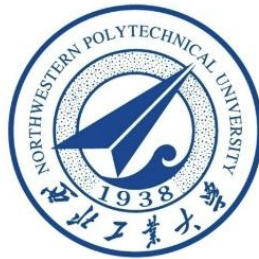


The 2026 ICPC Asia Xi'an Regional Contest

Contest Session

October 18, 2026



西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

Problem List

A	And Revive The Melody
B	Break the Speakers
C	Cryptarithm
D	Don't Fight The Music
E	Ether Strike
F	False Amber
G	Gungnir Fracture
H	Hydra
I	Incyde
J	Journey
K	Kikai No Tsubasa
L	Lucid Traveler
M	Melodiniq

This problem set should contain 13 (thirteen) problems on 25 (twenty-five) numbered pages.

Please inform a runner immediately if something is missing from your problem set.

Hosted By



西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

Problem Set Prepared By

Cfz 出题组

Cfz Problem Setting Group



Problem A. And Revive The Melody

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 1024 megabytes

Setsuna is studying music theory. There are n distinct pitches, numbered $1, 2, \dots, n$. *Setsuna* wants to compose a melody P in which every pitch appears exactly once. We call a sub-melody Q **compact** if and only if there exist integers $l \leq r$ such that the set of pitches appearing in Q is exactly $\{l, l+1, \dots, r\}$.

To compose the harmony, *Setsuna* specifies some requirements using two strings a, b , each of length n :

- For the sub-melody $P[1, i]$ consisting of the first i pitches:
 - If $a_i = 1$, this sub-melody must be compact.
 - If $a_i = 0$, this sub-melody must not be compact.
 - If $a_i = ?$, there is no restriction on this sub-melody.
- For the sub-melody $P[i, n]$ consisting of the last $n+1-i$ pitches:
 - If $b_i = 1$, this sub-melody must be compact.
 - If $b_i = 0$, this sub-melody must not be compact.
 - If $b_i = ?$, there is no restriction on this sub-melody.

After solving Don't Fight The Music, *Setsuna* wants to enumerate all melodies that satisfy the requirements. You need to help *Setsuna* find the number of such melodies so that she can check her result. Since the answer may be large, output it modulo 998244353.

Input

This problem contains multiple test cases.

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

For each test case:

- The first line contains an integer n ($2 \leq n \leq 2 \times 10^5$).
- The next two lines each contain a string of length n , namely a and b .

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5 , and that the strings contain only the characters 1, 0, and ?.

Output

For each test case, output one line containing an integer representing the answer.



Example

standard input	standard output
4	2
2	0
?1	8
1?	36
3	
???	
0??	
4	
??1?	
?0??	
5	
1??01	
10??1	

Note

For the third test case, all melodies that satisfy the requirements are listed below:

- [2, 1, 3, 4]
- [2, 3, 1, 4]
- [2, 3, 4, 1]
- [2, 4, 3, 1]
- [3, 1, 2, 4]
- [3, 2, 1, 4]
- [3, 2, 4, 1]
- [3, 4, 2, 1]



Problem B. Break The Speakers

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Milk has a tree with n nodes. Node i has a non-negative integer weight a_i . *Shama* wants to destroy the tree. She may perform one of the following three operations:

- Choose a node whose weight is 0 and whose degree is ≤ 1 , then delete the node and its incident edge, if any.
- Choose a node whose weight is nonzero and set $a_i \leftarrow a_i - 1$.
- Choose a node and set its weight to the MEX of the weights of all its neighboring nodes.

Shama wants to delete all nodes as quickly as possible. You need to help her find the minimum number of operations required to achieve this goal.

Recall that the MEX of a set is the smallest non-negative integer that does not occur in the set.

Input

This problem contains multiple test cases.

The first line contains a positive integer T ($1 \leq T \leq 2 \times 10^4$), representing the number of test cases.

For each test case:

- The first line contains an integer n ($1 \leq n \leq 2 \times 10^5$).
- The next $n - 1$ lines each contain two integers u_i, v_i ($1 \leq u_i, v_i \leq n$), representing an edge of the tree.
- The last line contains n integers a_i ($0 \leq a_i \leq 10^9$).

