

Problem A

Brace for Impact!

Time Limit: 2.0 seconds

The territory of an island country can be regarded as a strictly convex polygon C on the two-dimensional plane. The polygon C has N vertices $P_1, P_2, \dots, P_N$, where the coordinates of P_i are (x_i, y_i) . The vertices are given in counterclockwise order. Here, C denotes the closed polygonal region, including its boundary.

To prepare for tsunamis that may arrive without warning, the country has decided to build a tsunami shelter strictly inside its territory.

A tsunami's wavefront is modeled as a sufficiently long straight line that moves at unit speed without changing its shape or orientation. Let $\theta \in [0, 2\pi)$ denote the direction from which the tsunami arrives. The unit vector $(\cos \theta, \sin \theta)$ points from the shelter toward the sea from which the tsunami comes. Thus, the wavefront is perpendicular to this vector and moves in the direction $(-\cos \theta, -\sin \theta)$.

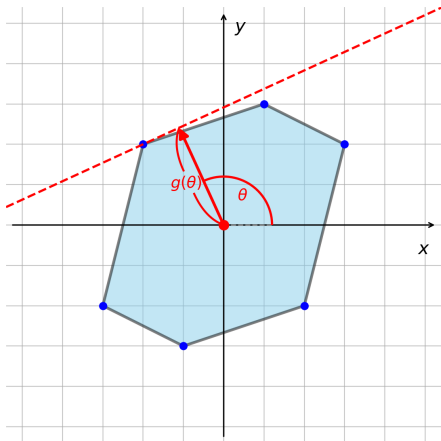
The instant the wavefront first touches the territory, a warning is issued throughout the country. Suppose that the shelter is located at (a, b) . Let $g(\theta)$ be the time from when the warning is issued until the wavefront reaches the shelter.

More precisely, $g(\theta)$ is the maximum real number c such that the line

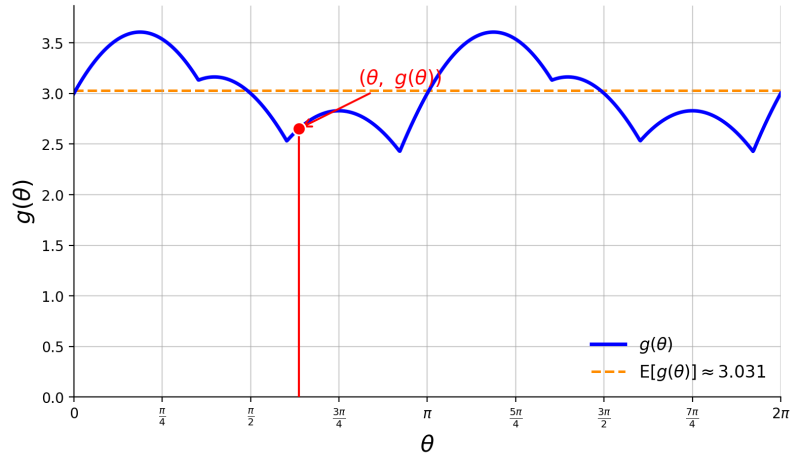
$$(x - a) \cos \theta + (y - b) \sin \theta = c$$

intersects C .

The following figures correspond to the first test case in the sample input, with the shelter placed at the origin (i.e., $(a, b) = (0, 0)$). They illustrate the wavefront and $g(\theta)$ for a particular arrival direction θ , and how $g(\theta)$ changes with the arrival direction. The red dashed line in the first figure is the position of the wavefront when it first touches the territory.



(a) $g(\theta)$ for one arrival direction



(b) The relation between the arrival direction and $g(\theta)$

Define the safety level $f(a, b)$ of a shelter at (a, b) as the expected value of $g(\theta)$ when θ is chosen uniformly at random from $[0, 2\pi)$.

Determine the maximum possible value of $f(a, b)$ when the shelter is placed strictly inside C .

Input

The input contains one or more test cases. The first line contains an integer T , the number of test cases ($1 \leq T \leq 10^4$). The descriptions of the test cases follow, each in the following format.

```

N
x1 y1
x2 y2
⋮
xN yN

```

The integer N is the number of vertices of C ($3 \leq N \leq 10^5$).

