Problem A. Sequence and Sequence

Consider the following two sequences P and Q. We denote $P(i)$ as the i-th element in sequence P, and $Q(i)$ as the i-th element in sequence Q:

- Sequence P is a **sorted** sequence where for all $k \in \mathbb{Z}^+$, k appears in sequence P for $(k+1)$ times ($\mathbb{Z}^+$ is the set of all positive integers). That is to say,

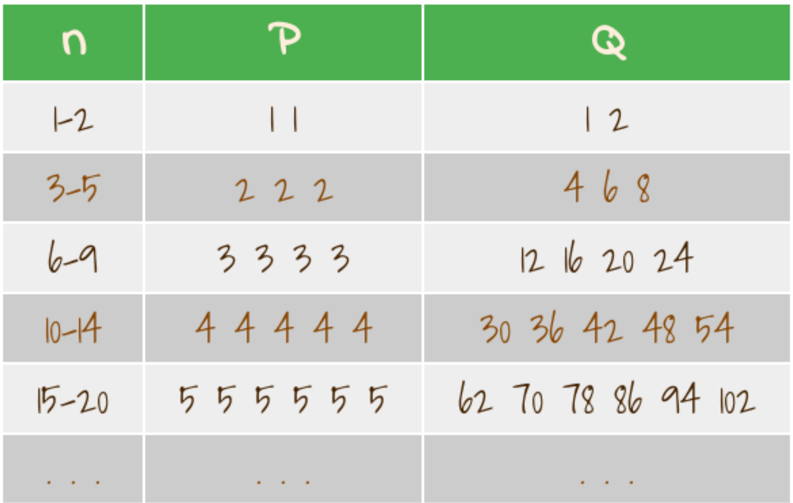
$$P = \{1, 1, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 6, \ldots\}$$

- Sequence Q can be derived from the following equations:

$$\begin{cases} Q(1) = 1 \\ Q(i) = Q(i-1) + Q(P(i)) & \text{if } i > 1 \end{cases}$$

That is to say,

$$Q = \{1, 2, 4, 6, 8, 12, 16, 20, 24, 30, 36, 42, 48, 54, 62, \ldots\}$$



n	P	Q
1-2	1 1	1 2
3-5	2 2 2	4 6 8
6-9	3 3 3 3	12 16 20 24
10-14	4 4 4 4 4	30 36 42 48 54
15-20	5 5 5 5 5 5	62 70 78 86 94 102
. . .	. . .	. . .

Given a positive integer n, please calculate the value of $Q(n)$.

Input

There are multiple test cases. The first line of the input contains an integer T (about 10^4), indicating the number of test cases. For each test case:

The first and only line contains an integer n ($1 \le n \le 10^{40}$).

Output

For each test case output one line containing one integer, indicating the value of $Q(n)$.

Example

standard input	standard output
4 10 100 1000 987654321123456789	30 2522 244274 235139898689017607381017686096176798

Problem B. Kawa Exam

BaoBao is taking an online exam of the Kawa programming language, which consists of n multiple choice questions. The exam is not easy, as for each question, BaoBao needs to select the one and only one correct answer from 10^5 available choices! But don't worry, as an active committer of the famous *open-kdk*, BaoBao obviously knows all the correct answers.

Although BaoBao is an expert in Kawa, the developers of the exam system are definitely not experts in software engineering. There are m bugs in the exam system, and the i-th bug can be described as (u_i, v_i), which means that BaoBao must select the same choice for question u_i and v_i (even if the choice he selects is incorrect!).

Time is limited, and the exam must go on. The developers only have time to fix one bug. Now the developers are wondering, for all $1 \le i \le m$, what's the maximum possible number of questions BaoBao can correctly answer if they fix the i-th bug. Please write a program to solve this problem so that the developers can select a proper bug to fix.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n \le 10^5$, $1 \le m \le 10^5$), indicating the number of questions and the number of bugs.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^5$), where a_i indicates the correct answer for question i.

For the following m lines, the i-th line has two integers u_i and v_i ($1 \le u_i, v_i \le n$), indicating the i-th bug.

It's guaranteed that neither the sum of n nor the sum of m over all test cases will exceed 10^6.

Output

For each test case output one line containing m integers $c_1, c_2, \dots, c_m$ separated by a space, where c_i indicates the maximum possible number of questions BaoBao can correctly answer if the i-th bug is fixed.

