Problem A. NJU Emulator

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

"Dr.JYY, s86 instruction set, and NEMU, what should that remind you of?"

"Oh, no..."

NJU Emulator(a.k.a, NEMU) is the latest s86 architecture simulator developed by Dr. JYY. s86 is stack-style computer architecture, with its machine instructions only operating on the top element of the stack. The computation model of s86 includes a stack and a program of finite length. Each element in the stack is in 64-bit unsigned integer type; We denote the stack as S and size of the stack as $size(S)$, then the program consists of the instructions in the following table and must end with a terminating instruction. When the s86 machine is running, the stack will first be initialized to empty, and then each instruction in the program will be executed in sequence until the last one is executed. After an *end* instruction, the machine will output the top element of the stack and halt.

Notation	Functionality	Restrictions
p1	push the constant 1 into S	No Restrictions
dup	add a copy the topmost element(of S) into S	$size(S) \geq 1$
pop	pop the topmost element of S	$size(S) \geq 1$
swap	swap the two topmost elements of S	$size(S) \geq 2$
add x	Add the xth element to the topmost element of S	$x \geq 0$ and $size(S) > x$
sub x	Subtract the xth element from the topmost element of S	$x \geq 0$ and $size(S) > x$
mul x	Multiply the xth element to the topmost element of S	$x \geq 0$ and $size(S) > x$
end	Output the topmost element of S and the program halts	$size(S) \geq 1$

In the table, the xth element refers to the xth element from the top to the bottom of the stack, where the topmost element is the 0th element.

Notably, all arithmetic operations(add, sub, mul) in the s86 instructions should be done modular 2^{64}, i.e., when the arithmetic operation has a result of X, the result in the s86 instruction set should be $X'(0 \leq X' < 2^{64})$, and $X - X'$ is a multiple of 2^{64}, it can be shown that such X' exist for any integer X.

Now that Dr. JYY has finished the development of NEMU. To test the correctness of NEMU, Dr. JYY wants you to write a program using s86 instructions, so that when the program halts, it should output the given integer N.

Input

The first line contains a number $T(1 \leq T \leq 10000)$, denoting the number of test cases.

Each test contains an integer $N(0 \leq N < 2^{64})$ in one line, denoting the number your program should output when halts.

Output

For each test case, you should output a piece of program consisting of s86 instructions in the form as in the table in the statement so that when your program halts, it should output the given integer N. **Also, your program should contain at most** 50 **s86 instructions.** It is guaranteed that there exists at least one such program under the limits of this problem. If any restriction in the instruction table is violated during the execution of your program, your answer would be considered incorrect.

If many programs satisfy the restriction, you should output any such program, which will be considered correct.

Example

standard input	standard output
3 0 2 18446744073709551615	p1 sub 0 end p1 add 0 end p1 dup add 1 swap sub 1 end

Note

The process of the execution of the last test case in the sample output is shown in the following table.

Instruction	Stack
p1	[1)
dup	[1, 1)
add 1	[1, 2)
swap	[2, 1)
sub 1	[2, 18446744073709551615)
end	The program outputs 18446744073709551615 and halts

Problem B. Just another board game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

"So now I move the piece to $(179, 231)$*. It's the* 999999999*th move of this game. Finally, one move to go!"*

"What? Isn't it only the 99999997*th move of this game?"*

*"Oh, f**k."*

After playing some games of Go, Roundgod and kimoyami decide to try something different. Now they are playing a new kind of game on a chessboard. The chessboard is a grid board with n rows and m columns. We assume that the upper left corner of the chessboard has coordinate $(1, 1)$, and the lower right corner of the chessboard has coordinate (n, m). There's a number on every grid of the board, with the number written on the grid on the ith row and jth column equal to a_{ij}. What's more, there's a chess piece on the upper left corner(i.e., $(1, 1)$) of the chessboard initially. Now the two players take turns to choose one of the following operations, starting from Roundgod:

1. Move the chess piece. If it's Roundgod's turn, he can move the chess piece to any position in the **same row**(It's also OK to move it to the current position, i.e., not moving it at all). If it's kimoyami's turn, he can move the chess piece to any position in the **same column**. (It's also OK to move it to the current position, i.e., not moving it at all)
2. *Screw it. I'm going home.* Finish the game immediately.

