Problem A. Two Triangles

Input file:	`area.in`
Output file:	`standard output`
Balloon Color:	`Pink`

Ehab was wondering what is the area of the shape that is formed by two right-angled triangles, he knows that the formula for the area of a triangle is half of the triangle base multiplied by the triangle height.

So he asked you to calculate **double** the area of a given right-angled triangle.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case will consist of one line that contains two space separated integers b and h ($1 <= b, h <= 100$) the base and height of the triangle respectively.

Output

For each test case print a single integer, **double** the area of the given right-angled triangle.

Example

area.in	standard output
1 3 6	18

Problem B. Good Powers of Two

Input file:	`even.in`
Output file:	`standard output`
Balloon Color:	`Yellow`

It's well known that any integer can be represented as a sum of distinct powers of two. For example, 63 can be represented as a sum of powers of 2 as: $2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5$. While 10 can be respresented as: $2^1 + 2^3$

It's also well known since we were young, that even numbers are good numbers, while odd numbers are evil, that's why we will only look for even numbers.

Given an integer n, in its representation as a sum of distinct powers of two, count the number of powers of two (2^x) in that representation, where x is **even**.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case consists of a single line that contains an integer n ($0 \le n \le 10^{18}$)

Output

For each test case print a single integer, the answer to the problem.

Example

even.in	standard output
2 63 10	3 0

Problem C. The End of Happy land

Input file:	`happy.in`
Output file:	`standard output`
Balloon Color:	`White`

Happy land consists of n cities numbered from 1 to n, and $n-1$ bidirectional roads connecting the cities. You can move from a city to another city if there is a road that connects the two cities.

It's guaranteed that any city in Happy land can be reached from any other city

You, as a neutral party, have been selected to partition the land to create two new countries (Funland, and Nufland) in a way that satisfies everyone.

After partitioning, any road between two cities that belong to different countries must have border control.

For a partitioning to be satisfactory, the following conditions must be met:

- Each city belongs to either Funland or Nufland.
- There must be at most one road that will have border control
- Nufland insists that the distance between any two cities in her claim be at most x roads.

You need to find the total number of different partitions possible to present in your next meeting.

Two ways of partitioning are different, if there is at least one city that belongs to either country in one partition and belongs to the other country in another partition.

Input

The first line of the input contains the number of test cases t. Each test case starts with two space-separated integers n and x $(1 \le n \le 10^3, 0 \le x \le 10^3)$.

Then $n-1$ lines follow, each line contains two space-separated integers u and v $(1 \le u, v \le n, u \ne v)$, which means there is a road between cities u and v.

Output

Print a single integer, the answer to the problem.

Example

happy.in	standard output
2 5 1 1 2 2 3 3 4 4 5 3 100 1 2 1 3	4 5

Note

For the second example, all combinations of cities are valid, except for two:

- {1} Funland, {2, 3} Nufland

- {2, 3} Funland, {1} Nufland

as in both of those partitions, more than one road will have border control.

Problem D. Broke Collector

Input file:	`collector.in`
Output file:	`standard output`
Balloon Color:	`Violet`

You are a broke collector who lives on a straight road with many shops. n interesting items are currently on sale, the ith item is at the shop at block x_i on the street.

You estimate the collector's value of the ith item to be v_i and it's currently sold for c_i coins, lucky for you, every minute all shops drop their prices down by 1 coin. You know that if the price falls to 0 other collectors will claim it immediately, so you are willing to pay 1 coin for any item.

Each minute you can move to the block on your right, or the block on your left, or stay in place (i.e. move from x to $x+1$ or $x-1$ or stay at x)

Note that you can only buy an item if you are at the shop it's sold at when the item costs 1 coin.

You can start from any block on the street, what is the maximum possible total collector's value that you can acquire?

Input

The first line of the input contains integer t, the number of test cases.

Each test case begins with an integer n ($1 \le n \le 1000$) number of items.

n lines follow, each line contains three integers x_i, c_i, v_i ($1 \le x_i, c_i \le 1000$) ($1 \le v_i \le 10^9$). Indicating that the ith item is sold at the shop at block x_i, sold for c_i coins and has a collector's value of v_i.

It is guaranteed that for each test case, no two items are sold at the same shop at the same price.

Output

Print only one integer the maximum collector's value that you can acquire.