Please, DO NOT output extra spaces at the end of each line, or your answer may be considered incorrect!

Example

standard input	standard output

3
7 5
1 2 1 2 1 2 1
1 2
1 3
2 4
5 6
5 7
3 3
1 2 3
1 2
1 3
2 3
2 3
12345 54321
1 2
1 2
1 1
	6 5 5 5 4
	1 1 1
	1 1 1

Note

The following table explains the first sample test case.

- The "possible choices" column indicates a possible set of choices to each question BaoBao can select, leading to a maximum possible number of correctly answered questions;
- The "correct" column indicates the number of correctly answered questions using the set of choices provided in the "possible choices" column.

Bug No.	**Possible Choices**	**Correct**
1	(1, 2, 1, 2, 1, 1, 1)	6
2	(2, 2, 1, 2, 1, 1, 1)	5
3	(1, 1, 1, 2, 1, 1, 1)	5
4	(1, 1, 1, 1, 1, 2, 1)	5
5	(1, 1, 1, 1, 1, 1, 1)	4

For the second sample test case, no matter which bug is fixed, BaoBao has to select the same choice for all the three questions. As the correct answer for each question is different, BaoBao can only correctly answer 1 question.

For the third sample test case, note that even if the developers fix the first bug, the second bug is still taking effect and BaoBao still has to select the same choice for the two problems. It's the same if the second bug is fixed.

Problem C. Flippy Sequence

DreamGrid has just found two binary sequences $s_1, s_2, \ldots, s_n$ and $t_1, t_2, \ldots, t_n$ ($s_i, t_i \in \{0, 1\}$ for all $1 \le i \le n$) from his virtual machine! He would like to perform the operation described below exactly twice, so that $s_i = t_i$ holds for all $1 \le i \le n$ after the two operations.

The operation is: Select two integers l and r ($1 \le l \le r \le n$), change s_i to $(1 - s_i)$ for all $l \le i \le r$.

DreamGrid would like to know the number of ways to do so.

We use the following rules to determine whether two ways are different:

- Let $A = (a_1, a_2, a_3, a_4)$, where $1 \le a_1 \le a_2 \le n, 1 \le a_3 \le a_4 \le n$, be a valid operation pair denoting that DreamGrid selects integers a_1 and a_2 for the first operation and integers a_3 and a_4 for the second operation;
- Let $B = (b_1, b_2, b_3, b_4)$, where $1 \le b_1 \le b_2 \le n, 1 \le b_3 \le b_4 \le n$, be another valid operation pair denoting that DreamGrid selects integers b_1 and b_2 for the first operation and integers b_3 and b_4 for the second operation.
- A and B are considered different, if there exists an integer k ($1 \le k \le 4$) such that $a_k \ne b_k$.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^6$), indicating the length of two binary sequences.

The second line contains a string $s_1 s_2 \ldots s_n$ ($s_i \in \{\text{'0'}, \text{'1'}\}$) of length n, indicating the first binary sequence.

The third line contains a string $t_1 t_2 \ldots t_n$ ($t_i \in \{\text{'0'}, \text{'1'}\}$) of length n, indicating the second binary sequence.

It's guaranteed that the sum of n in all test cases will not exceed 10^7.

Output

For each test case, output an integer denoting the answer.

Example

standard input	standard output
3 1 1 0 2 00 11 5 01010 00111	0 2 6

Note

For the second sample test case, there are two valid operation pairs: $(1, 1, 2, 2)$ and $(2, 2, 1, 1)$.

For the third sample test case, there are six valid operation pairs: $(2, 3, 5, 5)$, $(5, 5, 2, 3)$, $(2, 5, 4, 4)$, $(4, 4, 2, 5)$, $(2, 4, 4, 5)$ and $(4, 5, 2, 4)$.

Problem D. Magic Multiplication

BaoBao is now learning a new binary operation between two positive integers, represented by $\otimes$, in his magic book. The book tells him that the result of such operation is calculated by concatenating all multiple results of each digit in the two integers.

Formally speaking, let the first integer be $A = a_1a_2 \dots a_n$, where a_i indicates the i-th digit in A, and the second integer be $B = b_1b_2 \dots b_m$, where b_i indicates the i-th digit in B. We have

$$A \otimes B = \sum_{i=1}^{n} \sum_{j=1}^{m} a_ib_j = a_1b_1 + a_1b_2 + \cdots + a_1b_m + a_2b_1 + \cdots + a_nb_m$$

Note that the result of a_ib_j is considered to be a **string** (without leading zeros if $a_ib_j > 0$, or contains exactly one '0' if $a_ib_j = 0$), NOT a normal integer. Also, the sum here means **string concatenation**, NOT the normal addition operation.