The game ends when either of the two players chooses the second operation or when the game has already been going on for k turns. (Either of the two players' operations counts as one turn). The value of the game is defined as the number on the grid where the chess piece lands when the game ends. **Now, Roundgod wants to maximize this value, while kimoyami wants to minimize this value.** They don't have the patience to actually play this game for possibly that many turns, so they want you to calculate what will be the final value of the game if both players choose the optimal strategy?

Input

The first line contains a number $T(1 \le T \le 25)$, denoting the number of test cases.

The first line of each test case contains three integers $n, m, k(n, m \ge 1, 1 \le n \times m \le 10^5, 1 \le k \le 10^{18})$, denoting the size of the chessboard and the maximum number of turns the game will last, respectively.

Then n lines follow, the ith$(1 \le i \le n)$ of the n lines contain n integers $a_{i1}, a_{i2}, \ldots, a_{im}$, where $a_{ij}(0 \le a_{ij} \le 10^9)$ denotes the he number written on the grid on the ith row and jth column.

It is guaranteed that $\sum(n \times m) \le 10^6$ over all test cases.

Output

For each test case, output one integer in a line, denoting the final value of the game if both players choose the optimal strategy.

Example

standard input	standard output
3 2 2 2 1 2 2 1 2 2 1 1 2 2 1 2 3 2 1 3 2 3 2 1	1 2 2

Problem C. Dota2 Pro Circuit

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

TI10 and ICPC World Finals 2020, which will be held earlier? Take a bet!

The International(TI) is the biggest and most prestigious event of Dota2 and is commonly held annually in August. The Dota2 teams should try to earn points to be eligible to compete in TI. There are two ways of earning points, first is by competing in the regional contests, second is by competing in the tournaments, where all teams are gathered to compete together and earn points according to their rank in the tournament. A team's final score is the sum of scores from both the regional contests and tournaments.

Now that the regional contests have finished, there are n teams taking part, and the ith team has earned a_i points from the regional contests. Also, the team that gets the ith rank in the tournament can gain b_i points.

cyz is a huge fan of Dota2. So before the tournament starts, he will predict the final rank of all teams. cyz wants to know, for each team, what's its best possible rank and its worst possible rank after the tournament finishes.

If a team has a final score equal to x, its rank is defined as one plus the number of teams with a strict higher score than it. For example, if the final score of four teams are $700, 500, 500, 300$ respectively, then their final ranks are $1, 2, 2, 4$, respectively.

Input

The first line contains a number $T(1 \le T \le 20)$, denoting the number of test cases.

The first line of each test case contains one number $n(1 \le n \le 5000)$, denoting the number of different teams that participate in the regional contests and tournaments.

The next line contains n integers $a_1, a_2, \ldots, a_n(0 \le a_i \le 10^9)$, denoting the points of each team before the tournament starts.

Then follows one line containing n integers $b_1, b_2, \ldots, b_n(0 \le b_n \le b_{n-1} \le ... \le b_1 \le 10^9)$, where b_i denotes the number of points a team would get if ranking ith in the tournament.

It is guaranteed that there are at most 8 cases with $n > 100$.

Output

For each test case, output n lines, where the ith line contains two integers $best_i, worst_i(1 \le best_i \le worst_i \le n)$, denoting the best possible and worst possible rank a team would get after the tournament finishes.

Example

standard input	standard output
2 3 5 10 8 5 2 1 2 5 6 4 4	2 3 1 2 1 3 2 2 1 1

Problem D. Into the Woods

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Not all who wander are lost.

Perhaps it's just taking a walk, or perhaps it's for escaping from something. Roundgod temporarily left the city he used to live in and headed to a boundless forest.