For each $i = 1, \dots, N$, the integers x_i and y_i are the coordinates of P_i ($|x_i|, |y_i| \leq 10^9$). The points $P_1, P_2, \dots, P_N$ are the vertices of the strictly convex polygon C , given in counterclockwise order. In other words, every interior angle of C is less than π .

The sum of N over all test cases does not exceed 2×10^5 .

Output

Output T lines. The i -th line should contain the maximum safety level for the i -th test case.

Your output will be considered correct if its absolute or relative error from the exact answer does not exceed 10^{-9} .

Sample Input	Sample Output
<pre> 2 6 -3 -2 -1 -3 2 -2 3 2 1 3 -2 2 4 6 3 7 7 4 2 5 0 </pre>	<pre> 3.030772067921022 2.443410850956194 </pre>

In the first test case, the origin lies strictly inside C . Placing the shelter at the origin gives a safety level of approximately 3.03077207, which is the maximum possible.

Problem B

Deque Bracket Optimization

Time Limit: 2.0 seconds

A string satisfying one of the following conditions is defined as a **correct parenthesis sequence**.

- The empty string.
- A string obtained by concatenating $($, A , and $)$ in this order for some correct parenthesis sequence A .
- A string obtained by concatenating A and B in this order, where A and B are nonempty correct parenthesis sequences.

Let n be a positive integer. For an integer sequence $w = (w_1, w_2, \dots, w_{2n})$ of length $2n$, define $f(w)$ as follows.

For a correct parenthesis sequence $s = s_1 s_2 \dots s_{2n}$ of length $2n$, define its score $w(s)$ with respect to w as the sum of w_i over all i ($1 \leq i \leq 2n$) such that s_i is $($. Let $f(w)$ be the maximum value of $w(s)$ over all correct parenthesis sequences s of length $2n$.

You are given a positive integer Q . Initially, the integer sequence W is empty.

Process Q queries in order. In the i -th query, you are given integers t_i, x_i, y_i . Here, t_i is one of $1, 2, 3$ and indicates the following:

- If $t_i = 1$: prepend x_i to W , and then prepend y_i to W .
- If $t_i = 2$: prepend x_i to W , and then append y_i to W .
- If $t_i = 3$: append x_i to W , and then append y_i to W .

For each query, find $f(W)$ after the query has been processed.

Input

The input consists of one or more test cases. The first line of the input contains an integer T ($1 \leq T \leq 10^5$), the number of test cases. Each of the T test cases is given in the following format:

$$\begin{array}{l} Q \\ t_1 \ x_1 \ y_1 \\ t_2 \ x_2 \ y_2 \\ \vdots \\ t_Q \ x_Q \ y_Q \end{array}$$

The integer Q ($1 \leq Q \leq 4 \times 10^5$) represents the number of queries.

For each integer i ($1 \leq i \leq Q$), the integers t_i, x_i, y_i represent the contents of the query. Here, t_i is one of $1, 2, 3$, and $|x_i|, |y_i| \leq 10^7$.

The sum of Q over all test cases does not exceed 4×10^5 .

Output

For each of the T test cases, print Q lines. The i -th line should contain $f(W)$ after W has been modified by the i -th query.

Sample Input

```
3
4
3 2 3
3 4 5
3 6 7
3 -1 -1
3
1 10 100
2 1 100000
3 10000 1000
1
2 -1234567 7654321
```

Sample Output

```
2
6
12
19
100
101
100101
-1234567
```

Problem C

Two Greedy Customers

Time Limit: 3.0 seconds

A shop sells N types of products numbered from 1 to N . The price of product i is C_i .

Two customers, Alice and Bob, will visit the shop separately. When Alice purchases product i , she gains satisfaction U_i . When Bob purchases it, he gains satisfaction V_i . Initially, Alice has A units of money, and Bob has B units of money.

Before they visit, you, a shop clerk, choose an ordering of the N product types. Each customer independently considers the product types one by one in the chosen order and makes purchases according to the following rules.

- The customer inspects the products one by one from left to right.
- If the price of the current product is not greater than the amount of money the customer has at that moment, the customer always purchases one item of that product and pays its price.
- Otherwise, the customer does not purchase that product.

Each product is available in sufficient quantity, and the purchases made by Alice and Bob do not affect each other.

