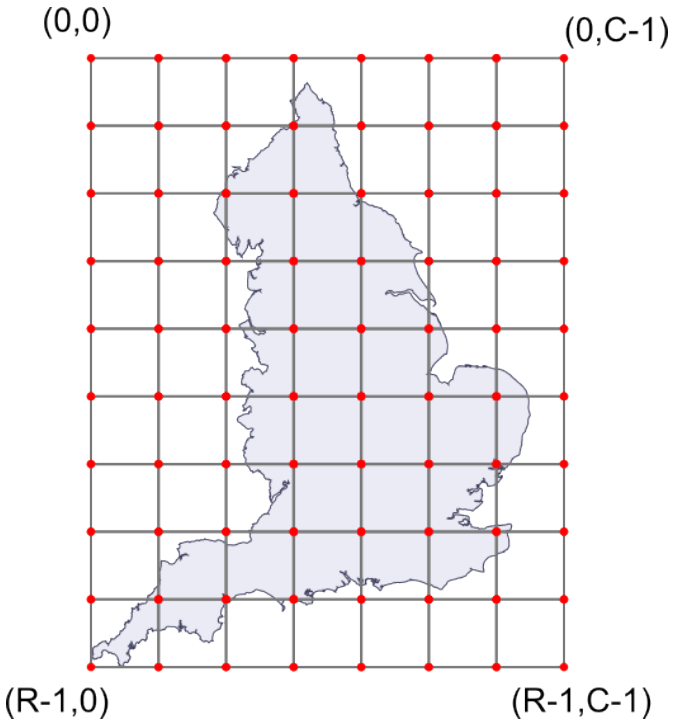
Standard Input	Standard Output
2 2 2 3 4 8 11 1 1 10 1 2 3	4095 1024

Problem E. Measuring Volume of Land

Recently, Frograms Inc. acquired a calm and beautiful uninhabited island in the Pacific Ocean to set up their new headquarter office. Before embarking on the construction, we want to come up with a good disaster plan. Luckily, the new office is architected to resist catastrophes such as hurricane, earthquake, and nuclear plant meltdowns. But there is one thing we haven't considered yet; the global warming.

Experts agree that in the future, as the average temperature of Earth rises, the ice sheets in Antartica and Greenland will melt and the global sea level will rise up to 6 meters. We want to avoid the situation in which our office gets underwater.

Therefore, you are given with the following task: given a **heightmap** of the island, calculate the volume of the land mass above sea level after the sea level rises by L meters, approximated from using **triangulated terrains** from a **heightmap**. If you don't know what **heightmap**s or **triangulated terrains** are, don't worry and read on.

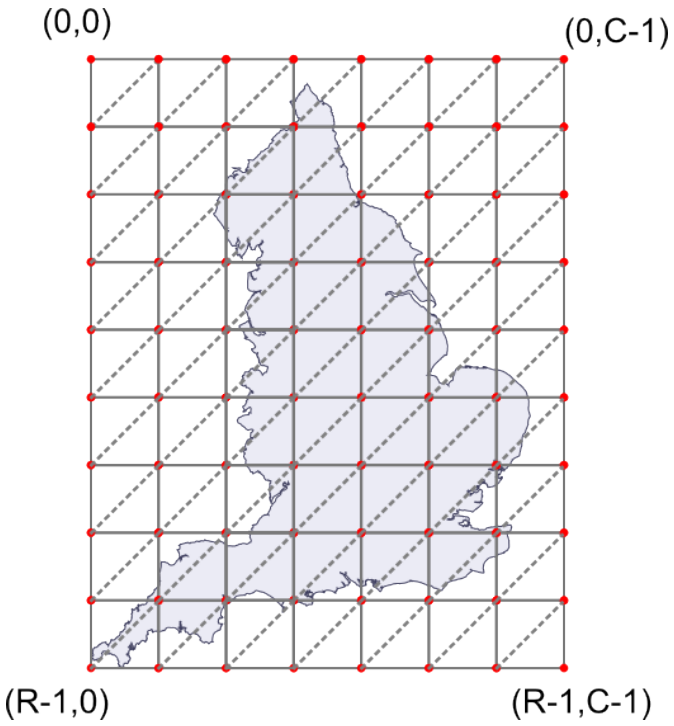


To measure the terrain landscape, a commercial satelite will scan a rectangular area of $R-1$ meters by $C-1$ meters. (You can be sure that this area will completely enclose the island.) The area is then divided into an uniform grid with $R \times C$ grid points, and the satelite will measure the elevation above sea level (in *centimeters*) at every grid point. The above figure gives such an example; the elevation will be measured for every red point in the grid. The resulting 2D array is called a **heightmap**, and this will be given as your input.

