Problem A. A+B Problem

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

You are given two integers A and B. Your task is to calculate the sum of these two integers.

Input

The first line of the input contains two integers A and B $(-10^9 \le A, B \le 10^9)$.

Output

Output a single line contains a single integer, indicating the answer.

Example

standard input	standard output
`2669 -154`	`2515`

Problem B. Best Interactive Player

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

This is an interactive problem.

There is a hidden integer x where $1 \le x \le 10^9$.

You want to find it by asking whether some integer y is bigger than x or not.

You can query in the format "? y", and the interactor will respond to you 1 when $y > x$ and 0 when $y \le x$.

Interaction Protocol

You can make no more than 100 queries. To make a query, output "? y" ($1 \le y \le 10^9$) on a separate line, then you should read the response from standard input.

To give your answer, print "! x" on a separate line. The output of the answer is **not** counted towards the limit of 100 queries.

After that, your program should terminate.

After printing a query, do not forget to output end of line and flush the output. To do this, use `fflush(stdout)` or `cout.flush()` in C++, `System.out.flush()` in Java, `flush(output)` in Pascal, or `stdout.flush()` in Python.

Example

standard input	standard output
	? 1
0	
	? 2
0	
	? 3
0	
	? 4
1	
	! 3

Problem C. Can You Guess My Sequence?

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

This is a communication problem.

Alice has a sequence $a_1, a_2, \cdots, a_n$ with the following properties.

- $a_1 < a_2 < \cdots < a_n$
- $0 \le a_i < 20$, for all $1 \le i \le n$.

Alice wants to transfer her sequence to Bob. To do that, she can transfer an integer x within $[0, 10^9]$ to Bob. Help them to find any strategies!

Interaction Protocol

In this problem, your solution will be run twice in each test.

First Run

In the first run, your solution will play the role of Alice. You should read the sequence from the input, and output a single integer to indicate the number you would like to transfer.

Input

The first line of the input contains a single string `Alice`, indicating the role of this run.

The next line of the input contains a single integer n, indicating the length of the sequence.

The next line of the input contains n integers $a_1, a_2, \cdots, a_n$, indicating the sequence.

It is guaranteed that $a_1 < a_2 < \cdots a_n$ and $0 \le a_i < 20$ for all $1 \le i \le 20$.

Output

Output a single line with a single integer x ($0 \le x \le 10^9$), indicating the number Alice should transfer to Bob.

Second Run

In the second run, your solution will play the role of Bob. You should read the number x from the input, and output the sequence you recovered.

Input

The first line of the input contains a single string `Bob`, indicating the role of this run.

The next line of the input contains a single integer x ($0 \le x \le 10^9$), indicating the number Alice transfered to Bob.

Output

The first line of the output contains a single integer n, indicating the length of the sequence you recovered.

The next line of the output contains n integers $a_1, a_2, \cdots, a_n$, indicating the sequence you recovered.

Examples

standard input	standard output
Alice 6 2 3 5 8 10 15	1024
Bob 1024	6 2 3 5 8 10 15

Problem D. Do You Like This Problem?

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

This is an output-only problem.

Are you tired on solving traditional competitive programming tasks? No worries! Here's a problem about ordering contest names!

As you might know, there are 23 stages in the last season of the Universal Cup, including the trial stage. Can you sort those stages in order of when they were held?

A . America

B . Gomel

C . Guangzhou

D . Hangzhou

E . Hong Kong

F . Iberia

G . India

H . Nanjing

I . Ookayama

J . Osijek

K . Poland

L . Qingdao

M . Ranoa

N . Shaanxi

O . Shandong

P . Shanghai

Q . Shenyang

R . Shenzhen

S . Slovenia

T . Taiwan

U . Ukraine

V . Zaporizhzhia

W . Zhejiang

Output

This is an output-only problem. It means you do not need to submit any programs. The only thing you need to submit is the ansewr file it self.

The answer file should contains a single line including a single string, indicating the sorted result.

Please note that the unique letter ID of the stage should be used to represent it.

Example

standard input	standard output
<no input>	EDCBAFGHIJKLMNOPQRSTWVU

Note

The example output does not solve the problem, it is given only to demonstrate the format.

