

ECUST Programming Contest 2025

Main Contest Session Problemset

October 21, 2025 13:00 ~ 17:00

Problem List

ID	Problem Title	Time / Memory Limit	Notes
A	Golden Alleyway	1s/1GB	
B	Seeing is believing	1s/1GB	
C	Time Trouble	1s/1GB	
D	Mosaic Garden	1s/1GB	
E	Fortress Fall	1s/1GB	
F	The Tower (XVI)	2s/1GB	
G	Midnight Monsoon	2s/1GB	
H	Square the Circle	3s/1GB	Interactive Problem
I	Chromatic Complex	6s/1GB	
J	Attack from clone	1s/1GB	
K	Clockwork	1s/1GB	
L	Forest Path	2s/1GB	
M	FLOATING POINT	1s/1GB	
N	In Filtration	2s/1GB	
O	Nelumbo	3s/1GB	

The problemset contains a total of 15 problems and 24 pages.

If your booklet is missing any pages, please contact a runner immediately.

Please be sure to carefully read the competition rules and technical regulations included in the contest attachments.

Do not open the problemset or touch the mouse and keyboard before the contest begins.

To ensure fairness, Problems **A** **B** **C** **D** are more suitable for new contestants. The remaining problems are arranged in random order, meaning the problem difficulty is not guaranteed to increase progressively. The English version is translated by GPT-5. In case of any discrepancies, the Chinese problem statement shall prevail.

Problem **A** Golden Alleyway

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy arrives at ECUST! She steps into the legendary “Lanqiao Cup Laboratory”. After hearing stories of several legendary contestants, she wants to become one of them —but first, she decides to study a practical question.

At this school, a team that wins an ICPC gold medal may apply for a prize of ¥7,500, a silver medal brings ¥3,000, and a bronze medal brings ¥1,500.

ICPC is a team contest with up to three members (teams of one or two are also allowed). The prize money is divided equally among all team members.

In Maddy’s dream, her team has n members, wins g golds, s silvers, and b bronzes. Please compute how much prize money she personally receives.

Input

A single line contains four integers n, g, s, b ($1 \leq n \leq 3$, $0 \leq g, s, b \leq 7$), representing the team size and the numbers of gold, silver, and bronze medals.

It is guaranteed that $g + s + b \leq 7$.

Output

Output a single integer m , the amount of money (in yuan) that Maddy receives.

Sample 1

standard input	standard output
3 2 1 2	7000

Explanation for Sample 1

Maddy’s team wins two gold medals for ¥15,000, one silver for ¥3,000, and two bronzes for ¥3,000, totaling ¥21,000. Split evenly among three members, Maddy receives ¥7,000.

Sample 2

standard input	standard output
1 0 0 4	6000

Note

In programming contests, submitted programs should not print any prompt text such as “Please enter two integers:” at the beginning of the output. Doing so may result in a “Wrong Answer” or other unintended judging outcomes.

“Yuan” (元, symbol “¥”) is the currency unit used in Mainland China.

Problem **B** Seeing Is Believing

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy and Baddy are engaged in a battle of minds!
In Baddy's heart lies a hidden number a , which satisfies the following conditions:

- a is a natural number.
- $l \leq a < l + 6$.
- The remainder of a divided by 2 is m_2 , and the remainder of a divided by 3 is m_3 .

Now that Maddy knows l , m_2 , and m_3 , please help her find a possible value of a that Baddy might be thinking of. If no such number exists, report that there is no solution.

Input

A single line contains three integers l, m_2, m_3 ($0 \leq l \leq 100, 0 \leq m_2 < 2, 0 \leq m_3 < 3$), representing the range of possible values for Baddy's number, and the remainders of dividing a by 2 and 3, respectively.

Output

If a valid answer exists, output one integer a in one line — a number that Baddy could have been thinking of. If there are multiple valid answers, you may output any of them.
If no valid answer exists, output a single integer -1 .

Sample 1

standard input	standard output
0 1 2	5

Explanation for Sample 1

Let's verify step by step:

- 5 is a natural number.
- $0 \leq 5 < 0 + 6$.
- $5 \bmod 2 = 1, 5 \bmod 3 = 2$.

Therefore, 5 satisfies all of Baddy's conditions and is a possible value.

Sample 2

standard input	standard output
63 1 1	67

Note

0 is considered a natural number.

Problem C Time Trouble

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy's digital watch was smashed. Surprisingly, the digits flew out and fell onto the ground. She now picks up two numbers from the ground. Please help her form a new 24-hour time *Time* using exactly these two numbers, or determine that it is impossible.

The required 24-hour time is a string in the form **HH:MM**, and must satisfy:

- **HH** and **MM** are nonnegative integers of at most two digits. If a field has only one digit, it should be padded with a leading **0**.
- The ranges are $0 \leq HH \leq 23$ and $0 \leq MM \leq 59$.
- **HH**, **MM** must correspond to the two given numbers without omission or duplication; you may swap the order of the two given numbers.

