Inter University Programming Contest | United Group Presents BUET CSE Fest 2024

https://toph.co/c/inter-university-buet-cse-fest-2024

Toph

Schedule

The contest will run for **5h0m0s**.

The standings will be frozen for the last **1h0m0s** of the contest.

Authors

The authors of this contest are Ahmed.032170, AlphaCoder101, Iftekhar_Hakim, longlongint, LUMBERJACK_MAN, neo11235, shariful_islam, Sk_Sabit, TamimEhsan, and tbs_sajal15.

Rules

This contest is formatted as per the official rules of ICPC Regional Programming Contests.

You can use C++17 GCC 13.2, C++20 Clang 16.0, C++20 GCC 13.2, C++23 GCC 13.2, Java 1.8, PyPy 7.3 (3.10), and Python 3.12 in this contest.

Be fair, be honest. Plagiarism will result in disqualification. Judges' decisions will be final.

Notes

There are 12 challenges in this contest.

Please make sure this booklet contains all of the pages.

If you find any discrepencies between the printed copy and the problem statements in Toph Arena, please rely on the later.

A. Anya Restores Order

Anya doesn't like disorderliness, everything she has needs to be in order and complete. Today, she went to a store and bought some powerballs also called lottery balls. The balls are small in size and have a number written on it. Anya asked the storekeeper to give her a complete package of balls with consecutive numbers from 1 to B meaning she wanted B balls each with a unique number and no numbers missing between them. Then, she asked for them to be packed into tubes, each containing the same number of balls, arranged in sequential order. A complete and organized package!

The store keeper packed the balls and delivered them to Anya. When she opened the packages, she was speechless! It's all wrong! Not only the packages are unevenly filled with different numbers of balls, the balls are not in sequential order too! To make things worse there are some balls missing and there were even some that probably belonged to other orders. What a disaster!

Despite the mess, she is determined to restore order. She decides to take two tubes of balls and transfer all of the balls into another one and discard the empty one. As she merges the tubes, she will occasionally ask you to help her find any missing balls that should be numbered lower than the highest-numbered ball in the tube. If there are multiple missing balls you report the smallest one. If there are no missing balls, then report that to her by saying $complete$! Please note that due to the same packaging error, there may be multiple balls of the same number in a tube (that shopkeeper will pay for this).

Help Anya to attain her order.

You are given N array indicating the tubes. Each array A_i has arbitrary size M_i contains some positive integers A_{ij} representing the balls. You will be given Q queries. There will be two type of queries

- $1\ i\ j$ Append array A_j after array A_i and delete the array A_j. **It is guaranteed that there is an array with id i and j at the time of this query.**
- $2\ i$ Output the missing number ball of the array A_i according to the description. It is guaranteed that there is an array with id i at the time of this query.

Input

The first line of input will have a positive integer T, the number of independent test cases.

Each case will begin with two positive integer N, Q, the number of arrays and the number of queries. Then the next N lines each will start with $M_i$indicating the size of the array followed by M_i integers A_{ij}indicating the elements of the array.

Next Q lines indicates queries each of them in the form of mentioned previously

- $1 \leq T \leq 10^5$
- $1 \leq N \leq 10^5$
- $1 \leq M_i \leq 10^5$
- $1 \leq A_{ij} \leq 10^9$
- Sum of M_i over all tests cases $\leq 10^6$
- Sum of Q over all tests cases $\leq 2 \times 10^6$

Each query will start with a integer $1 \leq t \leq 2$

- For $t = 1$ it will follow $i\ j, 1 \leq i, j \leq \mathbf{n}$
- For $t = 2$ it will follow $i, 1 \leq i \leq \mathbf{n}$

Output

For each query of type 2, output the answer in a new line

Samples

Input	Output
1 4 7 4 2 4 1 2 3 5 3 7 2 6 8 5 3 1 5 1 10 2 1 1 1 2 2 1 1 3 1 2 3 1 4 3 2 4	3 6 complete 9

B. Boat-Flix

There are N swimmers who want to cross a river. You are given K boats, each with a speed of Y meters per second, and each boat can be used by only one swimmer at a time. A boat cannot move on its own; it must be driven by a swimmer. The swimmers, on the other hand, can swim at a speed of X meters per second. The width of the river is D meters. A swimmer can swim for part of the way, ride a boat for part of the way, and even leave a boat midway for another swimmer to take it.