Problem E. Quine

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

> "*A quine is a computer program which takes no input and produces a copy of its own source code as its only output. The standard terms for these programs in the computability theory and computer science literature are "self-replicating programs "self-reproducing programs and "self-copying programs".*"
>
> Wikipedia - Quine (computing)

Write a program that can output its own source code.

The code must contain at least 10 visible characters.

Input

There's no input for this problem.

Output

Output the source code of the program.

The output of the program must be identical to the source code you submitted.

Example

standard input	standard output

`<no input>`	`#include <cstdio>`
	
	`int main() {`
	`    printf("Hello World!\n");`
	`}`

Note

The example output does not solve the problem, it is given only to demonstrate the format.

Problem F. Traveling in Cells

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

There are n cells arranged in a row. The i-th cell has a color c_i and contains a ball with value v_i.

You're going to travel several times in the cells. For each travel, you'll be given an integer x and a set of colors $\mathbb{A} = \{a_1, a_2, \cdots, a_k\}$ where $c_x \in \mathbb{A}$. The travel starts from cell x. During the travel, if you're located in cell i you can next move to cell $(i-1)$ or $(i+1)$. Note that you can't move out of these n cells. Also at any time, the color of cell you're located in must belong to set $\mathbb{A}$.

When you're in cell i, you can choose to remove the ball in the cell and gain its value v_i. As there is only one ball in each cell, you can only remove the ball from each cell once.

Your task is to process q operations in order. Each operation is one of the following three types:

- $1\ p\ x$: Change c_p to x.
- $2\ p\ x$: Change v_p to x.
- $3\ x\ k\ a_1\ a_2\ \ldots\ a_k$: Given the starting cell x and the color set $\mathbb{A} = \{a_1, a_2, \cdots, a_k\}$ of a travel, imagine that you're going on this travel, calculate the maximum total value you can gain. Note that this travel is only an imagination, thus the balls won't be truely removed. That is, all queries are independent.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains two integers n and q ($1 \le n \le 10^5$, $1 \le q \le 10^5$) indicating the number of cells and the number of operations.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le n$) where c_i is the initial color of the i-th cell.

The third line contains n integers $v_1, v_2, \ldots, v_n$ ($1 \le v_i \le 10^9$) where v_i is the initial value of ball in the i-th cell.

For the following q lines, the i-th line describes the i-th operation. The input format is listed as follows:

- $1\ p\ x$: $1 \le p \le n$ and $1 \le x \le n$.
- $2\ p\ x$: $1 \le p \le n$ and $1 \le x \le 10^9$.
- $3\ x\ k\ a_1\ a_2\ \ldots\ a_k$: $1 \le x \le n$, $1 \le a_1 < a_2 < \ldots < a_k \le n$ and $c_x \in \{a_1, a_2, \cdots, a_k\}$.

It's guaranteed that neither the sum of n nor the sum of q of all test cases will exceed 3×10^5. Also the sum of k of all test cases will not exceed 10^6.

Output

For each operation of type 3 output one line containing one integer, indicating the maximum total value you can gain.

Example

standard input	standard output
2 5 10 1 2 3 1 2 1 10 100 1000 10000 3 3 1 3 3 3 2 2 3 2 5 20000 2 3 200 3 3 2 1 3 3 3 3 1 2 3 1 3 4 2 1 100000 1 2 2 3 1 2 1 2 4 1 1 2 3 4 1000000 1000000 1000000 1000000 3 4 4 1 2 3 4	100 110 1200 21211 100010 4000000

Problem G. Matching

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

Given an integer sequence $a_1, a_2, \cdots, a_n$ of length n, we construct an undirected graph G from the sequence. More precisely, for all $1 \le i < j \le n$, if $i - j = a_i - a_j$, there will be an undirected edge in G connecting vertices i and j. The weight of the edge is $(a_i + a_j)$.

Find a matching of G so that the sum of weight of all edges in the matching is maximized, and output this maximized sum.

Recall that a matching of an undirected graph means that we choose some edges from the graph such that any two edges have no common vertices. Specifically, not choosing any edge is also a matching.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \le n \le 10^5$) indicating the length of the sequence.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($-10^9 \le a_i \le 10^9$) indicating the sequence.