Example

collector.in	standard output
2 3 1 1 1 2 2 2 3 3 3 4 2 2 7 1 3 3 3 3 13 4 1 10	6 23

t-1

	1	2	3	4
1	3	7	13	0
2				
3				
4				

Note

In the first test case you can start at $x = 1$ buying the first item directly, then in one minute move to $x = 2$ buying the second item and then move to $x = 3$ buying the third item.

In the second test case, you can start at $x = 4$ and buy the fourth item ($v = 10$) then move to $x = 3$ in one minute, stay there for another minute, then buy the third item ($v = 13$)

	1	2	3	4
0	20	13	13	23
1	3	20	13	0

Problem E. Fibonacci Collection

Input file:	`fib.in`
Output file:	`standard output`
Balloon Color:	`Cyan`

You are an esteemed collector, you decided to start collecting numbers, your friends suggested Fibonacci numbers and that is not such a bad idea. As a reminder, Fibonacci numbers follow this definition:

$$F_1 = F_2 = 1, F_n = F_{n-1} + F_{n-2} (n > 2)$$

The first few elements of the sequence are: $1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$

You will be given m queries of two types.

type 1) 1 n - add the nth fibonacci number to your collection.

type 2) 2 n - upgrade all your collection by n steps, such all your F_k become F_{k+n}.

After each query, output the current sum of your collection. Since the sum can be large, output the sum modulo $10^9 + 7$.

For instance, if your collection consists of $\{F_4, F_8, F_9\}$ (sum: 58) after query 1 4, you have $\{F_4, F_4, F_8, F_9\}$ (sum: 61). After query 2 3, you have $\{F_7, F_7, F_{11}, F_{12}\}$ (sum: 259)

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case will consist of two lines, The first line of the each case consists of a single integer m the number of queries $(1 \le m \le 5 \times 10^5)$.

Then m lines, each is T n $(T \in \{1, 2\}, 1 \le n \le 10^6)$ describing the query as stated in the statement.

Output

For each test case answer each query with a single integer: the sum of the collection modulo $10^9 + 7$

Example

fib.in	standard output
2 6 1 4 1 8 1 9 1 4 2 3 1 1 2 2 5 1 10	3 24 58 61 259 260 0 55

Note

Warning: large Input/Output data, be careful with certain languages.

Problem F. Semi-neat Numbers

Input file:	`neat.in`
Output file:	`standard output`
Balloon Color:	`Black`

Neat numbers are non-negative integers **without leading zeros** that consist of non-decreasing digits (e.g. 12236, 14599 are neat numbers, while 01, 10, 1232 are not).

A Semi-neat number is a non-negative integer that can be split into exactly x contiguous non-overlapping neat numbers.

Given a possibly very large integer number n, and the integer x, check whether n is a semi-neat number or not, according to the definitions mentioned above, and if it is, print any x neat numbers you can get after splitting n.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case will consist of two lines, The first line of the input contains the integer n ($0 \le n \le 10^{100000}$).

The second line contains the integer x ($1 \le x \le floor(log_{10} n) + 1$).

Output

For each test case, print a single line contains x space-separated integers, the answer to the problem, or 1"(without quotes), if it's impossible to find an answer.

All numbers must have no leading zeros.

If there is more than one solution, you can print any of them.

You must output the neat numbers ordered according to their positions in the given number.

Example

neat.in	standard output
3 2450 3 101 2 1234 3	2 45 0 -1 1 2 34

Note

Another valid solution for the first example test case is 24 5 0.

Problem G. The Encoder

Input file:	`encoder.in`
Output file:	`standard output`
Balloon Color:	`Purple`

Yasser decided to encode some string s using a key character key, by inserting 'k' after each occurrence of key in the string (e.g. if key is 'a', and s is "abcdaf then encoding s using key will produce "akbcdakf").

Yasser was very happy with this encoding algorithm till he found out that it's so trivial that even Omar was capable of breaking it. So he decided to make the encoding more complex by choosing a key string $keys$, consisting of unique characters, and then he applies the encoding by inserting 'k' after each occurence of any character in $keys$.

For example, if s is "**abcdaf**" and $keys$ is "**a**" then the encoding s will produce "**akbcdakf**".

Also if $keys$ is "**ac**", then the encoding will produce "**akbckdakf**", by inserting 'k' after every character that is either 'a' or 'c'.

Of course this one couldn't be broken by Omar, so he seeks help from you to break it!

Given the encoded string s and the key string $keys$, find the original string or state that such string doesn't exist.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case consists of two lines, The first line contains the string s ($1 \leq |s| \leq 10^5$). Each character of s is a lowercase English letter.