All swimmers start from the same bank of the river at the same time. Initially the boats are also present on the same bank. Your task is to determine the minimum amount of time required for the last swimmer to reach the opposite bank.

Input

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

Each test case consists of a two lines. First line contains two space separated integers: N, K, and the second line contains three space seperated integers, D, X, and Y.

- $1 \leq N, K \leq 10^6$
- $1 \leq D, X, Y \leq 10^9$

Output

For each test case, output a single line containing the minimum amount of time required for the last of N swimmers to reach the opposite bank.

Absolute or relative errors of up to 10^{-6} in the output will be ignored.

Samples

Input	Output
2 3 2 100 10 15 5 6 120 10 15	7.7777777778 8.0000000000

C. Dual Wield

For an array X of natural numbers, define $F(X) = Len^{\dagger}(Set^{\ddagger}(X))$. You are given two arrays A and B of size n. For any $1 \leq i \leq n$, you can swap A_i and B_i. Output the maximum value of $F(A) + F(B)$ along with A and B after doing all operations. You can do any non negative integer number of operations.

$\dagger Set(X) = \{x : x \in X\}$ i.e. the set formed by taking unique elements from array X.

$\ddagger Len(S) =$ Number of elements in set S.

Input

The input format is as below.

N

$A_1\ A_2\ \cdots A_N$

$B_1\ B_2\ \cdots B_N$

Here N is the size of array A and B.

Constraints:

$1 \leq N \leq 2 \times 10^5$

$1 \leq A_i \leq 2N$

$1 \leq B_i \leq 2N$

Output

Print 3 lines. On the first line, print the maximum possible value of $F(A) + F(B)$. On the second and the third line, print array A and B respectively after doing all operations.

Samples

Input	Output
2 1 1 1 2	3 1 2 1 1

Input	Output
1 1 1	2 1 1

In the 1^{st} sample, after swapping A_2 and B_2, $F(A) = 2$ and $F(B) = 1$. Thus $F(A) + F(B) = 3$ which is maximum.

D. Graffiti

Given a **simple** undirected **connected** graph with N vertices. Each vertex will have a value A_i associated with it. You can perform the following operation any number of time **(possibly 0)**:

- Choose two vertices such that they are **connected** by an edge. Replace their value with the **XOR** of their **current** value. For example, choosing u and v will result in, $A_u = A_v = A_u \oplus A_v$

You have to **maximize** the **sum** of the value of all the vertices of the **final** graph. Print the **maximum** sum of value.

Input

The first line will contain two space separated integers N and M denoting the number of vertices and number of edges respectively.

The second line will have N space separated integers $A_1\ A_2 \cdots A_N$ representing the initial values of each vertex.

Next M lines each contains two space separated integers U_i and V_i indicating that there is an edge between those two vertices.

Constraints:

$1 \leq N \leq 2 \times 10^5$

$N - 1 \leq M \leq 3 \times 10^5$

$0 \leq A_i \leq 10^9$

$1 \leq U_i < V_i \leq N$

The graph contains no self-loop or no multi-edge, which means-

$U_i \neq V_i$

$(U_i, V_i) \neq (U_j, V_j)$ where $i \neq j$

Output

Print the **maximum** sum of the value of all the vertices of the **final** graph.

Samples

Input	Output
4 3 1 7 2 8 1 2 2 3 3 4	56

Input	Output
4 4 1 7 2 8 1 2 1 3 2 3 3 4	60

E. Interesting Game

Alice and Bob are once again playing Nim. There are n piles of stones, and i^{th} pile has a_i stones. Alice and Bob take turns, and on each turn, a player must remove a positive number of stones from exactly one pile. The player who cannot make a move because there are no stones left in any pile loses.