It is guaranteed that, in each test case, the given $n - 1$ edges form a tree, and that the sum of n over all test cases does not exceed 2×10^5 .

Output

For each test case, output one line containing an integer representing the answer.

Example

standard input	standard output
1 5 1 2 1 3 2 4 2 5 1 0 0 4 2	9

Note

One optimal strategy for the sample is as follows:



- Delete node 3.
- Decrease a_1 by 1, then delete node 1.
- Apply the MEX operation once to a_4 , then decrease a_4 by 1 and delete node 4.
- Delete node 2.
- Apply the MEX operation once to a_5 , then delete node 5.

This uses 9 operations in total.



Problem C. Cryptarithm

Input file: `standard input`
Output file: `standard output`
Time limit: 2 seconds
Memory limit: 1024 megabytes

Ris is trying to enter a password using a mouse. The password she needs to enter is an integer sequence b of length m , while her clipboard contains a sequence a of length n . To complete her task, *Ris* may use the following three operations to modify a workspace c that is initially empty:

- Paste: pay a cost of x and append every element of a , in order, to the end of c .
- Scroll: pay a cost of y and increase or decrease one element of c by 1.
- Delete: pay a cost of z and delete one element from c .

You need to help *Ris* find the minimum total cost among all ways to transform c into b .

Input

This problem contains multiple test cases.

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

For each test case:

- The first line contains five integers n, m, x, y, z ($1 \leq n, m \leq 3000, 0 \leq x, y, z \leq 10^6$).
- The second line contains n integers a_i ($0 \leq a_i \leq 10^9$).
- The third line contains m integers b_i ($0 \leq b_i \leq 10^9$).

It is guaranteed that the sum of $n \cdot m$ over all test cases does not exceed 9×10^6 , and that the sum of $n + m$ over all test cases does not exceed 2×10^5 .

Output

For each test case, output one line containing an integer representing the answer.

Example

standard input	standard output
4	63
6 7 3 2 5	15872
1 1 4 5 1 4	0
1 9 1 9 8 1 0	1000000001000000
3 2 5 6 7	
11 45 14	
1919 810	
1 7 0 0 0	
998244353	
1 0 1 0 0 0 0	
1 1 1000000 1000000 1000000	
0	
1000000000	



Note

For the first test case, one optimal sequence of operations is as follows:

- Paste twice. Then c becomes $[1, 1, 4, \mathbf{5}, \mathbf{1}, \mathbf{4}, 1, 1, 4, \mathbf{5}, \mathbf{1}, 4]$.
- Delete 5 elements, keeping the bold elements from the previous line. Then c becomes $[1, \mathbf{5}, 1, \mathbf{4}, \mathbf{5}, 1, \mathbf{4}]$.
- Scroll 16 times to adjust the bold elements from the previous line. Then c becomes $[1, 9, 1, 9, 8, 1, 0]$.

The total cost is $2 \cdot 3 + 16 \cdot 2 + 5 \cdot 5 = 63$.



Problem D. Don't Fight The Music

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Setsuna is studying music theory. There are n distinct pitches, numbered $1, 2, \dots, n$. *Setsuna* wants to compose a melody P in which every pitch appears exactly once. We call a sub-melody Q **compact** if and only if there exist integers $l \leq r$ such that the set of pitches appearing in Q is exactly $\{l, l+1, \dots, r\}$.

To compose the harmony, *Setsuna* specifies some requirements using two strings a, b , each of length n :

- For the sub-melody $P[1, i]$ consisting of the first i pitches:
 - If $a_i = 1$, this sub-melody must be compact.
 - If $a_i = 0$, this sub-melody must not be compact.
 - If $a_i = ?$, there is no restriction on this sub-melody.
- For the sub-melody $P[i, n]$ consisting of the last $n+1-i$ pitches:
 - If $b_i = 1$, this sub-melody must be compact.
 - If $b_i = 0$, this sub-melody must not be compact.
 - If $b_i = ?$, there is no restriction on this sub-melody.

Before solving And Revive The Melody, *Setsuna* wants to find one melody that satisfies all the requirements. You need to help *Setsuna* find **any** such melody, or report that no such melody exists.

Input

This problem contains multiple test cases.

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

For each test case:

- The first line contains an integer n ($2 \leq n \leq 2 \times 10^5$).
- The next two lines each contain a string of length n , namely a and b .