Given a heightmap, there are multiple ways to approximate the actual shape of the terrain. One obvious way is to assume that each point in a heightmap correspond to a 1×1 meter square with the given elevation (if you know what it is, think Minecraft) — but those approximations are too crude.

Another way that is still simple, but gives a better precision is approximating the surface as a set of triangles in the 3D space. We can do it as follows: let's take all the squares in the grid, and cut them

diagonally to make two congruent triangles. For any square with four corners at (r, c), $(r+1, c)$, $(r, c+1)$ and $(r+1, c+1)$, we cut them through the line passing $(r+1, c)$ and $(r, c+1)$. See below for an example.



After that, we can shift each grid point in the 3D space vertically to match its measured height. Now, each triangle connects three points in the 3D space. This is how we approximate the shape of the terrain.

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 will contain three integers R, C ($3 \le R, C \le 100$) and L ($0 \le L \le 2^{16} - 1$). The following R lines will contain the heightmap. Each of those lines will contain C nonnegative integers $H_{r,c}$ ($0 \le H_{r,c} \le 2^{16} - 1$), representing the elevation of the point above sea level, in *centimeters*.

Since the island is on the sea, the measurements at the border of the heightmap are always zero.

Output

Print exactly one line for each test case. The line should contain the volume of the land in unit of cubic meters(m^3), assuming the sea level has risen by L meters. Your result should be accurate to within a relative or absolute error of 10^{-8}.

Sample input and output

Standard Input	Standard Output
2 3 3 0 0 0 0 0 100 0 0 0 0 4 5 50 0 0 0 0 0 0 100 50 50 0 0 50 100 50 0 0 0 0 0 0	1.00000000000 0.58333333333

Problem F. rograms 초등학교

프로그램스 초등학교에서 학생들을 두 팀으로 나누어 보물찾기 놀이를 했다. 보물찾기 놀이는 상품의 가치가 숫자로 쓰여 있는 조그만 종이 쪽지를 선생님들이 여기저기에 숨겨 놓은 뒤 정해진 시간 내에 찾아낸 보물의 가치의 총 합이 더 큰 팀이 승리하는 놀이이다. 보물 찾기가 시작된 뒤 학생들이 보물을 찾아오면 박태훈 교장선생님이 보물 가치의 총 합을 구해서 바로 바로 어떤 팀이 앞서고 있는지 방송으로 알려 주었다. 그런데 놀이가 끝나고 보니 어떤 보물을 어떤 팀이 찾아왔는지를 적어놓지 않았다. 중요한 정보는 아니지만 대회 기록을 정리하는 오경윤 선생님은 방송으로 나간 팀들의 순서와 맞아 떨어지도록 보물을 나누고 싶었다. 학생들이 찾아온 보물들의 가치와 중간중간 방송으로 알려진 앞서나간 팀의 번호가 주어졌을때 이에 맞추어 찾아온 순서와 찾아온 팀을 구해보는 프로그램을 작성하라.

Input

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

각 테스트 케이스는 세 줄로 구성된다. 첫 줄에는 학생들이 찾아 온 보물의 수를 나타내는 정수 N $(1 \leq N \leq 10,000)$ 이 주어진다. 그 다음 줄에는 각 보물의 가치를 의미하는 N 개의 정수 T_i $(1 \leq T_i \leq 1000)$ 가 공백으로 구분되어 주어진다. 마지막으로, 세 번째 줄에는 보물을 찾아 온 순간마다 방송되었던 앞서고 있는 팀의 번호가 1 또는 2로 N 차례 공백으로 구분되어 주어진다.

Output

각 테스트 케이스마다 N 행에 걸쳐 보물을 찾아 온 순서에 따라 보물의 가치와 해당 보물을 찾아 온 팀의 번호를 하나의 공백으로 구분하여 출력한다. 가능한 방법이 여러 가지가 있는 경우, 그 중 한 가지만 출력하도록 한다.

Sample input and output

Standard Input	Standard Output
2 5 3 7 8 2 11 1 2 2 1 1 10 1 37 28 13 30 100 26 8 17 5 1 2 1 1 1 2 2 1 1 2	7 1 8 2 3 2 11 1 2 1 17 1 26 2 28 1 13 2 8 1 30 2 5 2 37 1 1 1 100 2

Problem G. Image Filter

알고리즘 대회를 준비할 때면, 출제진은 서로 데이터를 교환하고 검증하는 작업을 거치게 된다. 높이맵(heightmap)을 활용한 문제를 내던 B모씨는 데이터를 모두 만들었고 만족하였다.