For example, $23 \otimes 45 = 8101215$. Because $8 = 2 \times 4$, $10 = 2 \times 5$, $12 = 3 \times 4$ and $15 = 3 \times 5$.

BaoBao is very smart and soon knows how to do the inverse operation of $\otimes$. Now he gives you the result of a $\otimes$ operation and the numbers of digits in the two original integers. Please help him to restore the two original integers A and B.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two positive integers n and m ($1 \le n, m \le 2 \times 10^5$), where n indicates the length of A and m indicates the length of B. Here length of an integer means the length of the string when writing the number in decimal notation without leading zeros.

The second line contains only one positive integer C without leading zeros, indicating the result of $A \otimes B$. The length of C is no more than 2×10^5.

It's guaranteed that the sum of lengths of C over all test cases will not exceed 2×10^6.

Output

For each test case output one line.

If there exist such A and B that $A \otimes B = C$, output one line containing two integers A and B separated by one space. Note that A and B should be positive integers without leading zeros, the length of A should be exactly n, and the length of B should be exactly m.

If there are multiple valid answers, output the answer with the smallest A; If there are still more than one answer, output one of them with the smallest B.

If such A and B do not exist, print "Impossible" (without quotes) on a single line.

Example

standard input	standard output

4
2 2
8101215
3 4
100000001000
2 2
80101215
3 4
1000000010000
	23 45
	101 1000
	Impossible
	Impossible

Problem E. Plants vs. Zombies

BaoBao and DreamGrid are playing the game *Plants vs. Zombies*. In the game, DreamGrid grows plants to defend his garden against BaoBao's zombies.

Plants vs. Zombies(?)
(Image from pixiv. ID: 21790160; Artist: socha)

There are n plants in DreamGrid's garden arranged in a line. From west to east, the plants are numbered from 1 to n and the i-th plant lies i meters to the east of DreamGrid's house. The i-th plant has a defense value of d_i and a growth speed of a_i. Initially, $d_i = 0$ for all $1 \le i \le n$.

DreamGrid uses a robot to water the plants. The robot is in his house initially. In one step of watering, DreamGrid will choose a direction (east or west) and the robot moves exactly 1 meter along the direction. After moving, if the i-th plant is at the robot's position, the robot will water the plant and a_i will be added to d_i. Because the water in the robot is limited, at most m steps can be done.

The defense value of the garden is defined as $\min\{d_i | 1 \le i \le n\}$. DreamGrid needs your help to maximize the garden's defense value and win the game.

Please note that:

- Each time the robot MUST move before watering a plant;
- It's OK for the robot to move more than n meters to the east away from the house, or move back into the house, or even move to the west of the house.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two integers n and m ($2 \le n \le 10^5$, $0 \le m \le 10^{12}$), indicating the number of plants and the maximum number of steps the robot can take.

The second line contains n integers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^5$), where a_i indicates the growth speed of the i-th plant.

It's guaranteed that the sum of n in all test cases will not exceed 10^6.

Output

For each test case output one line containing one integer, indicating the maximum defense value of the garden DreamGrid can get.

Example

standard input	standard output

2
4 8
3 2 6 6
3 9
10 10 1
	6
	4

Note

In the explanation below, ‘E’ indicates that the robot moves exactly 1 meter to the east from his current position, and ‘W’ indicates that the robot moves exactly 1 meter to the west from his current position.

For the first test case, a candidate direction sequence is {E, E, W, E, E, W, E, E}, so that we have $d = \{6, 6, 12, 6\}$ after the watering.

For the second test case, a candidate direction sequence is {E, E, E, E, W, E, W, E, W}, so that we have $d = \{10, 10, 4\}$ after the watering.

Problem F. Tournament

DreamGrid, the king of Gridland, is making a knight tournament. There are n knights, numbered from 1 to n, participating in the tournament. The rules of the tournament are listed as follows:

- The tournament consists of k rounds. Each round consists of several duels. Each duel happens between exactly two knights.
- Each knight must participate in exactly one duel during each round.
- For each pair of knights, there can be at most one duel between them during all the k rounds.
- Let $1 \le i, j \le k$, $i \ne j$, and $1 \le a, b, c, d \le n$, a, b, c, d be four distinct integers. If
  - Knight a fights against knight b during round i, and
  - Knight c fights against knight d during round i, and
  - Knight a fights against knight c during round j,

  then knight b must fight against knight d during round j.