But the game is too boring for Bob. To make it more challenging, he decided that he will only make moves on the largest pile during his turn. If there are multiple largest pile during his turn, he can choose any of them. Bob plays optimally under this restriction, choosing to remove stones only from the current largest pile.

At the start of each game, you'll be informed who will play first. Can you find who will win if both players play optimally?

Input

The first line of input contains the number of test cases $1 \leq T \leq 10^5$

Each test case starts with a line with a number $1 \leq n \leq 10^5$

Next line contain n number $1 \leq a_i \leq 10^9$

Then a line contain either “ALICE” or “BOB”, the player who will go first.

Sum of n over all test cases is less than 10^5.

Output

For each test case print who will win in a single line “ALICE” or “BOB”

Samples

Input	Output
2 5 1 2 3 4 5 ALICE 2 10 10 ALICE	ALICE BOB

F. Memoir of Sifat

This is an interactive problem.

Sifat has got a binary string of length N and he wants to play with you. He will keep the string hidden, but not the integer N. To play with him, you will serially ask him some queries. In each query, you will ask him with a binary string of your choice and he will reply whether your given string is a subsequence of the hidden string.

Sifat will not accept more than 1024 queries, however he is always honest.

A subsequence of a string is a string which can be obtained by removing several (possibly zero) characters from the original string without changing the order of characters.

Input

In the first line of input, Sifat will provide you an integer $N (1 \leq N \leq 1000)$.

Then you start the interaction with your queries.

For each interaction, you should print a line consisting of your query binary string S. Length of S must not exceed $N$and it must be **non-empty**. Each query will be replied with a single line.

If S exactly matches with the hidden string, Sifat will reply with "*Correct*".

Otherwise, if S is a subsequence of the hidden string, Sifat will reply with "*Yes*".

Otherwise, Sifat will reply with "*No*".

Your program should get the "Correct" reply within 1024 queries. **You must terminate your program immediately once you get the "Correct" reply. Otherwise you may not get the accepted verdict.**

If you keep asking more than 1024 queries, you may get "Wrong answer"/"CPU Limit Exceeded"/"Runtime Error". Similarly, if your interaction is malformed, you may get "Wrong answer"/"CPU Limit Exceeded"/"Runtime Error".

After outputting each line, don't forget to flush the output. For example:

- fflush(stdout) in C/C++;
- System.out.flush() in Java;
- sys.stdout.flush() in Python;

- flush(output) in Pascal;

Output

There is no more explicit output apart from the interaction in this problem.

Sample interaction: The hidden string in this case is 0110.

Line starting with $>$ denotes Sifat's line, line starting with $<$ denotes your provided line.

> 4

< 010

$> Yes$

< 0101

$> No$

< 011

$> Yes$

< 0110

$> Correct$

G. Power of a string

A string has power p if its can be divided into consecutive blocks of size p and each block consisting of the same character.

For example:

- The string "TTTTAAAAPPPP" has a power of 4 because it can be divided into three blocks of size 4: "TTTT", "AAAA", and "PPPP"
- The string "AAPPXXYY" has a power of 2, as it can be divided into four blocks of size 2: "AA", "PP", "XX", and "YY"
- The string "CCAATTT" has a power of 1 because each character can be treated as a block of size 1

Given, a string S and a integer p, find the length of longest sub-sequence of S which has a power of p or more. If there is no such subsequence, then output 0.

Input

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

For each test case:

- The first line contains a string S consisting of uppercase English letters $(1 \leq |S| \leq 100,000)$.
- The second line contains a integer p $(1 \leq p \leq |S|)$

The total sum of lengths of all strings S across all test cases does not exceed $500,000$

Output

For each test case, output a single integer on a new line, representing the length of the longest subsequence of S that has a power of p or more. If there is no such subsequence, then output 0.

Samples

Input	Output
5 ABACACCBCDBDD 2 AABBCCDD 1 ABCABCABCABC 3 ABABABABABAB 4 AAAAABBBBBCCCCCDDDDD 4	9 8 4 6 20

For the first test case, the string “ABACACCBCDBDD” has a subsequence “AAACCCDDD” whose power is 3 (which is greater than $p = 2$). So, the answer for this case is 9.