Input

The first line contains an integer T ($1 \leq T \leq 10^4$), the number of test cases.

Each test case consists of a single line with two integers a and b ($0 \leq a, b \leq 99$), the two numbers Maddy picked up.

Output

For each test case, if it is possible to form a valid 24-hour time using the two given numbers, output one line with a string *Time* —any valid answer.

Otherwise, output a single integer -1 .

Sample 1

standard input	standard output
5	09:41
9 41	-1
9 61	09:50
50 9	00:06
0 6	-1
25 35	

Explanation for Sample 1

For the first sample:

- **09** and **41** are nonnegative integers of at most two digits, and the one-digit field is padded with a leading **0**.
- **09** and **41** satisfy the respective range constraints.
- **09**, **41** correspond one-to-one with the given numbers 9 and 41, without omission or duplication.

Note that **41:09** is not a valid answer because it violates the range constraints for the respective fields. For the second sample, it is impossible to form a valid 24-hour time regardless of the ordering.

Problem **D** Mosaic Garden

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Welcome to the ECUSTPC 2025 contest arena!

In this 4-hour competition, you will join hands with Maddy to tackle a series of interesting problems together.

Maddy's next task for you is an operation on the specific string **ECUST**. You are required to generate all strings that are equal to **ECUST** in a case-insensitive sense.

All generated strings must be pairwise distinct and printed in ascending order according to their ASCII lexicographical order.

Input

There is no input for this problem.

Technically, a random line of input will be provided by the system, but you may safely ignore it and simply output the correct result.

Output

Output several lines, each containing a string that is case-insensitively equal to **ECUST**. The strings must be sorted in ascending ASCII order.

Sample 1

This problem has no input. The sample below only illustrates the output format; it is not the complete output, and this sample will not appear in the attached files.

standard input	standard output
NO_INPUT	ECUST
	ECUS ^t
	ECUs ^T
	<lines omitted>
	ecust

Note

The ASCII lexicographical order of two strings a and b is defined as follows:

- a is a prefix of b ; or
- At the first position where a and b differ, the ASCII value of a 's character is smaller than that of b 's.

Two strings are equal in a case-insensitive sense if they become identical after converting all uppercase letters to lowercase. For example, **ECUST**, **ecust**, and **eCuSt** are considered equal under case-insensitive comparison.

The ASCII code of **0** is 48, the uppercase letter **A** is 65, and the lowercase letter **a** is 97.

Problem **E** Fortress Fall

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy arrives before the Autumn Fortress. Here she gathers n ingredients, whose freshness values are $a_1, a_2, \dots, a_n$.

Maddy will use these ingredients to make dinner over several days. In particular, because she has only two recipes with skill values x and y , each day she will cook *exactly* two dishes —having fewer than two is not allowed.

The daily cooking procedure is as follows:

- Maddy may choose two distinct, previously unused ingredients in any order. Let their freshness be a and b . She then cooks them, assigning each ingredient to one recipe. The resulting dish values are ax and by , so the total deliciousness for that day is $ax + by$. Note that Maddy may assign the recipes to the two ingredients in any order, and each day's dinner must use two different ingredients to make two dishes.

Please help Maddy choose how many days to cook and compute, under optimal choices, the total deliciousness D over those days. (Of course, if everything would taste terrible, she can choose not to cook at all; in other words, the number of days may be 0.)

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), the number of test cases.

For each test case, the first line contains three integers n , x , and y ($2 \leq n \leq 10^5$, $-10^6 \leq x, y \leq 10^6$), denoting the number of ingredients and the skill values of the two recipes.

The next line contains n integers $a_1, a_2, \dots, a_n$ ($-10^6 \leq a_i \leq 10^6$), the freshness of the ingredients.

It is guaranteed that the total $\sum n$ over all test cases does not exceed 3×10^5 .

Output

For each test case, output a single integer D , the maximum total deliciousness achievable under optimal choices.

Sample 1

standard input	standard output
6	35
4 1 2	56
4 5 6 7	0
5 2 8	23
0 -1 -2 -6 7	23
6 -1 -2	165
1 2 3 4 5 6	
6 3 4	
-1 0 0 1 2 3	
6 -3 -4	
1 0 0 -1 -2 -3	
5 1 -10	
-9 2 -5 3 -7	

Explanation for Sample 1

For the first sample, we choose to cook dinner on 2 days:

- On day one, choose ingredients with freshness 7 and 5, assign them to recipes with skill values 2 and 1, yielding $7 \times 2 + 5 \times 1 = 19$.
- On day two, choose ingredients with freshness 6 and 4, assign them to recipes with skill values 2 and 1, yielding $6 \times 2 + 4 \times 1 = 16$.

The total deliciousness is 35.