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5 , and that the strings contain only the characters 1, 0, and ?.

Output

For each test case, output one line:

- If no solution exists, output the integer -1 .
- Otherwise, output n integers $P_1, P_2, \dots, P_n$ representing a valid answer.



Example

standard input	standard output
4	1 2
2	-1
?1	2 3 1 4
1?	2 1 3 5 4
3	
???	
0??	
4	
??1?	
?0??	
5	
1??01	
10??1	

Note

For the third test case, all constrained sub-melodies are listed below:

- $[2, 3, 1]$ is compact.
- $[3, 1, 4]$ is not compact.



Problem E. Ether Strike

Input file: standard input
Output file: standard output
Time limit: 6 seconds
Memory limit: 1024 megabytes

*An intense dissatisfaction with the world,
and a compulsion to do something about it...*

— Akira Complex, *Ether Strike*

Hikari has collected n glass shards, numbered from 1 to n . Each shard has 64 binary attributes. The k -th bit describes the k -th attribute, so the **attribute value** of a shard is an integer in $[0, 2^{64})$.

Then, m events happen to the “world”. Each event is one of the following:

1. A fracture of the first type has its own attribute value x and affects every shard whose index is in $[l, r]$. If an affected shard differs from the fracture in **more than** d attributes, it is forcibly transformed into a shard with attribute value x .
2. *Hikari* needs to deal with the shards whose indices are in $[l, r]$. You need to tell her the sum of their attribute values.
3. A fracture of the second type has its own attribute value x and affects every shard whose index is in $[l, r]$. Every affected shard is forcibly transformed into a shard with attribute value x .

Input

The first line contains three integers n, m, d ($1 \leq n, m \leq 2 \times 10^5, d = 3$), representing the number of shards, the number of events, and the attribute threshold, respectively.

The second line contains n integers a_i ($0 \leq a_i < 2^{64}$), representing the initial attribute value of each shard.

The next m lines each begin with an integer opt , representing the type of the event:

- If $\text{opt} = 1$, three more integers l, r, x follow ($1 \leq l \leq r \leq n, 0 \leq x < 2^{64}$), describing a fracture of the first type.
- If $\text{opt} = 2$, two more integers l, r follow ($1 \leq l \leq r \leq n$), describing a query from *Hikari*; since the answer is large, you have to output it modulo 2^{64} .
- If $\text{opt} = 3$, three more integers l, r, x follow ($1 \leq l \leq r \leq n, 0 \leq x < 2^{64}$), describing a fracture of the second type.

It is guaranteed that at least one event has $\text{opt} = 2$.

Output

For each query from *Hikari*, output one line containing an integer representing the sum of the attribute values modulo 2^{64} .



Examples

standard input	standard output
6 11 3 0 7 15 56 9223372036854775808 18446744073709551615 2 1 6 1 1 6 0 2 1 6 3 2 5 18446744073709551615 2 1 6 1 1 4 17293822569102704640 2 1 6 1 3 6 18446744073709551615 2 1 6 3 1 3 1 2 1 6	9223372036854775885 9223372036854775871 18446744073709551612 13835058055282163711 16140901064495857660 0
8 10 3 56 0 7 63 9223372036854775808 9223372036854775815 17293822569102704640 15 2 1 8 1 1 4 0 1 1 4 7 2 1 4 1 5 8 17293822569102704640 2 5 8 3 3 6 18446744073709551615 2 1 8 1 1 8 18446744073709551615 2 1 8	17293822569102704788 14 5764607523034234880 16140901064495857667 18446744073709551608

Note

To ensure that the PDF is displayed correctly, several blank lines have been inserted into the last sample. You can download the original sample from Domjudge.

For the first sample, let $H = 2^{63}$, $U = 2^{64} - 1$, and $F = 15 \cdot 2^{60}$.

- Initially, the sequence is $[0, 7, 15, 56, H, U]$.
- In the second event, the numbers of differing attributes are 0, 3, 4, 3, 1, 64, respectively. Since $d = 3$, the sequence becomes $[0, 7, 0, 56, H, 0]$.
- After the fourth event, the sequence is $[0, U, U, U, U, 0]$.
- After the sixth event, the sequence is $[F, F, F, F, U, 0]$.
- After the eighth event, the sequence is $[F, F, U, U, U, U]$.
- After the tenth event, the sequence is $[1, 1, 1, U, U, U]$.