In the third test case, the string “ABCABCABCABC” has a subsequence “AAAA” whose power is 4 (which is greater than $p = 3$). So, the answer for this case is 4.

H. Search Engine of Nicola

Being frustrated with captcha of popular search engine Koogle, The great programmer Nicola decided to make her own search engine. In her search engine, all information of the world can be considered as a string S where each topic of searching is a substring of S. There is a special feature in Nicola's search engine: while typing, a user is notified with number of different possible topics based on what she has typed so far. In other words, when a user types a string X, the search engine shows $|X|$ numbers one after another such that i^{th} ($1 \leq i \leq |X|$) number shows the number of **unique** substrings of S which has $X_{1...i}$ (i.e. prefix of X with length i) as prefix. As a search engine, it has to handle plenty of such X (i.e. there are Q queries of X). This problem is very easy for Nicola but she is currently busy. Can you help her by solving it?

Input

First line of input contains the number of test cases, T ($1 \leq T \leq 10$).

Each test case begins with a string S ($1 \leq |S| \leq 10^5$) consisting of lowercase English Letters. In the next line, there is an integer Q ($1 \leq Q \leq 10^5$) denoting the number of query strings. Each of the next Q lines contains a query string X consisting of lowercase English letters ($1 \leq |X| \leq 10^5$).

Over all test cases, $\sum |S| \leq 3 \times 10^5$, $\sum |X| \leq 5 \times 10^5$.

Output

For each query string X, output an array of integer of length $|X|$, where i^{th} ($1 \leq i \leq |X|$) number of the array corresponds to the number of substrings of S of current test case having $X_{1...i}$ as a prefix.

Samples

Input	Output
1 abacb 3 abc ba acb	7 4 0 4 3 7 2 1

For Query 1:

1. Substrings with prefix “a“: ”a” (Appeared multiple times but counted only once), ”ab”, “aba”, “abac”, “abacb”, “ac”, “acb”
2. Substrings with prefix “ab“: ”ab”, “aba”, “abac”, “abacb”
3. No substring with prefix “abc“

I. Tiebreaker the Heartbreaker

After two hours of a football match between Team X and Team Y, it will be penalties to decide who will win. Team X and Team Y will take shots alternatively, Team X will go first. After 5 penalties each, the team scoring higher number of goals from the penalties will win. However, it may be possible that the match does not require all 10 shots if the winner is certain. You will be provided the results of the previous shots of the penalty shootout, you have to decide whether it is possible to determine the result of the match after this shot. In other words:

1. If the current shot is scored, the team taking the shot will win
2. If the current shot is missed, the team taking the shot will lose
3. No particular outcome of the current shot is enough to determine the winner.

Input

The first line will contain the number of test cases $1 \leq T \leq 100$.

Each of the next T lines will describe a test case using a binary string of length $1 \leq len \leq 9$ indicating the results of the previous shots where the character of position 1 is the result of the penalty shot taken by Team X, the character of position 2 is the result of the penalty shot taken by Team Y, the character of position 3 is the result of the penalty shot taken by Team X, the character of position 4 is the result of the penalty shot taken by Team Y and so on. The character is 0 if the shot is missed, 1 if the shot is scored. It is guaranteed that the winner is not decided yet.

Output

For each test case, output a single integer:

- 1: If the current shot is scored, the team taking the shot will win
- 0: If the current shot is missed, the team taking the shot will lose
- -1: No particular outcome of the current shot is enough to determine the winner.

Samples

Input	Output
3 1100 1101011 1011111	-1 1 0

In the first test, scoreline is 1-1, Team X is going to take the shot. Whether they score (1100111110 is a win for Team X, 1100111101 is a win for Team Y) or miss (1100001110 is a win for Team X, 1100001101 is a win for Team Y), the result will not be decided.

In the second test, Team Y will take the shot where the scoreline is Team X 2-3 Team Y. If Team Y score, it will be Team X 2-4 Team Y. Even if Team X score from the last shot, they cannot win.