Roundgod then began to wander in the forest. If we consider the forest as a two-dimensional Cartesian plane, then Roundgod starts from $(0, 0)$, and each time he chooses one of the four directions uniformly at random and moves one step forward(i.e.,the possible changes of Roundgod's coordinates are $(-1, 0), (1, 0), (0, -1), (0, 1)$). After Roundgod took n steps, he stopped, in a trance, realizing that he had been wandering for too long and that it was time to return to his casual life. Now there's only one remaining problem that he wonders: During the n steps he moved, how far he was from the starting point? As Roundgod has forgotten the moves he has taken, he only wants to know the expected answer for all his possible routes.

Formally, Roundgod starts from $(0, 0)$, each time chooses one of the four directions uniformly at random and moves one step forward. You should calculate, among all his 4^n possible routes, the expected maximum **Manhattan distance** between any position on his route and $(0, 0)$.

Specifically, the Manhattan distance between two points $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ in the Cartesian plane is defined as $d(p_1, p_2) = |x_1 - x_2| + |y_1 - y_2|$. Also, for any prime modular $10^8 \le MOD \le 10^9 + 7$, we can prove that the answer can be written as a fraction $\frac{P}{Q}$, where P and Q are coprime integers and $Q \not\equiv 0 \pmod{MOD}$. You need to output $P \cdot Q^{-1} \pmod{MOD}$ as the answer, where $Q^{-1} \pmod{MOD}$ represents the modular inverse of Q with respect to MOD.

Input

The first line contains a number $T(1 \le T \le 10)$, denoting the number of test cases.

The first line of each test case contains two numbers $n, MOD(1 \le n \le 10^6, 10^8 \le MOD \le 10^9 + 7)$, denoting the number of steps taken and the modular. It is guaranteed that MOD is prime.

Output

For each test case, output one integer in a line, denoting the answer.

Example

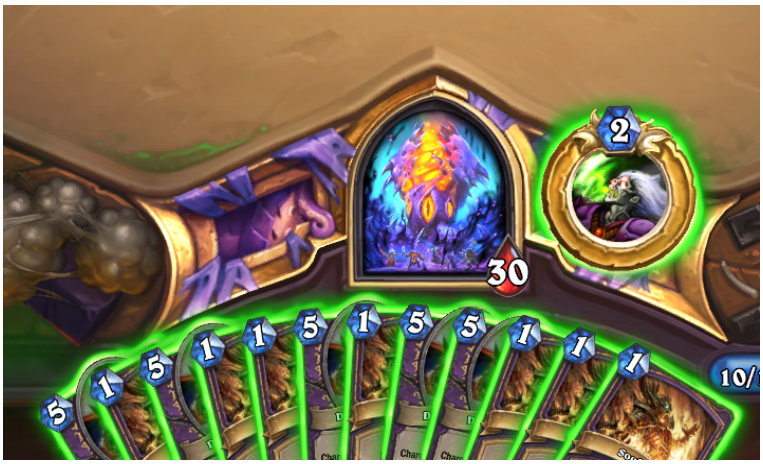
standard input	standard output
3 1 1000000007 2 998244353 50 1000000007	1 249561090 603242629

Problem E. Did I miss the lethal?

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	256 megabytes

How much damage can you deal with two "Soulfire"s in hand? 8? Are you sure?

-skyline- is playing Hearthstone, and his favorite class is Warlock. There's a special mechanism designed for cards of the Warlock class that after playing some cards from hand, one may need to discard some cards from the remaining hand randomly.



-skyline- has a hand of n cards, with each card can deal some damage, but also possibly would discard some other cards from hand after playing. He is planning for a lethal turn. As a professional player of Hearthstone, he doesn't believe in his luck, so he wants to calculate that in the **worst case**, what is the maximum damage he can deal using this hand(We ignore the mana costs of the cards and assume that every card can be played, if not discarded).

Formally, there are n cards in -skyline-'s hand, and the ith card has two properties d_i and a_i, denoting the damage it can deal and the number of cards one needs to discard after playing this card, respectively.

-skyline- can choose the order to play cards from his hand, one by one. After he chooses to play the ith card from his hand(after playing, this card no longer exists in his hand), he would deal a_i damage to the enemy hero, then d_i cards from his remaining hand are uniformly chosen at random, then discarded(If currently -skyline- has a hand of less than d_i cards, then all cards in his hand would be discarded). After a card is discarded from hand, it can not be played anymore.