For the second sample, we cook dinner on 1 day, choosing ingredients with freshness 7 and 0, assigned to recipes with skill values 8 and 2, yielding $7 \times 8 + 0 \times 2 = 56$.

For the third sample, we choose not to cook at all, since any dinner would have negative deliciousness.

Problem **F** The Tower (XVI)

Time Limit: 2 seconds
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy arrives before a faded tower. On the stone stele below the tower, a number n is engraved. Maddy discovers a mechanism on a nearby stone. After she presses it, the stele's number changes as follows:

- Take each decimal digit of n , square it, and then concatenate the results in the original order to form a new number n' .

Curious, Maddy presses the mechanism m times. She records the original n and the resulting numbers after each press, for a total of $m + 1$ numbers:

$$n, n', n'', \dots, n^{(m)}.$$

Maddy wants to know the value ans , which is the sum of these $m + 1$ numbers taken modulo 9. Please compute ans .

Input

The first line contains an integer T ($1 \leq T \leq 10^3$), the number of test cases. Each test case consists of a single line with two integers n and m ($1 \leq n, m \leq 10^9$), denoting the initial number on the stele and the number of times Maddy presses the mechanism, respectively.

Output

For each test case, output one line with a single integer ans , the required sum modulo 9.

Sample 1

standard input	standard output
4	2
1 100	6
2 4	6
74700 1	5
2279 1	

Explanation for Sample 1

For the first test, 1 remains 1 after every press. Each term equals 1, so the total is $1 \times 101 = 101$, and $101 \bmod 9 = 2$.

For the second test, the numbers evolve as: $2 \rightarrow 4$, $4 \rightarrow 16$, $16 \rightarrow 136$, $136 \rightarrow 1936$. The sum of these five numbers modulo 9 is $(2 + 4 + 16 + 136 + 1936) \bmod 9 = 2094 \bmod 9 = 6$.

For the third test, the transformation is: $74700 \rightarrow 49164900$.

For the fourth test, the transformation is: $2279 \rightarrow 444981$.

Note

Taking a number modulo 9 means the remainder after division by 9.

Problem **G** Midnight Monsoon

Time Limit: 2 seconds
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

In front of Maddy are n pufferfish, each with a size value a_i . When pufferfish with greatly differing sizes are placed together, they explode! Therefore, Maddy decides to line them up in a row.

Let the sizes of the pufferfish from left to right be $a'_1, a'_2, \dots, a'_n$. Maddy defines the *explosiveness* of this arrangement as

$$B = \sum_{i < j, |i-j| > 1} |a'_i - a'_j|.$$

At midnight, the pufferfish change! Specifically, there will be m transformations. In each transformation, the size of one pufferfish id increases by $delt$ (if $delt < 0$, its size decreases by $|delt|$).

After each transformation, Maddy wants you to find an arrangement of the pufferfish that minimizes the explosiveness, and output the corresponding minimal B .

Note that each transformation is *cumulative* —that is, each change affects all subsequent calculations and transformations.

Also note that the size of a pufferfish may be negative.

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), denoting the number of test cases.

For each test case, the first line contains two integers n and m ($2 \leq n \leq 10^5$, $1 \leq m \leq 10^5$), representing the number of pufferfish and the number of transformations.

The next line contains n integers $a_1, a_2, \dots, a_n$ ($-10^8 \leq a_i \leq 10^8$), representing the initial sizes of the pufferfish. Note that the size values may be negative.

Then follow m lines, each containing two integers id and $delt$ ($1 \leq id \leq n$, $-10^8 \leq delt \leq 10^8$), representing the index of the modified pufferfish and the change in its size for that transformation.

It is guaranteed that within a single test case, $\sum |delt| \leq 10^8$, and across all test cases, $\sum n, \sum m \leq 3 \times 10^5$.

Output

For each test case, output one integer B after each transformation, denoting the minimal explosiveness among all possible arrangements after that transformation.

Sample 1

standard input

```
1
4 3
1 2 3 4
4 0
2 5
1 -2
```

standard output

```
3
6
8
```

Explanation for Sample 1

After the first transformation, the pufferfish sizes remain $\{1, 2, 3, 4\}$. An optimal arrangement is $\{2, 4, 1, 3\}$, giving

$$B = |a_1 - a_3| + |a_1 - a_4| + |a_2 - a_4| = |2 - 1| + |2 - 3| + |4 - 3| = 3.$$

After the second transformation, the sizes are $\{1, 7, 3, 4\}$. An optimal arrangement is $\{3, 7, 1, 4\}$, giving

$$B = |a_1 - a_3| + |a_1 - a_4| + |a_2 - a_4| = |3 - 1| + |3 - 4| + |7 - 4| = 6.$$

After the third transformation, the sizes are $\{-1, 7, 3, 4\}$. An optimal arrangement is $\{3, 7, -1, 4\}$, giving

$$B = |a_1 - a_3| + |a_1 - a_4| + |a_2 - a_4| = |3 - (-1)| + |3 - 4| + |7 - 4| = 8.$$

Problem **H** Square the Circle

Time Limit: 3 seconds
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

This is an interactive problem.