As DreamGrid's general, you are asked to write a program to arrange all the duels in all the k rounds, so that the resulting arrangement satisfies the rules above.

Input

There are multiple test cases. The first line of the input is an integer T, indicating the number of test cases. For each test case:

The first and only line contains two integers n and k ($1 \le n, k \le 1000$), indicating the number of knights participating in the tournament and the number of rounds.

It's guaranteed that neither the sum of n nor the sum of k in all test cases will exceed 5000.

Output

For each test case:

- If it's possible to make a valid arrangement, output k lines. On the i-th line, output n integers $c_{i,1}, c_{i,2}, \ldots, c_{i,n}$ separated by one space, indicating that in the i-th round, knight j will fight against knight $c_{i,j}$ for all $1 \le j \le n$.

  If there are multiple valid answers, output the lexicographically smallest answer.

  Consider two answers A and B, let's denote $a_{i,j}$ as the j-th integer on the i-th line in answer A, and $b_{i,j}$ as the j-th integer on the i-th line in answer B. Answer A is lexicographically smaller than answer B, if there exists two integers p ($1 \le p \le k$) and q ($1 \le q \le n$), such that

  - for all $1 \le i < p$ and $1 \le j \le n$, $a_{i,j} = b_{i,j}$, and
  - for all $1 \le j < q$, $a_{p,j} = b_{p,j}$, and finally $a_{p,q} < b_{p,q}$.
- If it's impossible to make a valid arrangement, output "Impossible" (without quotes) in one line.

Please, DO NOT output extra spaces at the end of each line, or your answer may be considered incorrect!

Example

standard input	standard output
2	Impossible
3 1	2 1 4 3
4 3	3 4 1 2
	4 3 2 1

Problem G. Repair the Artwork

DreamGrid has a paper strip with n grids in a line and he has drawn some beautiful patterns in some grids. Unfortunately, his naughty roommate BaoBao drew some other patterns in some other grids when he wasn't at home. Now DreamGrid has to erase BaoBao's patterns without destroying his own patterns.

Let's number the grids from 1 to n from left to right. Each grid either contains DreamGrid's pattern, or contains BaoBao's pattern, or is empty.

Each time, DreamGrid can select two integers l and r (these two integers can be different each time) such that

- $1 \le l \le r \le n$, and
- for all $l \le i \le r$, the i-th grid either contains BaoBao's pattern, or is empty,

and change the i-th grid to an empty grid for all $l \le i \le r$.

How many ways can DreamGrid change all the grids containing BaoBao's pattern to empty grids by performing the above operation exactly m times? As the answer may be large, please print the answer modulo $10^9 + 7$.

Let $\{(a_1, b_1), (a_2, b_2), \dots (a_m, b_m)\}$ be a valid erasing sequence ($1 \le a_i \le b_i \le n$), where (a_i, b_i) indicates that DreamGrid selects integers a_i and b_i in the i-th operation. Let $\{(c_1, d_1), (c_2, d_2), \dots, (c_m, d_m)\}$ be another valid erasing sequence ($1 \le c_i \le d_i \le n$), where (c_i, d_i) indicates that DreamGrid selects integers c_i and d_i in the i-th operation. These two sequences are considered different, if there exists an integer k ($1 \le k \le m$) such that $a_k \ne c_k$ or $b_k \ne d_k$.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 1000$), indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n \le 100$, $1 \le m \le 10^9$), indicating the number of grids and the number of times to perform the operation.

The second line contains n integers a_i ($a_i \in \{0, 1, 2\}$).

- If $a_i = 0$, the i-th grid is empty.
- If $a_i = 1$, the i-th grid contains DreamGrid's pattern.
- If $a_i = 2$, the i-th grid contains BaoBao's pattern.

It's guaranteed that at most 50 test cases have $n > 50$.

Output

For each test case, output one line containing the number of ways modulo $10^9 + 7$.