-skyline- wants to choose the best strategy, such that **in the worst case**, he would deal the maximum number of total damage to the enemy hero. Please help -skyline- find out what this total damage is.

Input

The first line contains a number $T(1 \le T \le 20)$, denoting the number of test cases.

The first line of each test case contains one number $n(1 \le n \le 200)$, denoting the number of cards in -skyline-'s hand.

Then n lines follow. The ith line contains two integers $d_i(1 \le d_i \le 10^7)$ and $a_i(1 \le a_i \le 4)$, denoting the damage the ith card can deal and the number of cards one needs to discard after playing the ith card, respectively.

It is guaranteed that $\sum n \le 2000$ over all test cases.

Output

For each test case, output an integer in one line, denoting the maximum number of total damage -skyline-

can deal to the enemy hero in the worst case.

Example

standard input	standard output
2 3 3 1 4 2 5 2 6 10 1 10 1 15 2 20 3 20 3 25 4	7 40

Note

For the first test case of the sample, the optimal strategy for -skyline- is to play the first card, then play the only remaining card that is not discarded. In this case, -skyline- can deal a total of $3 + 4 = 7$ damage to the enemy hero in the worst case.

Problem F. Guess The Weight

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

"Guess The Weight', It's turn 2. Well let's play this and see what we can draw'"

***Draws "Innervate"**.*

"Wow that's lucky for me! All I need to do is to click "Greater' then getting a free second draw!"

Sees another "Innervate"

"Are you sure you want to uninstall Hearthstone from your computer?Yes."

uuzlovetree loves playing Hearthstone, and his favorite class is Druid. In Hearthstone, there's a spell for the Druid class called 'Guess the weight,', as shown below.

uuzlovetree knows the number of cards in the deck and knows the mana cost of each card. He wants to know the probability of getting the second card when he plays the 'Guess the weight' under the optimal guessing strategy.

Formally, assume there are currently m cards in the deck, with a number representing mana cost on each card. Now one randomly shuffles all m cards in the deck(each of the $m!$ possible arrangements of the cards appear with equal probability). When one plays the card 'Guess the weight,' he draws the first card of the deck and chooses one of the following two options:

1. Predict that the next(second) card of the deck has a **smaller** mana cost than the first card, then reveal the next card of the deck. If your prediction is correct, you draw the second card.
2. Predict that the next(second) card of the deck has a **greater** mana cost than the first card, then reveal the next card of the deck. If your prediction is correct, you draw the second card.

Caution: If the second card of the deck has equal mana cost with the first card, then no matter which option you choose, you cannot draw the second card of the deck.

Initially, there are $n \geq 2$ cards in the deck, with mana costs $a_1, a_2, \ldots, a_n$, respectively. Now q events happen to the deck(How can those events happen? Try Hearthstone to find out for yourself!), with each event in one of the following two forms:

1. Add x cards with mana cost w into the deck.
2. Find out when one applies an optimal strategy, what is the maximum probability that he can draw the second card when he plays 'Guess the weight.'

Input

The first line contains a number $T(1 \le T \le 10)$, denoting the number of test cases.

The first line of each test case contains two numbers $n(2 \le n \le 2 \cdot 10^5)$ and $q(1 \le q \le 2 \cdot 10^5)$, denoting the initial size of the deck, and the number of events, respectively.

Then one line follows, containing n integers $a_1, a_2, \ldots, a_n(1 \le a_i \le 2 \cdot 10^5)$, denoting the mana costs of the n initial cards.

Then q lines follow, with each line in one of the two following forms:

1. 1 x w, denoting that x cards with mana cost w is added into the deck. $(1 \le x \le 100, 1 \le w \le 2 \cdot 10^5)$.
2. 2, denoting a query that asks, when playing 'Guess the weight' with the current deck, what is the maximum probability that one can draw the second card.

It is guaranteed that $\sum n \le 10^6$ and $\sum q \le 10^6$ over all test cases.