이제 데이터를 검증하는 절차를 남겨놓고 있는 상황이었는데 한 가지 문제를 발견했다. 높이맵을 작성할 때는 별 생각 없이 16비트 그레이스케일 비트맵으로 만들었지만, 곧 높이맵의 파일 크기가 너무 커서 출제진끼리 실시간으로 주고 받기에 곤란함이 있다는 것을 깨닫게 되었다.

이러한 문제를 해결하기 위해 다음과 같이 필터를 사용한 비트맵 압축 방식을 설계하였다. 먼저, 압축은 각 칸마다 차례로 수행된다. p행 q열의 높이맵에서의 높이를 $H_{p,q}$라 하자. 압축할 대상 칸의 위치가 r 행 c 열일 때,

- $A = H_{r,c-1}$, 또는 그 위치에 칸이 존재하지 않으면 0
- $B = H_{r-1,c}$, 또는 그 위치에 칸이 존재하지 않으면 0
- $C = H_{r-1,c-1}$, 또는 그 위치에 칸이 존재하지 않으면 0

로 정의하자. 압축된 이미지에서 모든 높이는 다음 표에 기술한 다섯 가지 필터 중 하나를 사용해 표현하게 된다. 압축된 칸은 필터 번호와 그 필터 번호로 계산해 낸 예측값과의 차이로 표현된다.

번호	필터 예측값
0	0
1	A
2	B
3	$(A+B)$를 2로 나눈 몫
4	$A+B-C$

예를 들어 다음과 같은 2행 2열의 높이맵이 있다고 하자.

```
6  4
2  1
```

여기서 우측 하단의 '1'을 표현하는 경우를 살펴보면,

- 0번 필터의 경우 예측값과의 차이는 $1-0=1$
- 1번 필터의 경우 예측값과의 차이는 $1-2=-1$
- 2번 필터의 경우 예측값과의 차이는 $1-4=-3$
- 3번 필터의 경우 예측값과의 차이는 $1-(2+4)/2=-2$
- 4번 필터의 경우 예측값과의 차이는 $1-(2+4-6)=1$

이 된다.

압축된 전체 이미지는 가능한 모든 필터 사용 방법들 중에 예측값과의 차의 절대값의 총 합을 최소화해야 한다. 만일 그러한 방법이 여러 가지가 있다면, 압축된 각 칸들을 구성하는 값의 쌍을 왼쪽 위에서부터 열 방향으로 순차적으로 나열했을 때 사전순으로 최소인 방법을 선택해야 한다. 이와 같은 압축을 수행하는 프로그램을 작성하라.

Input

입력은 T 개의 테스트 케이스로 구성된다. 입력의 첫 줄에는 T가 주어진다.

각 테스트 케이스의 첫 줄에는 높이맵의 크기 R, C ($1 \leq R \leq 100, 1 \leq C \leq 100$)가 공백으로 구분되어 주어진다. 높이맵의 크기는 R행 C열이다. 이후 R개의 줄에, 높이맵의 픽셀의 값을 나타내는 C개의 정수 $H_{i,j}$ ($0 \leq H_{i,j} < 2^{16}$)들이 빈 칸을 사이에 두고 주어진다.

Output

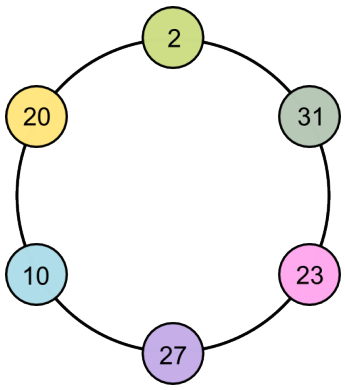
각 테스트 케이스마다 R개의 줄을 출력한다. 각 줄마다 적용한 필터의 번호와 예측값과의 차이를 나타내는 정수를 공백으로 구분하여 C쌍을 출력한다.

Sample input and output

Standard Input	Standard Output
3 2 2 6 4 2 1 3 3 0 0 0 0 100 0 0 0 0 4 5 0 0 0 0 0 0 100 50 50 0 0 50 100 50 0 0 0 0 0 0	0 6 3 1 3 -1 0 1 0 0 0 0 0 0 0 0 0 100 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 100 3 0 1 0 0 0 0 0 3 0 1 50 2 0 0 0 0 0 0 0 0 0 0 0 0 0

Problem H. XOR Necklace

Programs, Inc. has been busy developing necklaces with brand-new design and feature. Every programmer working at Programs now wears this necklace. A necklace consists of multiple beads stringed by a thread, as seen below. Each bead is given a number between 0 to 10^9 (inclusive).