Problem Description

Maddy and Baddy are playing a mechanism-guessing game!

The mechanism is a 2D figure. Maddy hides one mechanism in the 2D Cartesian plane, which is one of the following two types:

- **Bumper:** a lattice circle centered at $(0,0)$ that passes through at least one lattice point $P = (x,y)$ such that $0 \leq x, y \leq 10^5$ and at least one of x, y is positive.
- **Kevin:** a lattice square centered at $(0,0)$, whose vertices are all lattice points (integer coordinates). All coordinate absolute values are less than 10^5 . The square is guaranteed to be non-degenerate, i.e., its side length is at least 1.

Baddy does not know which mechanism it is. Each time, she may ask Maddy one query:

- Is the lattice point (x,y) *outside* the figure represented by the mechanism? (Points on the boundary are considered *inside*.)
- Maddy answers whether the point is outside or inside. If the point is on the boundary, Maddy answers that the point is inside.

Baddy may ask at most 35 queries. She must then tell Maddy:

- Whether the mechanism is *Kevin*, *Bumper*, or whether both are possible.

Note that giving your final answer does *not* count toward the 35-query limit.

Interaction

The first line contains an integer T ($1 \leq T \leq 10^3$), the number of test cases.

For each test case the interaction begins immediately. For each query, you must communicate with the interactor in the following format:

- Output one line starting with the character `?` to indicate a query, followed by two integers x and y asking whether the point (x,y) is outside the mechanism. You must ensure $0 \leq |x|, |y| \leq 2 \times 10^5$.
- If your query is valid and you have not exceeded the query limit, the interactor responds with one line containing a string Q . If $Q = \text{IN}$, then (x,y) is inside or on the boundary; if $Q = \text{OUT}$, then (x,y) is outside.
- If you exceed the allowed number of queries, the interactor outputs an integer -1 and terminates the interaction. Your verdict will be *Wrong Answer*.

Once you have determined the answer, output it in the following format:

- Output one line starting with the character `!`.
- Then output a string S , where S must be one of **Bumper**, **Kevin**, or **NotConfirm**, indicating respectively that the mechanism is a Bumper, a Kevin, or that both are possible.

If your answer is correct, the interactor will output one line with the string **Correct!**; otherwise, it will output one line with the integer -2 and terminate immediately, and your verdict will be *Wrong Answer*. After finishing one test case, you should immediately begin querying for the next one. If it was the last test case, you may exit safely.

The interactor is *not* adaptive, meaning that the hidden mechanism for each test case is fixed before the interaction begins and does not change during the interaction.

Every output must be followed by a newline and a flush, otherwise you may get unexpected results.

If you receive one of the interactor error codes -1 or -2 , terminate your interaction promptly; otherwise you may encounter unexpected results (such as *Time Limit Exceeded*).

To flush the output buffer, you may:

- For C or C++, use `fflush(stdout)` or `cout.flush()`.
- For Java or Kotlin, use `System.out.flush()`.
- For Python, use `stdout.flush()`.

Sample 1

interactor input	contestant output
1	? 1 0
IN	? 0 1
IN	? 100 100
OUT	? 0 2
IN	! Bumper
Correct!	

Note

The sample interaction is for reference only. Actual interactions are not unique, and the shown sequence may not be feasible or optimal. This problem's sample will not appear in the attached files.

There exist two possible answers *if and only if*:

- There is a Kevin shape K with covered lattice set S_K and a Bumper shape B with covered lattice set S_B such that both equal the current hidden mechanism's covered lattice set S_{hidden} .

In this case, it is impossible to distinguish the two shapes by querying, and you should output **NotConfirm**. However, if no such pair of mechanisms exists with identical covered sets, then the mechanism type is uniquely determined by querying; in that case, outputting **NotConfirm** will be judged *Wrong Answer*.

Problem **I** Chromatic Complex

Time Limit: 6 seconds
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy has found a brand-new sword and decides to decorate it with some *gems*. In the mysterious Celeste Mountain region, there are C distinct *gems* scattered around.

Maddy wants her sword to look extremely colorful, so she needs to collect at least k *gems*.

Before setting out, she obtains a map that records the detailed terrain of Celeste Mountain, described as follows:

- The region is divided into an $n \times m$ grid, where each cell is indexed by coordinates from $(1,1)$ to (n,m) .
- For a cell (i,j) , its adjacent cells are $(i+1,j)$, $(i-1,j)$, $(i,j+1)$, and $(i,j-1)$. Note that any adjacent cell must also lie within the region, i.e., its coordinates must satisfy $1 \leq i \leq n$ and $1 \leq j \leq m$.
- Each cell is one of three terrains: land, water, or lava.
- Exactly C cells contain *gems*, and *gems* can only appear on land or water cells.