It's guaranteed that the sum of n of all test cases will not exceed 5×10^5.

Output

For each test case output one line containing one integer indicating the maximum sum of weight of all edges in a matching.

Example

standard input	standard output
3 9 3 -5 5 6 7 -1 9 1 2 3 -5 -4 -3 3 1 10 100	30 0 0

Note

For the first sample test case, the optimal choice is to choose the three edges connecting vertex 3 and 5, vertex 4 and 7, and finally vertex 8 and 9. The sum of weight is $(5 + 7) + (6 + 9) + (1 + 2) = 30$.

For the second sample test case, as all edges have negative weights, the optimal matching should not choose any edge. The answer is 0.

For the third sample test case, as there is no edge in the graph, the answer is 0.

Problem H. Math Problem

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

Given two positive integers n and k, you can perform the following two types of operations any number of times (including zero times):

- Choose an integer x which satisfies $0 \le x < k$, and change n into $k \cdot n + x$. It will cost you a coins to perform this operation once. The integer x you choose each time can be different.
- Change n into $\lfloor \frac{n}{k} \rfloor$. It will cost you b coins to perform this operation once. Note that $\lfloor \frac{n}{k} \rfloor$ is the largest integer which is less than or equal to $\frac{n}{k}$.

Given a positive integer m, calculate the minimum number of coins needed to change n into a multiple of m. Please note that 0 is a multiple of any positive integer.

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 10^5$) indicating the number of test cases. For each test case:

The first line contains five integers n, k, m, a, b ($1 \le n \le 10^{18}$, $1 \le k, m, a, b \le 10^9$).

Output

For each test case output one line containing one integer, indicating the minimum number of coins needed to change n into a multiple of m. If this goal cannot be achieved, output -1 instead.

Example

standard input	standard output
4 101 4 207 3 5 8 3 16 100 1 114 514 19 19 810 1 1 3 1 1	11 2 0 -1

Note

For the first sample test case, initially $n = 101$. The optimal steps are shown as follows:

- Firstly, perform the second type of operation once. Change n into $\lfloor \frac{n}{4} \rfloor = 25$. This step costs 5 coins.
- Then, perform the first type of operation once. Choose $x = 3$ and change n into $4 \cdot n + 3 = 103$. This step costs 3 coins.
- Then, perform the first type of operation once. Choose $x = 2$ and change n into $4 \cdot n + 2 = 414$. This step costs 3 coins.
- As $414 = 2 \times 207$, n is a multiple of m. The total cost is $5 + 3 + 3 = 11$ coins.

For the second sample test case, perform the second type of operation twice will change n into 0. The total cost is $1 + 1 = 2$ coins.

For the third sample test case, as $n = 114 = 6 \times 19$ is already a multiple of m, no operation is needed. The total cost is 0 coins.

Problem I. Fast and Fat

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

You're participating in a team competition of trail running. There are n members in your team where v_i is the speed of the i-th member and w_i is his/her weight.

The competition allows each team member to move alone or carry another team member on their back. When member i carries member j, if member i's weight is greater than or equal to member j's weight, member i's speed remains unchanged at v_i. However, if member i's weight is less than member j's weight, member i's speed will decrease by the difference of their weight and becomes $v_i - (w_j - w_i)$. If member i's speed will become negative, then member i is not able to carry member j. Each member can only carry at most one other member. If a member is being carried, he/she cannot carry another member at the same time.

For all members not being carried, the speed of the slowest member is the speed of the whole team. Find the maximum possible speed of the whole team.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($1 \le n \le 10^5$) indicating the number of team members.

For the following n lines, the i-th line contains two integers v_i and w_i ($1 \le v_i, w_i \le 10^9$) indicating the speed and weight of the i-th member.

It's guaranteed that the sum of n of all test cases will not exceed 10^5.

Output

For each test case output one line containing one integer indicating the maximum speed of the whole team.

Example

standard input	standard output
2 5 10 5 1 102 10 100 7 4 9 50 2 1 100 10 1	8 1

Note

The optimal strategy for the sample test case is shown as follows:

- Let member 1 carry member 4. As $w_1 > w_4$, member 1's speed remains unchanged, which is still 10.
- Let member 3 carry member 2. As $w_3 < w_2$, member 3's speed will decrease by $w_2 - w_3 = 2$ and becomes $10 - 2 = 8$.