Most of the programmers have left for their summer vacation, and there is only a few programmers in the office today. So they decided to play a game using their own necklace to decide who will buy lunch for everyone. The rule is simple: for each pair of consecutive beads in the necklace, we take sum of their numbers. A programmer's score is defined as the XOR(denoted as $\oplus$) of all those sums. The programmer with the lowest score loses! More formally, if there are N beads and their value is denoted by A_1, A_2, $\cdots$, A_N in the order they are stringed together, then the score is:

$$(A_1 + A_2) \oplus (A_2 + A_3) \oplus ... \oplus (A_{N-1} + A_N) \oplus (A_N + A_1)$$

However, you, the most talented programmer in Programs, Inc., is going to cheat the game by removing zero or more beads to maximize the score. What is the maximum score if you are allowed to remove some of the beads in a given necklace? Note that you cannot reorder the beads, nor leave one or less beads in a necklace.

Input

The input consists of T test cases. The first line of the input contains T.

Each test case starts with a line containing a single integer N ($2 \leq N \leq 500$), the number of beads on your necklace. The next line contains N integers A_1, A_2, $\cdots$, A_N, separated by a single space.

Output

For each test case, print the maximum score possible you can get in a single line.

Sample input and output

Standard Input	Standard Output
1 6 2 31 23 27 10 20	54

Problem I. Algospot Design School

"Woojoo! Where have you been?"

"What's the matter, teacher?"

"I told you that the admissions information on Algospot Design School was out, and that you needed to prepare a portfolio for that. Did you finish drawing it yet?"

"Yeah, sure. All done, perfectly."

"Really? Let me take a look at it."

"I forgot to bring it today. I can show you tomorrow."

"Are you sure? Don't forget!"

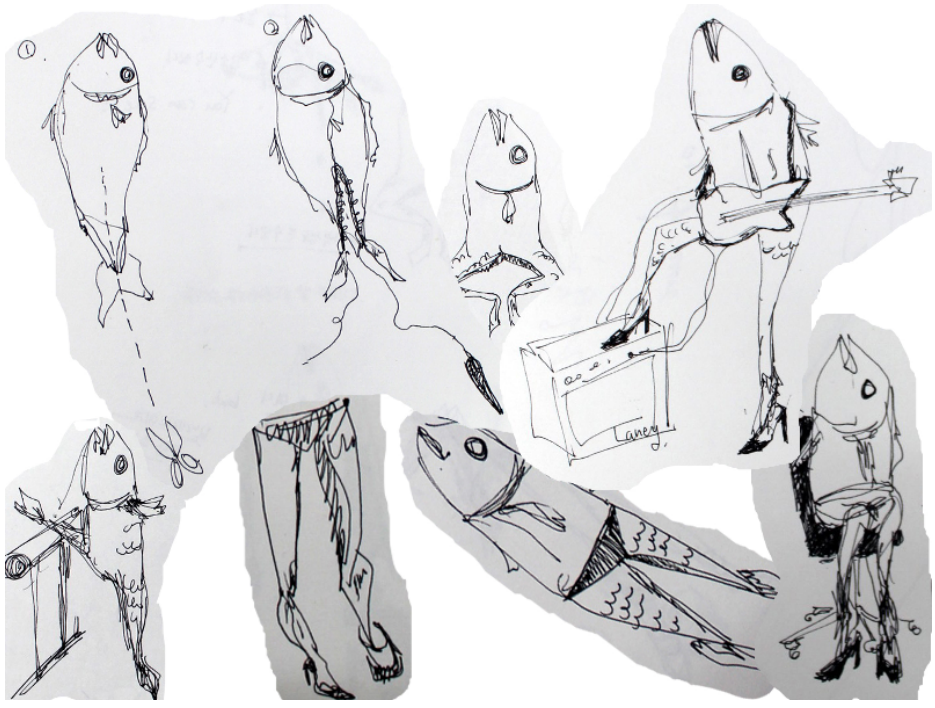
After talking to the teacher, Woojoo rushed to meet her best friend, Bada.

"Hey, Bada! I'm screwed!"

"What's up? Why?"

"I forgot to prepare a portfolio for Algospot Design School. Do you have any extra one that I can borrow?"

"Extra ones… Oh, I have one that I drew for the exhibition the other day. Wait a second…"



Bada found her sketchbook and gave it to Woojoo. The book had a lot of drawings, on both front and back of the pages.

"That's a gift for you. You can use however you would like."

"Really? Is that OK?"

"It's my pleasure."

On the next day, Woojoo brought the sketchbook to the teacher.

"Hmm… the drawings look good, but why did you on earth draw so many things on a single page? Worse yet, you have drawings on the back, too. I think you would need to cut the drawings out. Try to cut them so that you will be left with the most drawings."