Example

standard input	standard output
3	8
2 2	3
2 0	1
3 2	
2 1 0	
3 1	
2 1 0	

Problem H. Mirror

There is a non-transparent obstacle and a single-sided mirror in an infinite two-dimensional plane. The obstacle can be represented as a triangle and the mirror can be represented as a **directional** line segment pointing from $(x_{m,1}, y_{m,1})$ to $(x_{m,2}, y_{m,2})$, with the right side being reflective.

There are m stones at point (x_1, y_1) and DreamGrid would like to move all the stones to point (x_2, y_2). The following constraints must be satisfied:

- DreamGrid can only carry one stone each time.
- Once DreamGrid picks up a stone, he can only put it down at point (x_2, y_2).
- Let L be the path DreamGrid walked, then for each point p on L, DreamGrid should see all the stones directly or through the mirror.

Note that:

- If some part of the line vision is inside the obstacle (it's ok that the line vision passes a point or edge of the obstacle), it's considered, that DreamGrid cannot see the stone with this line of vision.
- If one of the two endpoints of the mirror is on the line of vision, it's considered, that DreamGrid can see the stone both in the mirror and through the mirror.
- The reflection process is governed by laws of physics — the angle of incidence is equal to the angle of reflection. The incident ray is in the same half-plane as the reflected ray, relative to the mirror.
- If the line of vision is parallel to the mirror, reflection doesn't take place, and the mirror isn't regarded as an obstacle.
- DreamGrid cannot move into the obstacle but can move on the edges or the vertices of the obstacle.
- DreamGrid cannot move through the mirror but can move on the mirror. Note that if DreamGrid is on the line segment of the mirror other than the two endpoints, he can only see the side he comes from, and cannot see through the mirror.

DreamGrid would like to know the shortest distance to move all stones from (x_1, y_1) to (x_2, y_2).

Input

There are multiple test cases. The first line of input is an integer T (about 100), indicates the number of test cases. For each test case:

The first line contains one integer m $(1 \le m \le 10^6)$, indicating the number of stones.

The second line contains four integers x_1, y_1, x_2 and y_2, indicating the start point and the target point.

The third line contains four integers $x_{m,1}$, $y_{m,1}$, $x_{m,2}$ and $y_{m,2}$, indicating the coordinates of the mirror.

Each of the next 3 lines has two integers x_i and y_i, indicating the coordinates of the vertices of the obstacle.

All the coordinates will not exceed 100 in absolute value. Both the start point and target point are outside the obstacle and the mirror. The mirror and the obstacle have no points in common.

It is guaranteed that no three points are collinear.

Output

For each test case, output a real number indicating the shortest distance, or output -1 if DreamGrid cannot move all the stones under the constraints.

Your answer will be considered correct if and only if the absolute error or relative error of your answer is less than 10^{-6}.

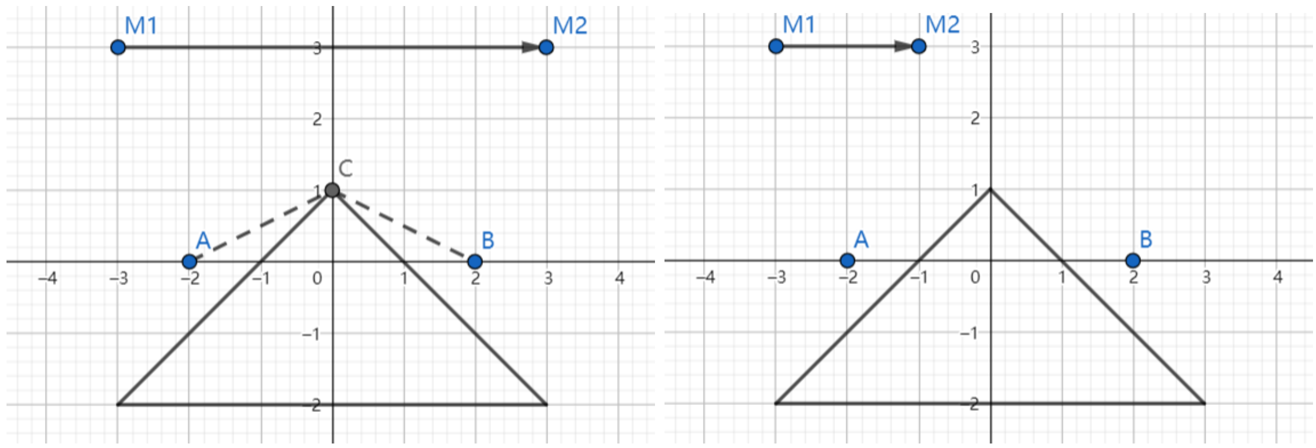
Example

standard input	standard output

2
2
-2 0 2 0
-3 3 3 3
0 1
-3 -2
3 -2
2
-2 0 2 0
-3 3 -1 3
0 1
-3 -2
3 -2
	13.416407864999
	-1

Note

We now welcome our special guest Mashiro, who heartily donates this problem to our problemset, to explain the sample test cases for us using her sketch book.