Maddy starts from a given cell S and needs to collect at least k *gems* in this area. Her actions consume *stamina*, governed by the rules below:

- Maddy initially stands at the given start cell S , which is guaranteed to be land or water and to contain no *gem*.
- Each move, she may move to an adjacent cell whose terrain is land or water; she cannot enter lava cells.
- If both the starting and ending cells of the move are *land*, then this move costs no *stamina*.
- Otherwise (i.e., the move involves *water* —whether moving from water, moving into water, or water $\leftrightarrow$ water), the move costs 1 point of *stamina*.

Please tell Maddy the minimum *stamina* stm required to collect at least k *gems*, or report that it is impossible to collect k *gems*.

Input

The first line contains an integer T ($1 \leq T \leq 100$), the number of test cases.

For each test case, the first line contains four integers n, m, C , and k ($1 \leq n, m \leq 2 \times 10^3$, $1 \leq k \leq C \leq 15$), denoting the grid dimensions, the total number of *gems* on the map, and the number of *gems* Maddy needs. The next line contains two integers S_x and S_y ($1 \leq S_x \leq n$, $1 \leq S_y \leq m$), the row and column of Maddy's starting position.

Then follow n lines, each a string of length m , $S_1, S_2, \dots, S_n$, where $S_i = \overline{s_{i,1}s_{i,2} \dots s_{i,m}}$ describes the terrain of row i . For any $1 \leq i \leq n$, $1 \leq j \leq m$:

- If $s_{i,j} = 0$, the cell is land;
- If $s_{i,j} = 1$, the cell is water;
- If $s_{i,j} = 2$, the cell is lava.

Then C lines follow; each contains two integers x, y ($1 \leq x \leq n$, $1 \leq y \leq m$), giving the coordinates of a cell containing a *gem*.

It is guaranteed that in each test case, none of the *gems* nor the starting cell lies on lava, and all these coordinates are pairwise distinct. It is also guaranteed that the sum of all $n \cdot m$ over the input does not exceed 4×10^6 , and at most 5 test cases satisfy $C > 10$.

Output

For each test case:

- If Maddy can collect at least k *gems*, output a single integer stm , the minimum *stamina* required;
- Otherwise, output a single integer -1 .

Sample 1

standard input	standard output
2	2
3 5 3 2	-1
1 1	
00000	
11211	
11011	
1 4	
3 1	
3 5	
1 5 1 1	
1 1	
01121	
1 5	

Explanation for Sample 1

For the first sample, the illustration is as follows:

	1	2	3	4	5
1	起点			宝石 1	
2					
3	宝石 2				宝石 3

One optimal route has transitions

$$(1, 1) \rightarrow (1, 2) \rightarrow (1, 3) \rightarrow (1, 4) \rightarrow (1, 3) \rightarrow (1, 2) \rightarrow (1, 1) \xrightarrow{1 \text{ stamina}} (2, 1) \xrightarrow{1 \text{ stamina}} (3, 1).$$

Here in the figure, “宝石” means *gem*, “起点” means starting position.

Problem Attack from clone

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input : Standard Input
Output: Standard Output

Problem Description

Maddy and Baddy decide to compete with wits!
They start with a multiset S of n positive integers:

$$S = \{a_1, a_1 + d, a_1 + 2d, \dots, a_1 + (n - 1)d\}.$$

They play the following game:

- Maddy moves first, and the players alternate.
- On each move, the current player inserts one positive integer into S .
- The game ends after each player has moved exactly m times.
- Maddy aims to make the final k -connectivity count of S as large as possible, while Baddy aims to make it as small as possible.

The k -connectivity count is defined as follows:

- Sort S . For every pair of adjacent numbers, if their difference is greater than k , increase the count by 1, with the **initial count being 1**.
- Equivalently, view the elements of S as vertices of a graph; connect two vertices if their absolute difference is at most k . The k -connectivity count equals the number of connected components.

Assuming both Maddy and Baddy are perfectly rational and always choose optimal strategies, determine the k -connectivity count of S at the end of the game.

Input

The first line contains an integer T ($1 \leq T \leq 100$), the number of test cases.
For each test case, a single line contains five integers n, a_1, d, m , and k ($1 \leq n, a_1, m \leq 100, 0 \leq d, k \leq 100$), denoting the initial size of S , the first term and common difference of the arithmetic progression, the number of moves per player, and the parameter k , respectively.

Output

For each test case, output a single integer *ans* —the k -connectivity count of S at the end of the game under optimal play by both players.