J. Treasure Hunt: Battle Of Agincourt

Once upon a time, during the tumultuous years of the Hundred Years' War, a pivotal event known as the Battle of Agincourt took place. This battle was fought between the English army, led by King Henry V, and the French knights. Against overwhelming odds, the English emerged victorious, thanks in part to their clever use of longbows and their determination.

After the battle, the English soldiers found themselves in chaotic battlefield, where the French forces had fled, leaving behind their possessions. The area resembled a maze, filled with tents, supplies, and hidden treasures that the French had once guarded. This maze-like structure was dotted with remaining of the battle and opportunities for the English soldiers to collect what they could find.

Among these soldiers was a brave individual who was not only a skilled warrior but also had a keen eye for treasure. He was particularly fascinated by the tales of the riches left behind by the French knights. Legend had it that scattered throughout the battlefield were unique treasures that represented the wealth and power the French had once held.

With a heart full of determination, the soldier set out on a quest to explore this maze of the battlefield. The layout was like a vast grid, where each cell represented a different location filled with potential rewards, such as weapons, armor, and supplies. Each cell (i, j) had a specific treasure value $a_{i,j}$ that symbolized the riches and stories left behind by the fallen knights.

Eager to uncover these treasures, the soldier navigated through the pathways of the battlefield, excited about the adventures that awaited him. Each step taken in this chaotic environment was not just about collecting loot but about embracing the stories and lessons from the battle that would stay with him forever.

Starting from the top-left corner of the grid $(1, 1)$, the soldier could only move down or right. As he collected treasures from the cells along his journey, he aimed to **maximize** his total profit. He understood that the treasures he gathered were not merely material wealth but a reflection of the hard-fought victory and the history that surrounded him.

The total profit the soldier aimed to achieve was calculated using the following formula:

$$Total\ Profit = (a_{C_1} + a_{C_2} + \ldots + a_{C_k}) - \\ (|C_2 - C_1| + |C_3 - C_2| + \ldots + |C_k - C_{k-1}|)$$

Here, $C_1, C_2, C_3, \ldots.., C_k$ represented the cells he took treasure from, and $|C_i - C_j|$ denoted the Manhattan distance between the cells.

The soldier's mission was clear: to traverse the maze-like battlefield, gathering treasures while reflecting on the lessons learned from the great battle of Agincourt, which was **maximizing** the profit.

Input

First line of the input contains an integer T, the number of test cases.

The first line of each test case contains two space separated integers N and M which represents dimensions of the grid.

Each of the next N lines contain M space separated values, where $a_{i,j}$ represents the treasure value at the cell in i-th row and j-th column.

Constraints

$1 \leq T \leq 1000$

$1 \leq N \leq 1000$

$1 \leq M \leq 1000$

$0 \leq a_{ij} \leq 10^9$

Sum of $N * M$ of all test cases will not exceed 10^6.

Output

For each test case output a single integer the maximum profit he can earn.

Samples

Input	Output
5 4 3 8 4 4 1 1 5 7 9 9 5 5 6 2 4 5 9 7 8 1 2 9 5 1 3 1 9 4	35 31 12 23 39

Input	Output
5 1 0 9 9 4 4 3 5 1 4 8 8 4 8 3 0 8 7 2 0 0 8 8	

K. Ultra-Secured Lab

Professor Lukaku is going to build a lab on a convex polygon region with n vertices for his experimental purpose. To ensure the confidentiality of his works, he needs to build wall vertical with polygon plane (i.e. in our case, the polygon is on XY-plane and the wall height is expanded along positive Z-axis) having negligible density along the edges of the polygon. However, Professor Lukaku has a weird requirement regarding the wall:

- For each possible pair (P, B) where P belongs to the set of all points inside the polygon (**including points on the edges of the polygon**) and B belongs to the set of all points on the edges of the given polygon, if the distance between P and B is d, the height of the wall at B must be at least d^2.

The height of the wall need not to be uniform. You need to figure out the minimum possible surface area of the wall.

Input

The first line contains an integer, $1 \leq T \leq 2000$, the number of test cases.