- Member 5 shall move alone. His/Her speed is 9.

So the answer is 8.

Problem J. Building Company

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

You're the boss of a building company. At the beginning, there are g types of employees in the company, and different types of employees have different occupations. For the i-th type of employees, their occupation can be numbered as t_i and there are u_i employees in total.

There are n building projects in the market waiting to be undertaken. To undertake the i-th project, your company must meet m_i requirements. The j-th requirement requires that your company has at least $b_{i,j}$ employees whose occupation is $a_{i,j}$. After undertaking the project, your company will become more famous and will attract k_i types of employees to join your company. The occupation of the j-th type of employees is $c_{i,j}$ and there are $d_{i,j}$ employees in total.

You can undertake any number of projects in any order. Each project can be undertaken at most once. Calculate the maximum number of projects you can undertake.

Note that employees are not consumables. After undertaking a project the number of employees in your company won't decrease.

Input

There is only one test case in each test file.

The first line of the input first contains an integer g ($1 \le g \le 10^5$) indicating the number of types of employees in the company at the beginning. Then g pairs of integers $t_1, u_1, t_2, u_2, \cdots t_g, u_g$ follow ($1 \le t_i, u_i \le 10^9$), where t_i and u_i indicate that there are u_i employees whose occupation is t_i. It's guaranteed that for all $1 \le i < j \le g$ we have $t_i \ne t_j$.

The second line contains an integer n ($1 \le n \le 10^5$) indicating the number of projects waiting to be undertaken.

For the following $2n$ lines, each two lines describe a project.

The $(2i - 1)$-th line first contains an integer m_i ($0 \le m_i \le 10^5$) indicating the number of requirements to undertake the i-th project. Then m_i pairs of integers $a_{i,1}, b_{i,1}, a_{i,2}, b_{i,2}, \cdots, a_{i,m_i}, b_{i,m_i}$ follow ($1 \le a_{i,j}, b_{i,j} \le 10^9$) where $a_{i,j}$ and $b_{i,j}$ indicate that the company is required to have at least $b_{i,j}$ employees whose occupation is $a_{i,j}$. It's guaranteed that for all $1 \le x < y \le m_i$ we have $a_{i,x} \ne a_{i,y}$.

The $2i$-th line first contains an integer k_i ($0 \le k_i \le 10^5$) indicating the number of types of employees to join the company after undertaking the i-th project. Then k_i pairs of integers $c_{i,1}, d_{i,1}, c_{i,2}, d_{i,2}, \cdots, c_{i,k_i}, d_{i,k_i}$ follow ($1 \le c_{i,j}, d_{i,j} \le 10^9$) where $c_{i,j}$ and $d_{i,j}$ indicate that there are $d_{i,j}$ employees whose occupation is $c_{i,j}$ joining the company. It's guaranteed that for all $1 \le x < y \le k_i$ we have $c_{i,x} \ne c_{i,y}$.

It's guaranteed that neither the sum of m_i nor the sum of k_i will exceed 10^5.

Output

Output one line containing one integer indicating the maximum number of projects you can undertake.

Example

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

Note

We explain the sample test case as follows. Let (t, u) indicate u employees whose occupation is t.

First, undertake the 5-th project with no requirements. After undertaking the project, there are 2 employees, whose occupation is 3, joining the company. The company now have these employees: $\{(1, 2), (2, 1), (3, 2)\}$.

Next, undertake the 1-st project. After undertaking the project, no employee joins the company. The company now still have these employees: $\{(1, 2), (2, 1), (3, 2)\}$.

Next, undertake the 2-nd project. After undertaking the project, there are 2 employees, whose occupation is 3, and 1 employee, whose occupation is 2, joining the company. The company now have these employees: $\{(1, 2), (2, 2), (3, 4)\}$.

Next, undertake the 4-th project. After undertaking the project, there are 3 employees, whose occupation is 1, joining the company. The company now have these employees: $\{(1, 5), (2, 2), (3, 4)\}$.

As the company does not have 3 employees whose occupation is 2, we cannot undertake the 3-rd project.