Each query asks for the sum modulo 2^{64} . In particular, the last answer is 0 because $3 + 3(2^{64} - 1) = 3 \cdot 2^{64}$.



Problem F. False Amber

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Ras and *Salt* are playing a simplified version of Texas Hold'em called CAFE MiLK Poker. The game uses $2n$ cards with ranks $1 \sim 2n$, and proceeds as follows:

- One chip is placed in the center of the table. Each player has m chips and n cards.
- *Ras* and *Salt* each draw one card **uniformly at random** from their own deck.
- Both players know the two complete decks and the card they drew, but neither knows the card drawn by the other player.
- *Ras* may choose to **check** or go **all-in**.
- If *Ras* checks, both players reveal their cards, and the player with the higher card takes the chip on the table.
- If *Ras* goes all-in:
 - *Salt* may choose to **fold** or **call**.
 - If *Salt* folds, *Ras* takes the chip on the table.
 - If *Salt* calls, both players reveal their cards, and the player with the higher card takes a total of $m + 1$ chips from the table and the other player.

Ras and *Salt* may each use a **mixed strategy** $\pi = p_i$: whenever the player makes a decision while holding card i , they choose one action with probability p_i and the other action with probability $(1 - p_i)$. Both players want to maximize their own **expected profit**, defined as the number of chips they have at the end of the game minus m .

You are *Chiffon*, *Ras*'s best friend, and you know that *Salt* can read minds: although she cannot know which card *Ras* drew, she can know *Ras*'s strategy! Therefore, you need to help *Ras* find a strategy that maximizes her expected profit. Formally, you need to choose a mixed strategy π_r for *Ras* that maximizes *Ras*'s expected profit when *Salt* may choose her strategy π_s based on π_r .

Input

This problem contains multiple test cases.

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

For each test case:

- The first line contains two integers n, m ($1 \leq n \leq 2 \times 10^5, 1 \leq m \leq 10^9$).
- The second line contains n integers a_i ($1 \leq a_i \leq 2n$), representing *Ras*'s cards.
- The third line contains n integers b_i ($1 \leq b_i \leq 2n$), representing *Salt*'s cards.

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5 , and that all a_i and b_i are pairwise distinct; in other words, every integer from 1 through $2n$ occurs exactly once.

Output

For each test case, output one line containing a floating-point number representing the answer.

Your answer is considered correct if its absolute or relative error does not exceed 10^{-6} . Formally, let your answer be a and the correct answer be b . Your answer is accepted if and only if $\frac{|a-b|}{\max(1, |b|)} \leq 10^{-6}$.



Example

standard input	standard output
3	0.0000000000000000
1 1	0.8333333333333333
1	0.7474747474747474
2	
2 2	
1 4	
2 3	
3 10	
3 6 1	
5 2 4	

Note

In the first test case, *Ras* never goes all-in.

In the second test case, *Ras* goes all-in with probability $\frac{2}{3}$ when she draws card 1, and always goes all-in when she draws card 4. One possible strategy for *Salt* in this case is to always call. Therefore:

- With probability $\frac{1}{6}$, *Ras* draws card 1 and checks, earning a profit of 0.
- With probability $\frac{1}{3}$, *Ras* draws card 1 and goes all-in, earning a profit of -2 .
- With probability $\frac{1}{2}$, *Ras* draws card 4 and goes all-in, earning a profit of 3.

Her expected profit is $\frac{1}{6} \cdot 0 + \frac{1}{3} \cdot (-2) + \frac{1}{2} \cdot 3 = \frac{5}{6}$.



Problem G. Gungnir Fracture

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Geopelia is collecting the fragments of *Gungnir*. *Gungnir* has 2^n types of fragments, numbered $0, 1, \dots, 2^n - 1$. To restore it, for every $0 \leq i < 2^n$, *Geopelia* needs exactly a_i fragments marked i .

To obtain these fragments, *Geopelia* may perform the following operation any number of times:

- Choose two integers m and k satisfying $0 \leq m < 2^n$ and $0 \leq k < 2^n$.
- Obtain one fragment marked with each of $0 \oplus k, 1 \oplus k, \dots, m \oplus k$.