Sample 1

standard input	standard output
4	3
3 2 3 2 2	6
4 3 2 5 1	6
4 1 1 5 2	12
9 7 8 3 0	

Explanation for Sample 1

For the first test case, the initial

$$S = \{(2 + 0 \times 3), (2 + 1 \times 3), (2 + 2 \times 3)\} = \{2, 5, 8\}.$$

One possible sequence of optimal moves is:

- Maddy inserts 13, so $S = \{2, 5, 8, 13\}$;
- Baddy inserts 3, so $S = \{2, 3, 5, 8, 13\}$;
- Maddy inserts 18, so $S = \{2, 3, 5, 8, 13, 18\}$;
- Baddy inserts 7, so $S = \{2, 3, 5, 7, 8, 13, 18\}$.

After sorting, the adjacent differences are $D_1 = 1$, $D_2 = 2$, $D_3 = 2$, $D_4 = 1$, $D_5 = 5$, $D_6 = 5$. Among them, 2 differences exceed $k = 2$, so the *k-connectivity count* is 3.

Problem **K** Clockwork

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy plans to rebuild a clock tower...

Maddy has a sequence $\{a_i\}$ of length n . She wants to transform an initially all-zero sequence $\{h_i\}$ of length n into $\{a_i\}$ using the minimum number of operations. Each operation is defined as follows:

- Maddy chooses an index k ($1 \leq k \leq n$) and a direction $d \in \{L, R\}$.
- If $d = L$, then for all $1 \leq i \leq k$, set $h_i \leftarrow |i - k| + 1$.
- If $d = R$, then for all $k \leq i \leq n$, set $h_i \leftarrow |i - k| + 1$.
- Note that each operation completely overwrites the corresponding interval.

Please determine the minimum number of operations required. If it is impossible to obtain $\{a_i\}$ via the above operations, report that there is no solution.

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), the number of test cases.

For each test case, the first line contains an integer n ($1 \leq n \leq 10^5$), the length of the sequence.

The next line contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq n$), which is the target sequence $\{a_i\}$ that Maddy wants to construct.

It is guaranteed that the sum of n over all test cases does not exceed 3×10^5 .

Output

For each test case, if it is possible to obtain $\{a_i\}$ using the above operations, output a single integer — the minimum number of operations required.

Otherwise, output a single integer -1 .

Sample 1

standard input	standard output
4	3
6	-1
1 2 1 1 2 3	6
3	6
3 3 3	
6	
1 1 1 1 1 1	
11	
1 2 1 3 4 1 2 1 1 2 3	

Explanation for Sample 1

For the first sample, one optimal sequence of operations is:

1. Choose $k = 4$, $d = R$, obtaining $\{h_i\} = \{0, 0, 0, 1, 2, 3\}$;
2. Choose $k = 3$, $d = L$, obtaining $\{h_i\} = \{3, 2, 1, 1, 2, 3\}$;

3. Choose $k = 1$, $d = L$, obtaining $\{h_i\} = \{1, 2, 1, 1, 2, 3\}$.

It is easy to verify that this uses the minimum possible number of operations.

For the second sample, no matter what operations Maddy performs, h_2 can never become 3, so the answer is -1 .

Problem **L** Forest Path

Time Limit: 2 seconds
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy got lost in the woods... and she saw a peculiar tree.

In front of her stands a tree of size n —that is, an undirected, connected, acyclic graph T consisting of n vertices and $n - 1$ edges.

However, Maddy accidentally caused the tree's shape to change: a new branch appeared, connecting two **distinct** vertices that were originally **not adjacent**! In other words, the tree T has now become an undirected connected graph G' with n vertices and n edges.

To test you, Maddy provides the original tree structure and the values of $\sum D_i$ for each vertex, where $\sum D_i$ is the sum of distances from vertex i to all other vertices in the new graph G' , i.e.

$$\sum D_i = \sum_{j \in V} D_{G'}(i, j).$$

Your task is to determine which two vertices are connected by the newly grown branch.

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), the number of test cases.

For each test case:

- The first line contains an integer n ($3 \leq n \leq 10^5$), the number of vertices. - The next $n - 1$ lines each contain two integers u and v ($1 \leq u, v \leq n, u \neq v$), indicating that there is an edge between u and v in the tree T . - The following line contains n integers $\sum D_1, \sum D_2, \dots, \sum D_n$, where $\sum D_i$ is the sum of distances from vertex i to all other vertices in G' .

It is guaranteed that: - The given graph in each test case forms a valid tree. - The hidden edge connects two distinct and non-adjacent vertices in the original tree. - The total $\sum n$ across all test cases does not exceed 3×10^5 .

Output

For each test case, output two integers x and y , representing the two vertices connected by the newly added branch.

If multiple valid answers exist, output any one of them. (For example, if $1\ 2$ is valid, then $2\ 1$ is also valid.)

Sample 1

standard input

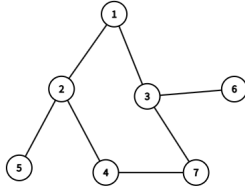
```
1
7
1 2
1 3
2 4
2 5
3 6
3 7
10 10 10 11 15 15 11
```

standard output

```
| 4 7
```

Explanation for Sample 1

The following figure shows the structure of G' :



Note

The distance $D_{i,j}$ between two vertices is defined as the minimum number of edges among all paths from vertex i to vertex j .