The score of each customer is the sum of the satisfaction gained from the products purchased by that customer. Determine the maximum possible sum of the scores of Alice and Bob by arranging the products appropriately.

Input

The input consists of a single test case in the following format.

```
N A B
C1 U1 V1
C2 U2 V2
⋮
CN UN VN
```

The integer N is the number of product types ($1 \leq N \leq 200$). The integers A and B are the initial amounts of money held by Alice and Bob, respectively ($1 \leq A, B \leq 120$).

For each $i = 1, \dots, N$, the integer C_i is the price of product i ($1 \leq C_i \leq 120$). The integers U_i and V_i are the satisfaction gained by Alice and Bob, respectively, when they purchase product i ($1 \leq U_i, V_i \leq 10^9$).

Output

Output the maximum possible sum of the scores of Alice and Bob in one line.

Sample Input 1	Sample Output 1
2 6 8 3 10 1 5 8 10	21

Sample Input 2

```
2 1 1
1 5 1
1 1 10
```

Sample Output 2

```
11
```

Sample Input 3

```
3 5 6
4 1 100
3 100 1
2 10 10
```

Sample Output 3

```
220
```

If the products are arranged in the order 1, 2 in the first sample, Alice purchases only product 1 and gets a score of 10, while Bob purchases both products and gets a score of 11. The total score is 21, which is the maximum possible.

Problem D

Robot Damage

Time Limit: 3.0 seconds

A laboratory has a straight test track, which can be regarded as a number line, and a robot that moves along it. At the beginning of each experiment, the robot is stationary at the origin.

Two walls are placed on the track, one at a negative coordinate and the other at a positive coordinate. The robot cannot pass through the walls and can move only within the closed interval between them.

The robot has a button. When the button is pressed, the robot performs the following N phases in order.

- Suppose that the robot is at coordinate X at the beginning of phase i . It attempts to move along the number line toward coordinate $X + V_i$. It moves to the right if $V_i > 0$ and to the left if $V_i < 0$.
- If no wall blocks its path, the robot takes one minute to reach its destination. Its speed is zero at the beginning of the phase. It then accelerates and decelerates, reaching the destination at speed zero. Its speed is positive at every intermediate point of the movement.
- If a wall lies strictly between its current position and its destination, the robot collides with the wall during the movement and ends the phase at the wall.

After completing all N phases, the robot stops at its current position. If the button is pressed again, it performs the same N phases again, starting from that position.

Every time the robot collides with a wall at positive speed, it takes one point of damage. However, it takes no damage in either of the following cases.

- Its destination is exactly at a wall, so it reaches the wall at speed zero.
- It is already touching a wall at the beginning of a phase and attempts to move beyond that wall. In this case, it does not move.

In other words, the robot takes damage during a phase if and only if a wall lies strictly between its position at the beginning of the phase and its intended destination. The amount of damage does not depend on its speed at the collision, as long as the speed is positive.

You are given Q queries. The queries are independent. The i -th query describes the following experiment.

- Place the walls at coordinates $-W_i$ and W_i , and place the robot at the origin.
- Press the button a total of T_i times, waiting for the previous button press to finish before pressing it again. Do not move the robot between button presses.

Determine the total amount of damage taken by the robot during this experiment.

You are given t test cases. Solve each of them.

Input

The input contains one or more test cases. The first line contains an integer t , the number of test cases ($1 \leq t \leq 10^5$). The descriptions of the test cases follow, each in the following format.

```

N
V1 V2 ... VN
Q
W1 T1
W2 T2
⋮
WQ TQ

```

The integer N is the number of phases performed on each button press ($1 \leq N \leq 2 \times 10^5$). For each $i = 1, \dots, N$, the integer V_i represents the displacement attempted in phase i ($-10^9 \leq V_i \leq 10^9$, $V_i \neq 0$).

The integer Q is the number of queries ($1 \leq Q \leq 2 \times 10^5$). For each $i = 1, \dots, Q$, the integers W_i and T_i represent the distance from the origin to each wall and the number of button presses, respectively, in the i -th query ($1 \leq W_i \leq 10^{16}$, $1 \leq T_i \leq 10^{12}$).

The sum of N over all test cases does not exceed 4×10^5 . The sum of Q over all test cases does not exceed 4×10^5 .