Output

For each event of the second type of each test case, output a fraction in the form of p/q in one line, denoting the maximum probability that one can draw the second card. It is required that p and q are both integers and $gcd(p, q) = 1$, where $gcd(p, q)$ denotes the greatest common divisor of p and q

Example

standard input	standard output
2 2 5 1 1 2 1 1 2 2 1 1 2 2 2 5 1 2 2 1 1 3 2 1 100 4 2	0/1 2/3 2/3 1/1 5/6 201/3502

Note

For the first test case of the sample, initially, the deck consists of two cards with mana cost 1. So no matter which choice one picks, he cannot draw the second card.

After adding a card with mana cost 2 into the deck, the optimal strategy one can apply is as follows: If the mana cost of the first card drawn is 1, then he predicts that the next card has a greater cost, otherwise he predicts that the next card has a smaller mana cost. One can certify that under this strategy the probability of drawing the second card is $\frac{2}{3}$.

After adding another card with mana cost 2 into the deck, the optimal strategy doesn't change. One can certify that under this strategy the probability of drawing the second card is still $\frac{2}{3}$.

Problem G. Boring data structure problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

"Why don't you write some background story for this boring data structure problem?"

"Because it's too boring..."

There's a queue(not the original queue one may refer to as a data structure, here a double-ended queue, to be more precise) that is initially empty. Then, there are infinite elements numbered $1, 2, \ldots$, entering the queue in the order of their numbers(i.e., the first element to enter the queue is numbered 1, the second, 2 et cetera). If an element leaves the queue, it will not enter the queue anymore.

Now there are q operations, each in one of the following forms:

1. Let the next element enter the left end of the queue.
2. Let the next element enter the right end of the queue.
3. Let the element numbered x leave the queue.
4. Ask the number of the element in the middle of the queue. (If there are m elements in the queue currently, you should output the number of the element that is the $\lceil \frac{m+1}{2} \rceil$th from the left).

Input

The first line of each test case contains one number $q(1 \leq q \leq 10^7)$, denoting the number of operations.

Then q lines follow, each in one of the following forms:

1. L, denoting the next element enters the left end of the queue
2. R, denoting the next element enters the right end of the queue
3. $G\ x$, denoting the element numbered x leaves the queue (It is guaranteed that the element numbered x is in the queue when this operation is applied)
4. Q denoting a query that asks the number of the element in the middle of the queue(It is guaranteed that the queue is not empty when this operation is applied)

It is guaranteed that the number of operations of the third and the fourth type is both less than $1.5 \cdot 10^6$.

Output

For each operation of the fourth kind, output a number in a line, denoting the answer to the query.

Example

standard input	standard output
9 L L L Q R Q G 1 R Q	2 1 4

Problem H. Integers Have Friends 2.0

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Acknowledgment: Special thanks to Codeforces Problem 1548E Integer Have Friends for providing the general statement for this problem.

Indian mathematician Srinivasa Ramanujan once quoted the famous words of Indian mathematician Srinivasa Ramanujan(?) that "every positive integer was one of his personal friends."

It turns out that positive integers can also be friends with each other! You are given an array a of distinct positive integers.

Define a **subsequence** $a_{c_1}, a_{c_2}, \ldots, a_{c_k}$ where $k \geq 1$ and $1 \leq c_1 < c_2 < \cdots < c_k \leq n$ to be a friend group if and only if there exists an integer $m \geq 2$ such that $a_{c_1} \mod m = a_{c_2} \mod m = \cdots = a_{c_k} \mod m$, where $x \mod y$ denotes the remainder when x is divided by y.

Your friend gispzjz wants to know the size of the largest friend group in a. Can you help him?

Input

The first line contains a number $T(1 \leq T \leq 30)$, denoting the number of test cases.

The first line of each test case contains one integer $n(2 \leq n \leq 2 \times 10^5)$, denoting the size of the array a.

Then one line containing n integers $a_1, a_2, \ldots, a_n(1 \leq a_i \leq 4 \times 10^{12})$ follow, representing the contents of the array a. **It is guaranteed that all the numbers in a are distinct.**

It is guaranteed that $\sum n \leq 10^6$ over all test cases.