Recall that $\oplus$ denotes the bitwise exclusive OR of two non-negative integers.

Help *Geopelia* find the minimum number of operations required to restore *Gungnir*. Note that *Geopelia* **cannot** remove any fragment she has obtained.

Input

The first line of the input contains an integer T ($1 \leq T \leq 10^4$) indicating the number of test cases. For each test case:

- The first line contains an integer n ($0 \leq n \leq 19$).
- The second line contains 2^n integers $a_0, a_1, \dots, a_{2^n-1}$ ($0 \leq a_i \leq 10^6$), where a_i is the required number of fragments marked i .

It is guaranteed that the sum of 2^n over all test cases does not exceed 2^{19} .

Output

For each test case, output one integer representing the minimum number of operations.

Example

standard input	standard output
5	100000
0	514
100000	1
1	21
114 514	3
2	
1 1 1 1	
3	
0 13 0 0 0 7 2 1	
4	
1 2 2 1 3 3 3 3 1 0 0 0 0 0 0 0	



Problem H. Hydra

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Neko is fighting a long battle against the *Ocean Gangster*. The battle consists of a sequence of rounds, all numbered by positive integers.

Neko has two defensive abilities: **Metallization** and **Plating**. Both abilities initially have level 0.

At the beginning of each round, *Neko* may perform any number of upgrades:

- Pay x to increase the Metallization level by 1.
- Pay y to increase the Plating level by 1.

After all upgrades have been performed, the round is resolved in the following order:

1. Let the current Metallization and Plating levels be M and P , respectively. *Neko* gains $M + P$ points of armor.
2. If the *Ocean Gangster* deals b points of damage in this round, *Neko* loses $\max(0, b - M - P)$ health points.
3. At the end of the round, all armor gained in this round disappears. The Metallization level remains unchanged, while the Plating level becomes $\max(P - 1, 0)$.

The *Ocean Gangster* will make n attacks in total. The i -th attack occurs in round a_i and deals b_i points of damage. The attack rounds are given in strictly increasing order. The *Ocean Gangster* does not attack in any round that is not listed. Note that even if there is no attack in a round, the Plating level still changes according to the rule above at the end of that round.

As a genius gaming streamer, *Neko* wants to lose **no** health points to any attack. Find the minimum total cost required to achieve this goal.

Input

This problem contains multiple test cases.

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

For each test case:

- The first line contains three integers n, x, y ($1 \leq n \leq 5 \times 10^5$, $1 \leq x, y \leq 10^9$), representing the number of attacks, the cost of increasing the Metallization level by 1, and the cost of increasing the Plating level by 1, respectively.
- The next n lines each contain two integers a_i, b_i ($1 \leq a_i, b_i \leq 10^9$), representing an attack of b_i damage made by the *Ocean Gangster* in round a_i .

It is guaranteed that the sum of n over all test cases does not exceed 5×10^5 , and that $a_1 < a_2 < \dots < a_n$.

Output

For each test case, output one line containing an integer representing the minimum cost required to lose no health points to any attack.



Example

standard input	standard output
2	18
3 5 2	7
1 3	
10 5	
20 2	
3 100 1	
1 5	
2 5	
3 5	

Note

In the first test case, *Neko* can first increase the Metallization level to 2 in round 1:

- In round 1, increase the Plating level to 1, obtaining $2 + 1 = 3$ points of armor in total.
- In round 10, increase the Plating level to 3, obtaining $2 + 3 = 5$ points of armor in total.
- In round 20, the 2 points of armor provided by Metallization are sufficient.

The total cost is $2 \times 5 + (1 + 3) \times 2 = 18$.

In the second test case, *Neko* does not need to upgrade Metallization. In round 1, increase the Plating level to 5. At the end of this round, its level decreases to 4. In each of rounds 2 and 3, increase the Plating level by 1. This provides 5 points of armor in all three rounds, for a total cost of $5 + 1 + 1 = 7$.



Problem I. Incyde

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Vanessa has n memories about humanity, numbered $0, 1, \dots, n-1$, arranged in this order on a circle. *Vanessa* divides them into m contiguous intervals, computes the sum S_i of the elements in each interval, and defines the weight of the division as the greatest common divisor of all the S_i .