For each test case, the first line contains an integer, the number of vertices of the convex polygon, $3 \leq n \leq 500$. Each of the next n lines contains two integers: $-5 \times 10^4 \leq x \leq 5 \times 10^4$ and $-5 \times 10^4 \leq y \leq 5 \times 10^4$ describing a vertex of the polygon. Vertices are given in counterclockwise order. No three consecutive points of a polygon will lie on the same line.

Output

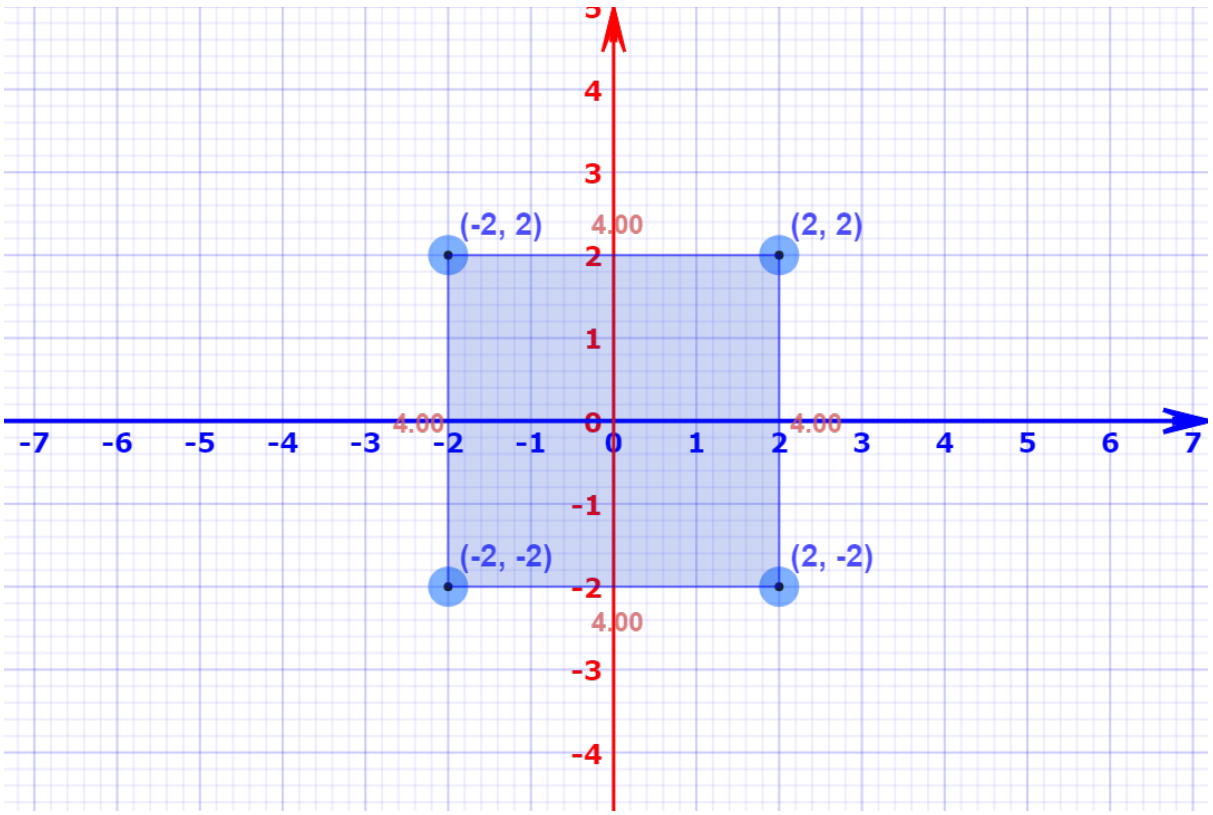
For each test case, output the minimum possible area of the wall. Your output will be considered correct if the absolute or relative error is not larger than 10^{-6}.