Output

For each test case, output a line consisting of a single integer, denoting the size of the largest friend group in a.

Example

standard input	standard output
3 3 10 12 15 4 4 6 9 19 6 2 8 11 15 19 38	2 3 4

Problem I. Little Prince and the garden of roses

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

He was standing before a garden, all a-bloom with roses.

"Good morning,"said the roses.

The little prince gazed at them. They all looked like his flower.

"Who are you?"he demanded, thunderstruck.

"We are roses,"the roses said.

And he was overcome with sadness. His flower had told him that she was the only one of her kind in all the universe. And here were five thousand of them, all alike, in one single garden!

"She would be very much annoyed,"he said to himself, "if she should see that... she would cough most dreadfully, and she would pretend that she was dying, to avoid being laughed at. And I should be obliged to pretend that I was nursing her back to life– for if I did not do that, to humble myself also, she would really allow herself to die..."

Then he went on with his reflections: "I thought that I was rich, with a flower that was unique in all the world; and all I had was a common rose. A common rose, and three volcanoes that come up to my knees– and one of them perhaps extinct forever... that doesn't make me a very great prince..."

And he lay down in the grass and cried.

Assume the garden of roses has n rows of roses, where there are n roses in each row, aligned in a square. Initially, each rose has a color(color of the petals, of course!). The rose in the i-th row and j-th column is colored with c_{ij}.(There are surely colorful roses! Why would people only think of red ones when it comes to roses?)

When little prince wanders in the garden of roses, he will burst into tears when he sees two roses with the same color in the **same row** or the **same column**, as that would remind him his flower is not unique in this world.

You, a kind person, definitely don't want to see the little prince cry. Fortunately, you arrived at the garden of roses just before the little prince. You have a magic painter that could paint the **stems** of some roses into different colors(You don't want to ruin the petals of the roses, do you?). Initially, the stems of all roses are green. You want to paint the stems of some roses into different colors, such that **no two roses in the same row or column would no longer have both the same color of petals and stems**. The only problem is the number of pigments with distinct colors you need to prepare. As preparing pigments with distinct colors is time-consuming, you want to use **as little pigments as possible**. Please find a way to paint the flowers such that the condition is satisfied. As long as the number of pigments is minimized, any valid way of painting flowers would be considered correct.

Input

The first line contains a number $T(1 \le T \le 100)$, denoting the number of test cases. The first line of each test case contains an integer $n(1 \le n \le 300)$, denoting the size of the garden. Then n lines follow, the ith$(1 \le i \le n)$ of the n lines contain n integers $c_{i1}, c_{i2}, \ldots, c_{in}$, where $c_{ij}(1 \le i, j \le n, 1 \le c_{ij} \le n^2)$ denotes the color of the rose in the ith row and jth column.

It is guaranteed that at most 10 cases $n > 20$ and at most 4 cases $n > 100$.

Output

For each test case, output two integers $d, m (0 \le d \le n^2, 1 \le m \le n^2)$ in the first line, denoting the number of distinct pigments used and the number of roses to be painted. For the next m lines, output three integers $i, j, c (1 \le i, j \le n, 1 \le c \le d)$ in a line, denoting that you would paint the stem of the rose in the ith row and jth column into color c. **If the stem of some rose is not painted, it is regarded to have color 0.** As long as the number of pigments is minimized, any valid way of painting the stems of the roses would be considered correct.

Example

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

Problem J. Unfair contest

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

I'm going to give my scores fairly. It's just that some contestant deserves a fairer score...

gispzjz and zyb are participating in a contest, with n referees awarding scores(according to their performance, usually) to them. For each contestant, each referee should name an integer in the interval $[1, h]$ as the score, and the final score of the contestant is the sum over all scores he gets after eliminating s highest scores and t lowest scores.