In the figures above, we indicate the start point as point A and the target point as point B. The mirror is indicated by the line segment pointing from $M1$ to $M2$, with the right side being reflective.

For the first sample test case, the optimal path is $A \to C \to B \to C \to A \to C \to B$.

For the second sample test case, as DreamGrid cannot see A from B, it's impossible to move all the two stones from A to B while following the constraints in the problem description.

(Image from pixiv. ID: 32084305)

Problem I. Soldier Game

DreamGrid and BaoBao are playing a game. There are n soldiers in the game, numbered from 1 to n. The i-th soldier has a power value of a_i. DreamGrid and BaoBao are going to divide the soldiers into several teams according to the rules below:

- A team must consist of 1 or 2 soldiers.
- Every soldier must belong to exactly 1 team.
- If a team consists of two soldiers (let's say they are the i-th and the j-th soldier), there must be $|i - j| = 1$.

The power value of a team is defined as the sum of the team members' power values. For the sake of fairness, they want to minimize the difference between the maximum team power value and the minimum team power value after the division. You are asked to find the minimum difference.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^5$), indicating the number of soldiers.

The second line contains n integers $a_1, a_2, ..., a_n$ ($-10^9 \le a_i \le 10^9$), where a_i indicates the power value of the i-th soldier.

It's guaranteed that the sum of n in all test cases will not exceed 10^6.

Output

For each test case output one line containing one integer, indicating the minimum difference between the maximum team power value and the minimum team power value.

Example

standard input	standard output

3
5
-1 4 2 1 1
4
1 3 2 4
1
7
	1
	2
	0

Note

We now explain the first sample test case. All possible divisions are listed below.

Division	Difference	Division	Difference
[-1], [4], [2], [1], [1]	4 - (-1) = 5	[-1, 4], [2], [1], [1]	3 - 1 = 2
[-1], [4], [2], [1, 1]	4 - (-1) = 5	[-1], [4, 2], [1, 1]	6 - (-1) = 7
[-1], [4], [2, 1], [1]	4 - (-1) = 5	[-1, 4], [2], [1, 1]	3 - 2 = 1
[-1], [4, 2], [1], [1]	6 - (-1) = 7	[-1, 4], [2, 1], [1]	3 - 1 = 2

So the answer is $\min(5, 5, 5, 7, 2, 7, 1, 2) = 1$.

Problem J. Books

DreamGrid went to the bookshop yesterday. There are n books in the bookshop in total. Because DreamGrid is very rich, he bought the books according to the strategy below:

- Check the n books from the 1st one to the n-th one **in order**.
- For each book being checked now, if DreamGrid has enough money (not less than the book price), he'll buy the book and his money will be reduced by the price of the book.
- In case that his money is less than the price of the book being checked now, he will skip that book.

BaoBao is curious about how rich DreamGrid is. You are asked to tell him the maximum possible amount of money DreamGrid took before buying the books, which is a non-negative integer. All he knows are the prices of the n books and the number of books DreamGrid bought in total, indicated by m.

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two integers n and m ($1 \le n \le 10^5$, $0 \le m \le n$), indicating the number of books in the bookshop and the number of books DreamGrid bought in total.

The second line contains n non-negative integers $a_1, a_2, ..., a_n$ ($0 \le a_i \le 10^9$), where a_i indicates the price of the i-th book checked by DreamGrid.

It's guaranteed that the sum of n in all test cases will not exceed 10^6.

Output

For each test case output one line.

If it's impossible to buy m books for any initial number of money, output "Impossible" (without quotes).

If DreamGrid may take an infinite amount of money, output "Richman" (without quotes).

In other cases, output a non-negative integer, indicating the maximum number of money he may take.