Samples

Input	Output
3 4 -2 -2 2 -2 2 2 -2 2 4	405.33333333 3242.66666667 1440.00000000

Input	Output
-4 -4 4 -4 4 4 -4 4 4 -4 -2 4 -2 4 2 -4 2	

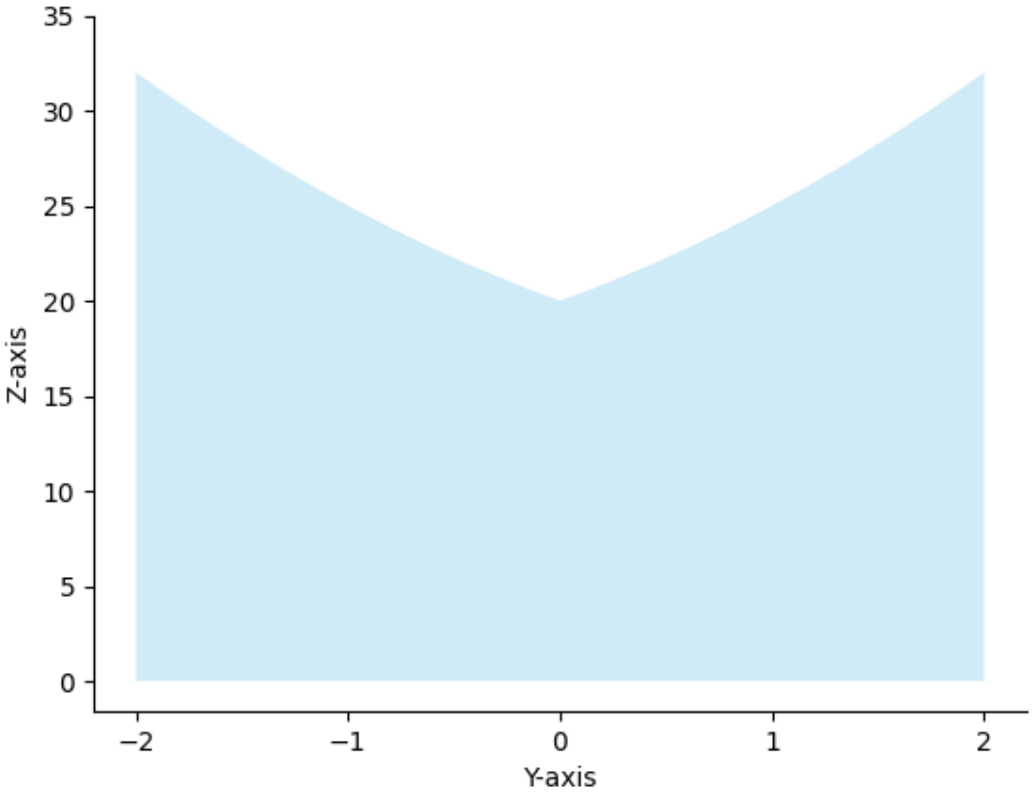
The polygon of the first case:



For this case:

- Wall height at point $(2, 1)$ is 25.
- Wall height at point $(2, -2)$ is 32.
- Wall portions on four edges are identical. So we are showing the wall portion from $(2, -2)$ to $(2, 2)$. The height of the wall in this part z can be represented as, $z = (y - 2)^2 + 16$ when $-2 \leq y < 0$, $z = (y + 2)^2 + 16$ when $0 \leq y \leq 2$, $z = 0$ otherwise. The wall area of this portion is $\frac{304}{3}$. So the total area is $4(\frac{304}{3}) = \frac{1216}{3}$.

The wall portion from $(2, -2)$ to $(2, 2)$ is shaded in the following figure:


35
30
25
20
15
10
5
0
Z-axis
−2
−1
0
1
2
Y-axis

L. Will Power

Dividing wealth is tricky. When people die, they leave their property to their children, who often end up fighting over it. Sometimes, part of the inheritance even goes to charity, making the whole process more complicated.

We will automate the process of wealth division within a family tree. Initially, there is only one node (Node 1) representing the original ancestor. Consider Node 1 to have **no wealth**. As time progresses, new nodes (children) will be added. Each node is assigned a certain amount of wealth at birth. When a node u dies, its wealth is distributed evenly among its children. If u has a child v who has already passed away, then v's share of u's wealth is divided equally to v's children. This process continues recursively until all shares have been distributed or until a deceased node has no living descendants, in which case the wealth is given to charity.