As one of the referees, you had a bet on gispzjz, so you want him to win this contest, but you also don't want this to look too obvious. Suppose you know the other $n-1$ referees have awarded scores $a_1, \ldots, a_{n-1}$ to gispzjz and $b_1, \ldots, b_{n-1}$ to zyb. You need to give out your scores a_n and b_n so that the final score of gispzjz is **strictly higher** than zyb. If that's achievable, you also need to minimize $a_n - b_n$, conditioned on the final score of gispzjz is strictly higher than zyb.

Input

The first line contains a number $T(1 \le T \le 12000)$, denoting the number of test cases.

The first line of each test case contains four integers $n, s, t, h(1 \le n \le 10^5, 0 \le s, t \le n-1, 1 \le h \le 10^9)$, denoting the number of referees, the number of highest and lowest scores that need to be eliminated, and the scoring range for referees, respectively. It is guaranteed that $s + t \le n - 1$.

Then one line containing $n-1$ integers $a_1, ..., a_{n-1}(1 \le a_i \le h)$ follow, denoting the scores already awarded to gispzjz.

Then another line containing $n-1$ integers $b_1, ..., b_{n-1}(1 \le b_i \le h)$ follow, denoting the scores already awarded to zyb.

It is guaranteed that $\sum n \le 10^6$ over all test cases.

Output

For each test case, if it's possible to make gispzjz's score strictly higher than zyb, then output the minimized $a_n - b_n$ in one line, otherwise output "IMPOSSIBLE"(without quotes) in one line.

Example

standard input	standard output
3 3 1 1 4 1 3 2 4 4 1 1 9 4 4 5 4 5 5 4 1 1 9 4 5 5 4 4 5	1 IMPOSSIBLE -4

Problem K. ZYB's kingdom

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	512 megabytes

"Dad, our city and the neighboring city haven't got any income for years. What's happening?"

"Well, son, there's an ongoing pandemic in the neighboring city, you know..."

"Fine. And what about our city?"

"I really wish to knock his head off when the king decided to build this country like a tree and made our city a leaf!"

There are n cities in ZYB's kingdom, and it's connected by $n-1$ bidirectional roads between the cities. Initially, the ith city has a prosperity index equal to c_i, when a trade happens between two cities u and v, the income of city u increases by c_v, and the income of city v increases by c_u. (Strange definition, isn't it?)

Every once in a while, ZYB will hold a trading festival to boost the economy in his kingdom(and to get more taxes, of course!). However, some cities must be shut down and not allowed to pass during the festival due to the pandemic. Formally, when a trading festival is held, a list of k cities $a_1, a_2, \ldots, a_k$ that are shut down are informed, and for any pair of cities $(u, v)(u < v)$, if the unique path between them doesn't contain any of the k cities, then a trade happens between them, i.e., the income of city u increases by c_v, and the income of city v increases by c_u.

Initially, the income of every city in ZYB's kingdom is zero. Then q events happen, each in one of the following forms:

1. Let's hold a trading festival! A list of k cities $a_1, a_2, \ldots, a_k$ are given, then for any pair of cities $(u, v)(u < v)$, if the unique path between them doesn't contain any of the k cities, then a trade happens between them.
2. Find the current total income of some city v.

It is guaranteed that $\sum n \le 10^6, \sum q \le 10^6$ and $\sum k \le 10^6$ over all test cases.

Input

The first line contains a number $T(1 \le T \le 20)$, denoting the number of test cases.

The first line of each test case contains two integers $n, q(1 \le n, q \le 2 \cdot 10^5)$, denoting the number of cities in zyb's kingdom and the number of events, respectively.

Then $n-1$ lines follow. Each line contains two integers u, v, denoting an edge between city u and city v.

Then one line containing n integers $c_1, c_2, \ldots, c_n(1 \le c_i \le 10^6)$ follows, denoting the prosperity index of each city.

Then q lines follow, each line describing an event in one of the following forms:

1. 1 $k(1 \le k \le n), a_1, a_2, ..., a_k(\forall 1 \le i \le k, 1 \le a_i \le n$ and all a_i are distinct), denoting an event of the first type.
2. 2 $v(1 \le v \le n)$, denoting an event of the second type that asks about the current income of city v.

Output

For each event of type 2, output a number in one line denoting the answer.

Example

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