Example

standard input	standard output

4
4 2
1 2 4 8
4 0
100 99 98 97
2 2
10000 10000
5 3
0 0 0 0 1
	6
	96
	Richman
	Impossible

Problem K. Airdrop

PUBG is a multiplayer online battle royale video game. In the game, up to one hundred players parachute onto an island and scavenge for weapons and equipment to kill others while avoiding getting killed themselves. Airdrop in this game is a key element, as airdrops often carry with them strong weapons or numerous supplies, helping players to survive.

Airdrop in the game(?)

Consider the battle field of the game to be a two-dimensional plane. An airdrop has just landed at point (x_0, y_0) (both x_0 and y_0 are integers), and all the n players on the battle field, where (x_i, y_i) (both x_i and y_i are integers) indicates the initial position of the i-th player, start moving towards the airdrop with the following pattern:

- If the position of a living player at the beginning of this time unit is not equal to (x_0, y_0), he will begin his next move.
  - Let's say he is currently at point (x, y). For his next move, he will consider four points $(x, y-1)$, $(x, y+1)$, $(x-1, y)$ and $(x+1, y)$.
  - He will select one of the four points whose Manhattan distance to the airdrop (x_0, y_0) is the smallest to be the destination of his next move. Recall that the Manhattan distance between two points (x_a, y_a) and (x_b, y_b) is defined as $|x_a - x_b| + |y_a - y_b|$.
  - If two or more points whose Manhattan distance to the airdrop is the same, he will use the following priority rule to break the tie: $(x, y-1)$ has the highest priority to be selected, $(x, y+1)$ has the second highest priority, $(x-1, y)$ has the third highest priority, and $(x+1, y)$ has the lowest priority.
  - At the end of this time unit, he arrives at his destination.
- If the position of a living player at the beginning of this time unit is equal to (x_0, y_0), he will continue to fatten his backpack with the supplies in the airdrop and stays at (x_0, y_0).

But the battle is tough and it's almost impossible for all the players to arrive at the airdrop safely. If two or more players meet at point (x', y') other than (x_0, y_0), where both x' and y' are integers, they will fight and kill each other and none of them survive.

BaoBao is a big fan of the game and is interested in the number of players successfully arriving at the position of the airdrop, but he doesn't know the value of x_0. Given the value of y_0 and the initial position of each player, please help BaoBao calculate the minimum and maximum possible number of players successfully arriving at the position of the airdrop for all $x_0 \in \mathbb{Z}$, where $\mathbb{Z}$ is the set of all integers (note that x_0 can be positive, zero or negative).

Input

There are multiple test cases. The first line of the input contains an integer T, indicating the number of test cases. For each test case:

The first line contains two integers n and y_0 ($1 \le n \le 10^5$, $1 \le y_0 \le 10^5$), indicating the number of players and the y value of the airdrop.

For the following n lines, the i-th line contains two integers x_i and y_i ($1 \le x_i, y_i \le 10^5$), indicating the initial position of the i-th player.

It's guaranteed that the sum of n in all test cases will not exceed 10^6, and in each test case no two players share the same initial position.

Output

For each test case output one line containing two integers $p_{\min}$ and $p_{\max}$ separated by one space. $p_{\min}$ indicates the minimum possible number of players successfully arriving at the position of the airdrop, while $p_{\max}$ indicates the maximum possible number.

Example

standard input	standard output

3
3 2
1 2
2 1
3 5
3 3
2 1
2 5
4 3
2 3
1 3
4 3
	1 3
	0 3
	2 2

Note

We now explain the first sample test case.

To obtain the answer of $p_{\min} = 1$, one should consider $x_0 = 3$. The following table shows the position of each player at the end of each time unit when $x_0 = 3$.

Time	Player 1	Player 2	Player 3
0	(1, 2)	(2, 1)	(3, 5)
1	(2, 2)	(2, 2)	(3, 4)
2	eliminated	eliminated	(3, 3)
3	eliminated	eliminated	(3, 2)

To obtain the answer of $p_{\max} = 3$, one should consider $x_0 = 2$. The following table shows the position of each player at the end of each time unit when $x_0 = 2$.

Time	Player 1	Player 2	Player 3
0	(1, 2)	(2, 1)	(3, 5)
1	(2, 2)	(2, 2)	(3, 4)
2	(2, 2)	(2, 2)	(3, 3)
3	(2, 2)	(2, 2)	(3, 2)
4	(2, 2)	(2, 2)	(2, 2)

Problem L. Sub-cycle Graph

Given an undirected simple graph with n ($n \geq 3$) vertices and m edges where the vertices are numbered from 1 to n, we call it a "sub-cycle" graph if it's possible to add a non-negative number of edges to it and turn the graph into exactly one simple cycle of n vertices.