Through the Eye of *Horus*, *Vanessa* sees all $\binom{n}{m}$ possible divisions, but she cannot endure so many memories. You need to help her calculate the **product** of the weights of all divisions. Since the answer may be large, output it modulo $10^9 + 123$.

Recall that a positive integer is a common divisor of a set S if and only if it divides every element of S .

Input

This problem contains multiple test cases.

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

For each test case, the only line contains two integers n, m ($2 \leq m \leq n \leq 10^{12}$).

Output

For each test case, output one line containing an integer representing the answer.

Example

standard input	standard output
8	108
4 2	400
5 2	895412257
10 3	711051382
2026 1018	1
1000000123 114514	52312939
2000000247 1000000123	472236908
67325072148 1145141919	0
999000122877 1997	

Note

For the first test case, the weights of all six divisions are listed below:

- $\gcd(0, 1 + 2 + 3) = 6$
- $\gcd(0 + 1, 2 + 3) = 1$
- $\gcd(0 + 1 + 2, 3) = 3$
- $\gcd(1, 2 + 3 + 0) = 1$
- $\gcd(1 + 2, 3 + 0) = 3$
- $\gcd(2, 3 + 0 + 1) = 2$

Therefore, the answer is $6 \cdot 1 \cdot 3 \cdot 1 \cdot 3 \cdot 2 = 108$.



Problem J. Journey

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Ritmo and *Fisica* are building a tuning device. A tuning device consists of a sequence a in which every integer in $1 \sim n$ occurs exactly twice, and a **regular bracket sequence** s containing n pairs of brackets. A tuning device works correctly if and only if it satisfies the following conditions:

- Create one vertex for each index.
- For every pair of matching brackets whose indices in s are (l_i, r_i) , add an edge between these two vertices.
- For every value in a , let (x_i, y_i) be the indices of its two occurrences, and add an edge between these two vertices.
- The procedure above produces one cycle containing all $2n$ vertices and all $2n$ edges.

Ritmo has already installed a . Now you need to help *Fisica* find a corresponding sequence s . It can be proven that such a regular bracket sequence always exists.

Recall that a bracket sequence is **regular** if and only if it can be constructed using the following rules:

- $()$ is a regular bracket sequence.
- If A is a regular bracket sequence, then $(+ A +)$ is a regular bracket sequence.
- If A, B are regular bracket sequences, then $A + B$ is a regular bracket sequence.

Input

This problem contains multiple test cases.

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

For each test case:

- The first line contains an integer n ($2 \leq n \leq 2 \times 10^5$).
- The second line contains $2n$ integers a_i ($1 \leq a_i \leq n$).

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5 , and that every integer in $1 \sim n$ occurs exactly twice in a .

Output

For each test case, output a regular bracket sequence s of length $2n$ representing a valid answer.



Example

standard input	standard output
5	()()
2	(())
1 2 1 2	()()()
2	()()()()
2 2 1 1	()()()()()()()
3	
1 2 3 1 2 3	
4	
1 2 3 4 3 2 1 4	
7	
3 6 7 1 5 6 2 2 4 3 4 7 1 5	



Problem K. Kikai No Tsubasa

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

This is a communication problem.

Teto is studying the following NP-complete problem:

- You are given an undirected simple graph G with n vertices and m edges. The vertices are numbered from 1 to n .
- A permutation $p = [p_1, p_2, \dots, p_n]$ of $1, 2, \dots, n$ is a **valid labeling** of the graph if and only if $p_u + p_v \leq n + 1$ for every edge (u, v) .
- It is guaranteed that at least one valid labeling exists. You need to find any valid labeling of the given graph.

In the year 6202, *Teto* not only invented an efficient algorithm for this problem with the help of a quantum computer, but also invented a time machine! Now, *Teto* wants to bring a solution to this problem back in time and accelerate the research into the algorithm. However, she finds that she cannot remember too much information: besides remembering the graph exactly, she can remember only 5×10^5 additional bits. Can you help her compress the information in the solution?