Problem **M** FLOATING POINT

Time Limit: 1 second
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy encounters a mysterious machine. To obtain the Crystal Heart, she must speak aloud a long sequence of numbers!

The machine first provides Maddy with an integer n . The sequence must satisfy the following conditions:

1. The sequence consists of n integers within the range $[0, 2^{30})$, denoted as $a_1, a_2, \dots, a_n$.
2. For all $1 \leq i < n$, it must hold that $a_i < a_{i+1}$ and $a_i \text{ XOR } a_{i+1} < a_i \text{ AND } a_{i+1}$.

Help Maddy construct such a sequence, or determine that no valid sequence exists.

Input

The first line contains an integer T ($1 \leq T \leq 10^3$), the number of test cases.

Each test case consists of a single integer n ($2 \leq n \leq 10^6$), representing the length of the sequence required by the machine.

It is guaranteed that the total sum of n over all test cases does not exceed 10^6 .

Output

For each test case, if such a sequence exists, output one line containing n integers —any valid sequence satisfying the machine's requirements.

If no such sequence exists, output a single integer -1 .

If multiple valid sequences exist, you may output any of them.

Sample 1

standard input	standard output
4	9 15
2	4 5 6
3	9 10 12 15
4	8 9 10 11 12
5	

Explanation for Sample 1

For the first test case, $a_1 \text{ XOR } a_2 = 9 \text{ XOR } 15 = 6$, $a_1 \text{ AND } a_2 = 9 \text{ AND } 15 = 9$, so $a_1 \text{ XOR } a_2 < a_1 \text{ AND } a_2$, and clearly $a_1 < a_2$.

For the second test case:

- $a_1 \text{ XOR } a_2 = 4 \text{ XOR } 5 = 1$, $a_1 \text{ AND } a_2 = 4 \text{ AND } 5 = 4$, hence $a_1 \text{ XOR } a_2 < a_1 \text{ AND } a_2$, and $a_1 < a_2$.
- $a_2 \text{ XOR } a_3 = 5 \text{ XOR } 6 = 3$, $a_2 \text{ AND } a_3 = 5 \text{ AND } 6 = 4$, hence $a_2 \text{ XOR } a_3 < a_2 \text{ AND } a_3$, and $a_2 < a_3$.

Note

For two integers a and b , $a \text{ XOR } b$ denotes the bitwise XOR, i.e., a bit in the result is 1 if and only if exactly one of a or b has a 1 in that bit position.

For two integers a and b , $a \text{ AND } b$ denotes the bitwise AND, i.e., a bit in the result is 1 if and only if both a and b have a 1 in that bit position.

Problem **N** In Filtration

Time Limit: 2 seconds
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy will command a king to purge Baddy's army of evil rooks...

Specifically, they are playing a game on an infinitely large chessboard.

Baddy controls n black rooks. These rooks remain fixed at their initial positions throughout the game, located at $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$.

Maddy chooses a position (X, Y) to deploy her white king. The king can move to any of the eight adjacent squares; for example, if the king is on square (p, q) , it can move to $(p-1, q-1), (p-1, q), (p-1, q+1), (p, q-1), (p, q+1), (p+1, q-1), (p+1, q),$ or $(p+1, q+1)$.

A black rook attacks any square that lies in the same row or column as itself, provided that there are no other pieces blocking the path. The square occupied by the rook itself is not included in its attack range. Maddy's goal is to control the white king to capture all of Baddy's black rooks. Specifically:

- The white king may move according to the above rules, but it cannot enter any square that lies within the attack range of a black rook; otherwise, the king will be captured.
- If the king moves onto a square occupied by a black rook, that rook will be captured and removed from the board, and it will no longer be able to attack.

Maddy's goal is to capture all n black rooks. Please determine whether there exists a sequence of captures and movements that allows the king to eventually capture all the rooks.

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), denoting the number of test cases.

Each test case consists of the following input:

The first line contains an integer n ($1 \leq n \leq 10^5$), denoting the number of black rooks.

The second line contains two integers X and Y ($1 \leq X, Y \leq 10^9$), denoting the starting position of Maddy's white king.

Then follow n lines, each containing two integers x_i and y_i ($1 \leq x_i, y_i \leq 10^9$), denoting the coordinates of Baddy's i -th black rook.

It is guaranteed that the total $\sum n \leq 3 \times 10^5$, all black rooks' coordinates (x_i, y_i) are distinct, and that initially, the white king is not within any rook's attack range.

Output

For each test case, output a single line containing one string **YES** if the white king can eventually capture all black rooks, or **NO** otherwise.

The judging is case-insensitive, meaning that **yes**, **YES**, **Yes**, or **YeS** will all be accepted as correct when the answer is affirmative.