Given two integers n and m, your task is to calculate the number of different sub-cycle graphs with n vertices and m edges. As the answer may be quite large, please output the answer modulo $10^9 + 7$.

Recall that

- A simple graph is a graph with no self loops or multiple edges;
- A simple cycle of n ($n \geq 3$) vertices is a connected undirected simple graph with n vertices and n edges, where the degree of each vertex equals to 2;
- Two undirected simple graphs with n vertices and m edges are different, if they have different sets of edges;
- Let $u < v$, we denote (u, v) as an undirected edge connecting vertices u and v. Two undirected edges (u_1, v_1) and (u_2, v_2) are different, if $u_1 \neq u_2$ or $v_1 \neq v_2$.

Input

There are multiple test cases. The first line of the input contains an integer T (about 2×10^4), indicating the number of test cases. For each test case:

The first and only line contains two integers n and m ($3 \leq n \leq 10^5$, $0 \leq m \leq \frac{n(n-1)}{2}$), indicating the number of vertices and the number of edges in the graph.

It's guaranteed that the sum of n in all test cases will not exceed 3×10^7.

Output

For each test case output one line containing one integer, indicating the number of different sub-cycle graphs with n vertices and m edges modulo $10^9 + 7$.

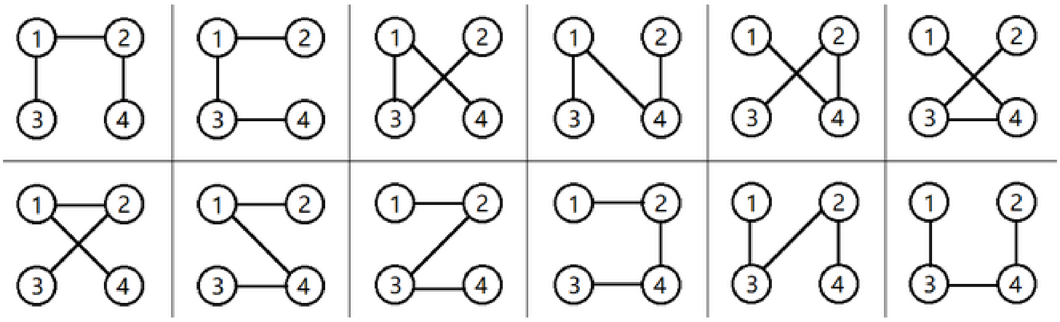
Example

standard input	standard output

3
4 2
4 3
5 3
	15
	12
	90

Note

The 12 sub-cycle graphs of the second sample test case are illustrated below.



Problem M. Function and Function

If we define $f(0) = 1, f(1) = 0, f(4) = 1, f(8) = 2, f(16) = 1, \dots$, do you know what function f means?

Actually, $f(x)$ calculates the total number of enclosed areas produced by each digit in x. The following table shows the number of enclosed areas produced by each digit:

Digit	Enclosed Area	Digit	Enclosed Area
0	1	5	0
1	0	6	1
2	0	7	0
3	0	8	2
4	1	9	1

For example, $f(1234) = 0 + 0 + 0 + 1 = 1$, and $f(5678) = 0 + 1 + 0 + 2 = 3$.

We now define a recursive function g by the following equations:

$$\begin{cases} g^0(x) = x \\ g^k(x) = f(g^{k-1}(x)) & \text{if } k \geq 1 \end{cases}$$

For example, $g^2(1234) = f(f(1234)) = f(1) = 0$, and $g^2(5678) = f(f(5678)) = f(3) = 0$.

Given two integers x and k, please calculate the value of $g^k(x)$.

Input

There are multiple test cases. The first line of the input contains an integer T (about 10^5), indicating the number of test cases. For each test case:

The first and only line contains two integers x and k ($0 \leq x, k \leq 10^9$). Positive integers are given without leading zeros, and zero is given with exactly one '0'.

Output

For each test case output one line containing one integer, indicating the value of $g^k(x)$.

Example

standard input	standard output

6
123456789 1
888888888 1
888888888 2
888888888 999999999
98640 12345
1000000000 0
	5
	18
	2
	0
	0
	1000000000

Note