More specifically, you need to design a program that, given G and p , can generate a **non-empty** binary string s of length at most 5×10^5 , and can then independently find a valid labeling p' using only G and s . Note that p' only needs to be a valid labeling of the graph and does not have to be identical to p .

To test your program, the judging system will run the same program twice on each test, corresponding to the two processes described above. The two runs do not share memory, files, or any other state. Apart from the same graph and the binary string output on the first run, the second run receives no information from the first run.

Input

The first line of each run contains a string op ($op \in \{\text{first}, \text{second}\}$), denoting which run of the program it is.

The second line contains two integers n and m ($2 \leq n \leq 10^5$, $0 \leq m \leq 5 \times 10^5$), denoting the number of vertices and the number of edges, respectively. It is guaranteed that n is even.

The next m lines follow. The i -th of these lines contains two integers u_i, v_i ($1 \leq u_i < v_i \leq n$), denoting an edge connecting vertices u_i and v_i .

If $op = \text{first}$, the last line contains n integers $p_1, p_2, \dots, p_n$, representing a valid labeling given to the program.

If $op = \text{second}$, the last line contains a **non-empty** binary string s , representing the information output on the first run. It is guaranteed that **the graph and the order of its edges are identical in both runs**.

It is guaranteed that all edges in the graph are distinct. If $op = \text{first}$, it is also guaranteed that p is a permutation of $1, 2, \dots, n$ and that $p_u + p_v \leq n + 1$ for every edge (u, v) .

Interaction Protocol

If $op = \text{first}$, output a **non-empty** string s consisting only of the characters 0 and 1, where $|s| \leq 5 \times 10^5$. Do not output anything else.

If $op = \text{second}$, output a permutation $p'_1, p'_2, \dots, p'_n$ of $1, 2, \dots, n$ such that $p'_u + p'_v \leq n + 1$ for every edge (u, v) .



Examples

standard input	standard output
first 8 7 1 5 2 6 3 7 4 8 1 2 2 3 3 4 1 2 3 4 8 7 6 5	1001101010010100010011010
second 8 7 1 5 2 6 3 7 4 8 1 2 2 3 3 4 1001101010010100010011010	4 1 3 2 5 8 6 7



Problem L. Lucid Traveler

Input file: standard input
Output file: standard output
Time limit: 4 seconds
Memory limit: 1024 megabytes

This is an interactive problem.

Having awakened, *Ivy* is exploring the road network of Node 08.

Node 08 can be modeled as an n -vertex **edge cactus**, and *Ivy* needs to find every edge in the graph. Fortunately, she may ask the Eye of *Horus* some queries. In each query, *Ivy* chooses two vertices x and y and a set of vertices S containing neither x nor y . The Eye of *Horus* tells her whether x and y are connected after all vertices in S , together with their incident edges, are deleted from the graph. However, the Eye of *Horus* has limited computational power, so such a query incurs a cost of $\max(|S| - 1, 0)$.

Time is limited. *Ivy* may make at most $3n^2$ queries, and the sum of the costs of all queries must not exceed $n\lceil\log_2 n\rceil + 2n$. You need to help *Ivy* devise a strategy that is guaranteed to determine every edge in the graph.

Recall that an edge cactus is an undirected simple connected graph in which every edge belongs to at most one simple cycle. A simple graph is a graph with neither multiple edges nor self-loops.

Note that the interactor is **adaptive**, meaning that the graph for each test case is not pre-determined. The interactor always guarantees that there is at least one graph consistent with all previous replies, but it may choose its reply based on your query. The interactor determines the final graph only when you submit your final answer.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \leq T \leq 128$) indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \leq n \leq 128$) indicating the number of vertices in the graph.

It is guaranteed that the sum of n over all test cases does not exceed 256.

The query-count and query-cost limits are evaluated **separately** for each test case.

Interaction Protocol

To ask a query, output one line. First output `?` followed by a space, then print three integers x , y , and k , followed by k integers $s_1, s_2, \dots, s_k$, all separated by spaces. Here, $k = |S|$, $1 \leq x, y \leq n$, $x \neq y$, $0 \leq k \leq n - 2$, and $s_1, s_2, \dots, s_k$ must be pairwise distinct and different from both x and y .