Here comes your task:

We place a single, square page from the sketchbook on a 2D plane, with two opposite corners at $(0, 0)$ and $(100, 100)$. When you want to cut out a piece of drawing from the page, you have to cut it along the smallest enclosing axis-aligned (i.e. borders are parallel to either X or Y axis) rectangle. You are given coordinates of these rectangles, and which side the drawing is on. Find the maximum number of pieces that can be cut out from the page. Please note that a single piece can only count for up to 1 drawing even if it contains other drawings on the other side.

Input

The input consists of T test cases, each representing a sheet of the sketchbook. The first line of the input contains T.

Each test case describes the drawings on the sheet. The first line of the test case contains the number of drawings on both sides of the sheet, N ($1 \leq N \leq 100$).

The following N lines describe the N drawings: the coordinates of the bottom-left corner, the coordinates of the upper-right corner, and its side (`F` for front, `B` for back). You can assume that no two drawings on the same page will overlap with each other.

Output

For each sheet, output the maximum number of drawings you can cut-out of the sheet.

Sample input and output

Standard Input	Standard Output
4 2 0 0 1 1 F 0 0 1 1 B 3 1 1 4 4 F 0 0 2 2 B 2 2 5 5 B 4 0 0 1 1 F 0 1 1 2 F 0 0 1 2 B 0 2 1 4 B 4 0 0 50 50 F 50 50 100 100 F 0 50 50 100 B 50 0 100 50 B	1 2 3 4

Problem J. GGGCCCDDD

2012년 8월 18일. 시대를 앞서가는 프로그래밍 연구소 Algospot에서는 새로운 슈퍼컴퓨터 AEX를 개발했다. Algospot의 연구원 altertain은 AEX의 성능을 알아보기 위해서 GCD(최대공약수)의 합을 계산하는 간단한 프로그램을 작성했다.

```
int gcd(int a, int b)
{
    return b ? gcd(b, a%b) : a;
}

sum = 0;
for (a = 1; a <= N; a++)
    for (b = 1; b <= N; b++)
        sum = sum + gcd(a, b);
```

"흠… 이 정도는 금방 계산하겠지…?"

슈퍼컴퓨터 AEX는 눈깜빡할 사이에 계산을 마치고 답을 출력했다.

"음… 그럼 루프를 하나 늘려볼까…?"

```
sum = 0;
for (a = 1; a <= N; a++)
    for (b = 1; b <= N; b++)
        for (c = 1; c <= N; c++)
            sum = sum + gcd(gcd(a, b), c);
```

"와… 엄청 빠르다!"

슈퍼컴퓨터 AEX는 이것도 순식간에 계산해냈다.

"뭐하고 있어?"

Algospot에 놀러온 Afocuns의 연구원 rainpunch가 altertain에게 물었다.

"어 왔어? 이번에 우리 연구소에서 슈퍼컴퓨터를 새로 개발했거든. 그거 테스트해보고 있었어. 여기 코드에 루프 하나씩 늘려보던 중…"

똑똑한 연구원인 rainpunch가 코드를 보더니 한마디 했다.

"뭐야? 이런 건 루프 10개라도 금방 계산하겠다… 아니? 음… 1000개라도 금방 하겠는데?"

"뭐라고!? 그럼 루프 M 개에 대해 누가 더 빨리 계산하나 진검승부 한 번 해볼까!?"

"좋아. 내가 의지의 차이를 제대로 보여주지."

"흥! 컴퓨터의 천재, 박수를 드려요!"

rainpunch가 떠올린 방법은 무엇일까? 우리도 한 번 좋은 방법을 찾아보자.

Input

입력은 T 개의 테스트 케이스로 구성된다. 입력의 첫 줄에는 T가 주어진다.

각 테스트 케이스에 대해, 한 줄에 정수 N과 M ($2 \leq N, M \leq 1000$) 이 공백으로 구분되어 주어진다.

Output

각 테스트 케이스마다 한 줄에 하나씩 답을 출력한다. 단, 답이 매우 클 수 있으므로 답을 1000000007로 나눈 나머지를 출력한다.

Sample input and output

Standard Input	Standard Output
2 2 2 10 4	5 10905

Notes

$N = 10$, $M = 4$ 인 경우, 아래처럼 계산해볼 수 있다.

```
sum = 0;
for (a = 1; a <= 10; a++)
    for (b = 1; b <= 10; b++)
        for (c = 1; c <= 10; c++)
            for (d = 1; d <= 10; d++)
                sum = sum + gcd(gcd(gcd(a, b), c), d);
```