The first line contains the string $keys$ ($1 \leq |keys| \leq 26$). Each character of $keys$ is a distinct lowercase English letter.

Output

For each test case output a single line, the original string, or "IMPOSSIBLE"(without quotes), if it's impossible to decode the given string.

Example

encoder.in	standard output
2 abc c akbkck abcde	IMPOSSIBLE abc

oblem H. Rick's Puzzle

Input file:	`puzzle.in`
Output file:	`standard output`
Balloon Color:	`Gold`

Rick is a genius scientist, the smartest in the galaxy. He travels with his grandson Morty. Morty is not as smart as his grandfather Rick, so Rick wanted to keep Morty busy to not disturb him with his stupid questions.

Rick, the genius, built a puzzle box to keep morty busy solving it, the puzzle box consist of an array of integers A of length N, the i^{th} integer has a value of A_i. To solve the puzzle Morty needs to give the answer of the puzzle, that is the maximum sum of the final array possible, after doing the following operation **zero or more** times:

In one operation it is allowed to select two indices i and j ($i \neq j$), remove them both from the array, then insert a new integer X, (if exist a valid X), where X is a non-trivial divisor of $A_i \times A_j$.

Let us remind you that a number's divisor is called non-trivial if it is different from one and from the divided number itself.

Morty kept trying for hours now but was not able to solve it, so he asked you to print the sum modulo 1000000007($10^9 + 7$) since the answer can be rather large.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case consists of two lines, the first line contains an integer n ($1 \leq n \leq 10^5$) representing the array size. The second line contains n space separated integers, represents the values of the array A ($1 \leq A_i \leq 10^5$).

Output

For each test case output a single integer, the maximum sum of the final array after applying the mentioned operation 0 or more times. Since the answer can be rather large print the answer modulo 1000000007($10^9 + 7$)

Example

puzzle.in	standard output
2 2 10 20 3 9 16 25	100 900

Problem I. RMQ .. But?

Input file:	`rmq.in`
Output file:	`standard output`
Balloon Color:	`Light Green`

Given an array a of n integers $(a_1, a_2, a_3, ..., a_n)$.

For a given l and r, lets define a F function that calculates the maximum integer in the range starting at l and ending at r, in other words $F(l, r) = max(a_l, a_{l+1}, ..., a_{r-1}, a_r)$.

You need to perform q queries of the following types:

type 1) $1\ l\ r$: For all $l \leq i \leq r$, set $a_i := F(l, r) - a_i$.

type 2) $2\ l\ r$: Print the value of $F(l, r)$.

Input

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

Each test case begins with two integers n, q $(1 \leq n, q \leq 10^5)$ denoting size of array and number of queries respectively.

The second line of each case contains n integers a_i $(1 \leq a_i \leq 10^5)$.

q lines follow, each containing three integers $type, l, r$ $(type \in \{1, 2\}, 1 \leq l \leq r \leq n)$.

Output

For each test case answer all queries of type 2 with a line containing a single integer: the value of $F(l, r)$.

Example

rmq.in	standard output
1 5 6 1 2 3 4 5 1 1 3 2 1 1 2 2 2 2 3 3 1 1 5 2 1 5	2 1 0 5

Note

Initially, the array is $[1, 2, 3, 4, 5]$, after the first query the array becomes $[2, 1, 0, 4, 5]$. After the fifth query, the array becomes $[3, 4, 5, 1, 0]$.

blem J. Crashing Robots

Input file: `robots.in`
Output file: `standard output`
Balloon Color: `Blue`

One day, Rick, the scientist, was bored so he built n robots to play with, he set the robots on his garage floor, which can be seen as an infinite 2D plane, and gave each robot an instructions command consisting of letters 'U', 'D', 'L' and 'R'. Any step takes exactly 1 second for the robot to do and all robots move simultaneously.

- 'U' represents a step up.
- 'D' represents a step down.
- 'L' represents a step left.
- 'R' represents a step right.

If two robots or more became in the same point at any second, they will **crash and stop moving** and will remain in this position forever. Also if a robot finished his instructions he will stop moving and will remain in this position forever.

Rick never built a regular robot, his robots are always interesting, thats why he made the robots movement to be instant (as a stop-motion movement) so when the robot is instructed to move in a direction, the robot disappears from his current position and appears in the adjacent position. In other words all crashing accidents happen in integer coordinates only.