There will be three types of queries:

1. **Type 1: A Node Dies**
   This records a node's death and distributes its wealth among its children.

2. **Type 2: A Child is Born**
   This adds a new node x to the family tree and assigns them a set amount of wealth y. **A dead node cannot give birth to another node**.

3. **Type 3: Calculate Wealth**
   This calculates the current wealth of a **alive** node in the family tree. Since the answer can be irrational, it must be output in the format $P \times Q^{-1} \mod M$ if the answer is $\frac{P}{Q}$ where $M = 10^9 + 7$.

Input

The input consists of multiple test cases. Each test case starts with an integer t (the number of test cases). For each test case:

1. The first line contains an integer $q (2 \leq q \leq 2 \times 10^5)$, representing the number of queries.

2. The next q lines contain the queries in the following format:

- **Type 1 Query**: An integer 1 followed by an integer x (the number of the deceased node, where $x \leq q + 1$ and x is alive in the tree).

- **Type 2 Query**: An integer 2 followed by three integers x (the node number, where $x \leq q+1$and x was never added to the tree), y (the wealth of the node, where $0 \leq y \leq 10^6$) and parent node p (p is alive and present in the tree).
- **Type 3 Query:** An integer 3 followed by an integer x (the number of the node, where $x \leq q+1$and x is alive).

It is guaranteed that there are no invalid queries. The sum of q over all test cases doesn't exceed 2×10^5.

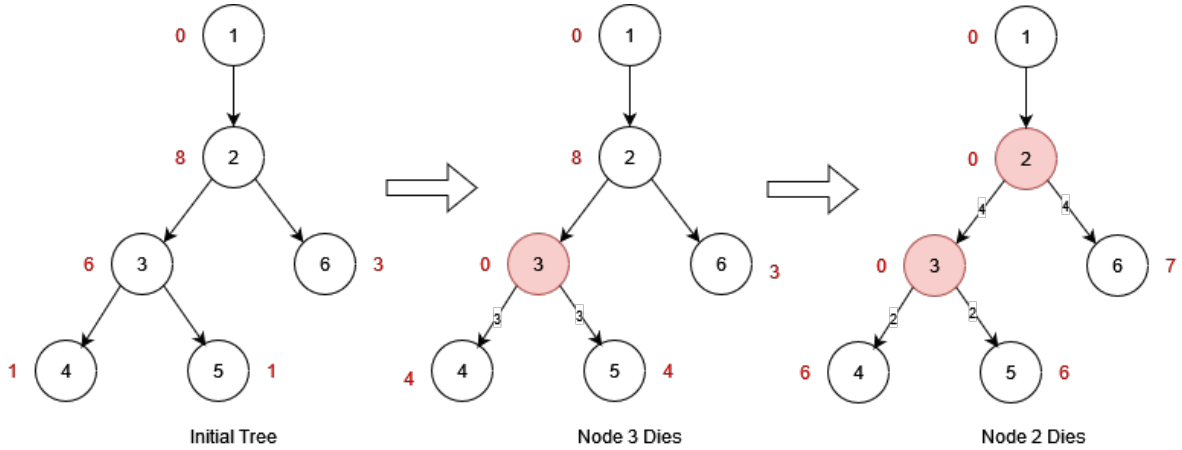
Output

For Type 1 Query and Type 2 Query, no output is required. For Type 3 Query, output the current wealth of the specified node in the format $P \times Q^{-1} \mod M$where the answer is $\frac{P}{Q}$ and $M = 10^9 + 7$

Samples

Input	Output
1 13 2 2 8 1 2 3 2 2 2 4 5 3 2 8 10 3 2 5 6 2 2 6 5 5 2 7 5 5 1 2 3 5 3 3 1 5 3 6 3 7	10 6 10 10

Input	Output
1 9 2 2 8 1 2 3 6 2 2 4 1 3 2 5 1 3 2 6 3 2 1 3 1 2 3 4 3 6	6 7



This diagram explains the distribution of wealth in the 2nd Sample IO.