Sample 1

standard input

```
4
1
4 6
5 7
2
1 1
8 2
8 9
2
4 1
2 3
7 8
4
7 10
2 11
5 3
9 20
14 6
```

standard output

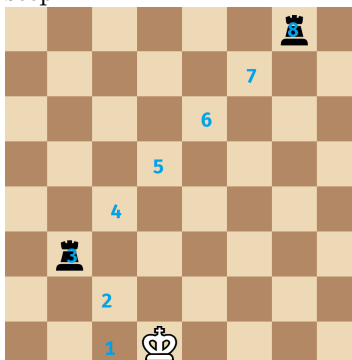
```
YES
NO
YES
NO
```

Explanation for Sample 1

For the first sample, the king can capture the black rook in one move.

For the third sample, the chessboard layout is shown below. The first coordinate increases from left to right, and the second coordinate increases from bottom to top. The bottom-left square is (1, 1).

A possible valid sequence is illustrated below, where the blue numbers indicate the king's position at each step.



Problem 0 Nelumbo

Time Limit: 3 seconds
Memory Limit: 1 Gigabyte
Input: Standard Input
Output: Standard Output

Problem Description

Maddy encounters blooming lotuses in a pond, but something about them feels strangely eerie...

In this lotus pond, there is a tree with n nodes —an undirected, connected, acyclic graph consisting of $n - 1$ edges.

Each node i ($1 \leq i \leq n$) has a weight a_i . A k -pair (x, y) on the tree is defined as follows:

- $1 \leq x < y \leq n$, where x and y are nodes of the tree.
- Let S be the multiset of weights of all nodes along the unique path between x and y (both a_x and a_y are included).
- The pair (x, y) is called a k -pair if $\gcd(S) = \min(S) = k$, that is, the greatest common divisor of all values in S equals their minimum value, and both equal k .

Maddy randomly chooses two distinct nodes (x, y) . If there exists some k such that (x, y) is a k -pair, she will receive k lanterns.

Please help Maddy find the expected number of lanterns q , and output its value modulo 998,244,353. Details on how to output are provided below.

Input

The first line contains an integer T ($1 \leq T \leq 10^5$), the number of test cases.

Each test case contains the following:

The first line contains an integer n ($2 \leq n \leq 10^5$), the number of nodes in the tree.

The next $n - 1$ lines each contain two integers u and v ($1 \leq u, v \leq n, u \neq v$), indicating that there is an edge between nodes u and v in the tree T .

The next line contains n integers $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq n$), representing the weights of the nodes.

It is guaranteed that the sum of all n over all test cases does not exceed 3×10^5 , and each test case forms a valid tree.

Output

For each test case, output a single integer ans , representing the expected number of lanterns Maddy will receive, modulo 998,244,353.

Sample 1

standard input	standard output
1	332748119
6	
1 2	
1 3	
2 4	
2 5	
3 6	
6 2 3 4 2 1	

Explanation for Sample 1

The given tree has 6 nodes and the following edges:

$$1-2, 1-3, 2-4, 2-5, 3-6,$$

with weights:

$$a_1 = 6, a_2 = 2, a_3 = 3, a_4 = 4, a_5 = 2, a_6 = 1.$$

There are $\binom{6}{2} = 15$ distinct node pairs in total. We check for each pair whether the multiset S of weights along their path satisfies $\gcd(S) = \min(S)$.

For example:

- Pair (1, 2): Path [6, 2], $\min = 2$, $\gcd = 2$, hence $k = 2$.
- Pair (1, 3): Path [6, 3], $\min = 3$, $\gcd = 3$, hence $k = 3$.
- Pair (1, 6): Path [6, 3, 1], $\min = 1$, $\gcd = 1$, hence $k = 1$.
- Pair (3, 4): Path [3, 6, 2, 4], $\min = 2$, $\gcd = 1$, does not satisfy the condition.

Final statistics:

$$k = 1 : 5 \text{ pairs: } (1, 6), (2, 6), (3, 6), (4, 6), (5, 6),$$

$$k = 2 : 6 \text{ pairs: } (1, 2), (1, 4), (1, 5), (2, 4), (2, 5), (4, 5),$$

$$k = 3 : 1 \text{ pair: } (1, 3),$$

$$k = 4, 5, 6 : 0 \text{ pairs.}$$

Thus, the expected number of lanterns is:

$$\frac{5 \times 1 + 6 \times 2 + 1 \times 3 + 0 \times 4 + 0 \times 5 + 0 \times 6}{15} = \frac{4}{3}.$$

Taking the result modulo 998,244,353 gives the answer **332748119**.

Note

It can be proven that the answer will always be a rational number $\frac{p}{q}$, where p and q are coprime. You need to output an integer ans satisfying $0 \leq ans < 998,244,353$ and

$$q \cdot ans \equiv p \pmod{998,244,353}.$$

It can also be proven that such an ans always exists under the problem's constraints.