Rick got bored quickly and did not run the robots, so he asked you to calculate for him the number of robots that are going to crash.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case starts with a line contains a single positive integer n ($1 \leq n \leq 1000$) — the number of robots.

The following n lines contains n non-empty strings — the instructions for every robot. Each string length will not exceed 1000 characters, and consisting of only instruction characters (U, D, L, R)

Then follows n other lines, each line containing 2 integers x and y ($-1000 \leq x, y \leq 1000$) — representing the initial position of each robot. It's guaranteed that all the initial positions are pairwise distinct.

Output

For each test case print a single integer — the number of robots that are going to crash.

Example

robots.in	standard output
2 3 U L DLL 0 0 1 1 2 2 1 LLDLL 0 0	3 0

Problem K. Spaceship Travel

Input file:	`space.in`
Output file:	`standard output`
Balloon Color:	Red

In the future world, humans may settle inside an enormous spaceship that allows up to 8×10^{24} humans at once!

The spaceship consists of chambers, each chamber occupies a $1 \times 1 \times 1$ unit of space. To ensure maximum capacity, the chambers are packed close to each other in a cubic grid. Thus, we can imagine the chambers organized in a 3D grid and we can refer to each chamber by its coordinates in 3D space relative to the central room of the spaceship; the chamber at $(0, 0, 0)$.

All chambers have exists in all directions, including vertically. A traveler can move from one chamber to an adjacent one in 1 unit of time. A chamber (x, y, z) is adjacent to $(x \pm 1, y, z)$, $(x, y \pm 1, z)$ and to $(x, y, z \pm 1)$. A traveler can move freely across adjacent chambers with each move taking 1 unit of time. So, for instance, to move from chamber $(3, 2, -1)$ to $(1, 4, 3)$ a traveler needs 8 units of time.

For added security, n police stations have been established at various chambers in the spaceship. Patrols need to be able to move efficiently from any police station to another. So they are asking you to help them solve this problem: from each station, which station takes the most time to travel to? The stations are given to you in decreasing order of their priority, in case of multiple possible answers choose the station with the highest priority.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case will consist of two lines, The first line of the input consists of a single integer n the number of police stations $(2 \leq n \leq 5 * 10^5)$. Follows are n lines the ith of which consists of three integers x_i, y_i, z_i $(-10^8 \leq x_i, y_i, z_i \leq 10^8)$ which means that the ith police station is in chamber (x_i, y_i, z_i)

Output

Output n integers v_i $(1 \leq v_i \leq n)$ such that v_i is the highest priority police station that is farthest away from ith station.

Example

space.in	standard output
2 3 5 7 4 1 2 3 9 8 7 4 5 7 9 -3 2 -3 1 4 3 -2 -8 -6	2 3 2 4 1 4 1

Problem L. Pattern Construction

Input file:	`pattern.in`
Output file:	`standard output`
Balloon Color:	`Orange`

Given two strings s of length n and pat of length m.

For every index i from 1 to n, you have to find the minimum index j such that $(i \le j \le n)$ such that a substring $[i, j]$ can be used to construct the pattern string pat.

When you select a substring you are allowed to perform the following operations if needed:

- Reorder the substring characters

- Delete some of the substring characters

- Select a character from the substring and change all occurrences of that character to any other character. Note that you can use this operation **at most once** per index i.

Your task is to calculate for each index i the minimum index j such that a substring from i to j can construct the pattern pat after applying the mentioned operations, or -1 if there is no valid j.

Input

The input starts with an integer T, number of test cases. Then T test cases follow.

Each test case consists of two lines, first line will be string s $(1 \le |s| \le 10^5)$ and the second line will be the string pat $(1 \le |pat| \le 10^5)$.

All characters in the input are lowercase English characters.

Output

For every test case print a single line contains n space separated integers, the answer for each index i.

Examples

pattern.in	standard output

`1`
`bcbca`
`aca`
	`3 5 5 -1 -1`

`2`
`baabb`
`c`
`abccba`
`abc`
	`1 2 3 4 5`
	`3 6 6 6 -1 -1`

Note

In the first test case:

For i = 1: select j = 3 and change all occurrence of 'b' with 'a'.

For i = 2: select j = 5 and change all occurrence of 'b' with 'a'.

For i = 3: select j = 5 and change all occurrence of 'b' with 'a'.

For i = 4 and i = 5: you can't find valid subarray starting from i, so print -1 for both.