After flushing your output, your program should read a single integer t indicating the answer to your query. If x and y are still connected after deleting S , then $t = 1$; otherwise, $t = 0$. Recall that you can ask at most $3n^2$ queries for a test case, and the sum of the costs of all queries must not exceed $n\lceil\log_2 n\rceil + 2n$.

If you want to report all edges in the graph, first output one line. First output `!` followed by a space, then print an integer m indicating the number of edges. Then output m lines, where the i -th line contains two integers u_i and v_i indicating an edge. The edges may be listed in any order, and the endpoints of each edge may be listed in either order, but the same edge must not be printed more than once.

After flushing your output, your program should continue processing the next test case, or exit immediately if there are no more test cases. Note that your final answer does not count as a query and incurs no query cost.

To flush your output, you can use:

- `fflush(stdout)` (if you use `printf`) or `cout.flush()` (if you use `cout`) in C and C++.
- `System.out.flush()` in Java and Kotlin.



- `sys.stdout.flush()` in Python.

Example

standard input	standard output
2	
3	
	? 1 2 1 3
1	
	! 3
	1 2
	1 3
	2 3
6	
	? 2 6 2 4 5
0	
	? 1 4 0
1	
	? 1 4 2 2 5
0	
	! 7
	1 2
	4 5
	1 5
	2 4
	4 3
	4 6
	3 6



Problem M. Melodiniq

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 1024 megabytes

Uni and her teammates are unlocking a high-difficulty song through Linked VERSE. During the unlocking process, four players simultaneously play a song containing n notes.

Each player starts with a Gauge of 2000. For every note, each player receives one of five judgements:

- JUSTICE HEAVEN (H): This judgement does not decrease the player's Gauge.
- JUSTICE CRITICAL (C): This judgement does not decrease the player's Gauge.
- JUSTICE (J): This judgement decreases the player's Gauge by 50.
- ATTACK (A): This judgement decreases the player's Gauge by 100.
- MISS (M): This judgement decreases the player's Gauge by 200.

Meanwhile, each player has an independent combo counter. When a player receives a JUSTICE HEAVEN judgement, their own combo count increases by 1; receiving any other judgement resets only their own combo count. When a player's combo count reaches 100, they increase the Gauge of each of the other three players by 30 and reset their own combo count.

For each note, the corresponding judgements of all players are resolved **simultaneously**; that is, each player's Gauge changes directly to the value obtained after all changes are applied at once. After the resolution, if a player's Gauge is ≥ 2000 , it is set to 2000.

However, *Uni* notices that after the judgements for all notes have been resolved, if a player's Gauge was ≤ 0 at **any** point, their Gauge is forcibly set to 0 before the game ends. Can you help her find each player's Gauge at the end of the game?

Input

This problem contains multiple test cases.

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

For each test case:

- The first line contains an integer n ($1 \leq n \leq 10^5$).
- Each of the next four lines contains a string s of length n , where the i -th character represents that player's judgement on the i -th note.

It is guaranteed that the sum of n over all test cases does not exceed 10^5 , and that every string contains only the five characters H, C, J, A, and M.

Output

For each test case, output four integers representing the final Gauge values of the four players, in input order.

Example

standard input	standard output
3	2000 1750 1500 1000
5	1300 2000 900 1050
CCCCC	0 2000 830 1930
JJJJJ	
AAAAA	
MMMMM	
10	
HCJAMHCJAM	
CHCHCHCHCH	
JAMJAMJAMJ	
MAJHAMMACH	
103	
MMMMMMMMMMCCCCCCCCC	
CCCCCCCCCCCCCCCCCCCC	
CCCCCCCCCCCCCCCCCCCC	
CCCCCCCCCCCCCCCCCCCC	
CCCCCCCCCCCCCCCCCCCC	
CCC	
HHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHH	
HHH	
CJJCJCJCJHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHH	
HHHHHJJJCJCJCJCJCJCJ	
CJCJCJCJCJCJCJCJCJCJCJ	
CJC	
HHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHAAHHHH	
HHHHHHHHHHHHHHHHHHHHHH	
HHHHHHHHHHHHHHHHHHHHHH	
HHH	

Note

To ensure that the PDF is displayed correctly, several blank lines have been inserted into the last sample. You can download the original sample from Domjudge.



Thank you for solving

Illustrator: *Nagisa11*