Output

For each test case, output Q lines. The i -th line should contain the answer to the i -th query.

Sample Input	Sample Output
4	2
4	2
1 -8 6 -7	0
2	1
5 1	1
9 2	20000000000000
3	
1 -2 1	
1	
1 100	
2	
3 4	
2	
8 2	
8 10000	
2	
10000000000 -10000000000	
1	
10 100000000000000	

In the first test case, query 1 causes collisions at -5 in phases 2 and 4, for 2 points of damage. In query 2, both collisions at -9 occur during the second button press.

In the second test case, the robot never hits a wall at positive speed, so it takes no damage.

Problem E

Vibe Coding

Time Limit: 3.0 seconds

Shobon has decided to use a generative AI to help develop a software project. The project consists of N tasks. Each task i ($i = 1, 2, \dots, N$) has a predetermined **implementation workload** A_i and **complexity** B_i .

Shobon can proceed with these tasks in any order. However, he cannot work on more than one task at the same time.

Each task i ($i = 1, 2, \dots, N$) can be completed in one of the following two ways:

- **Implement the task manually.** Shobon writes the code himself, which takes A_i units of time.
- **Use the generative AI.** To make the AI generate correct code, Shobon explains the current task and all previously completed tasks to the AI. Let S be the sum of the complexities of all previously completed tasks. Preparing this explanation takes $S + B_i$ units of time. The AI then finishes the implementation instantly, so no additional implementation time is required. Thus, completing this task by using the AI takes a total of $S + B_i$ units of time.

Regardless of which method is used, once task i is completed, its complexity B_i is included in future values of S when using the generative AI.

Find the minimum total time required to complete all tasks.

Input

The input consists of three lines in the following format.

$$\begin{array}{l} N \\ A_1 \ A_2 \ \dots \ A_N \\ B_1 \ B_2 \ \dots \ B_N \end{array}$$

In the first line, the number of tasks, N , is given ($1 \leq N \leq 2 \times 10^5$). The second line contains N integers, $A_1, A_2, \dots, A_N$, each between 1 and 10^9 , inclusive, representing the implementation workloads of the N tasks. The third line contains N integers, $B_1, B_2, \dots, B_N$, each between 1 and 10^8 , inclusive, representing the complexities of the N tasks.

Output

Output in a line the minimum total time required to complete all tasks.

Sample Input 1

3
10 6 10
5 2 4

Sample Output 1

18

Sample Input 2

1
4
10

Sample Output 2

4

Sample Input 3

```
5
100 100 100 100 100
5 1 4 2 3
```

Sample Output 3

```
35
```

Sample Input 4

```
12
18 7 25 12 30 5 16 22 9 28 14 20
6 11 2 8 1 13 5 3 10 4 9 7
```

Sample Output 4

```
118
```

For Sample Input 1, consider using the generative AI for tasks 2 and 3 in this order, and then implementing task 1 manually.

- When task 2 is started, no task has been completed yet. This task takes $0 + B_2 = 2$ units of time.
- When task 3 is started, task 2 has been completed. This task takes $B_2 + B_3 = 2 + 4 = 6$ units of time.
- Finally, implementing task 1 manually takes $A_1 = 10$ units of time.

Thus, all tasks can be completed in $2 + 6 + 10 = 18$ units of time, which is the minimum possible.

Problem F

Tree + 1

Time Limit: 2.0 seconds

You are given a tree with N vertices numbered from 1 to N .

Your goal is to visit every vertex of this tree at least once by following the steps below.

1. Choose any two vertices and add an edge connecting them.
2. Choose any one vertex and visit it.
3. Repeat the following action any number of times.
 - Choose any one of the vertices adjacent by an edge to the vertex you are currently visiting, and visit that vertex. This action costs 1.

Find the minimum possible total cost required to achieve the goal.

Input

The input consists of one or more test cases. The first line of the input contains an integer t ($1 \leq t \leq 10^5$), the number of test cases. Each of the t test cases is given in the following format:

$$\begin{array}{l} N \\ u_1 \ v_1 \\ u_2 \ v_2 \\ \vdots \\ u_{N-1} \ v_{N-1} \end{array}$$

The integer N ($2 \leq N \leq 5 \times 10^5$) represents the number of vertices in the given tree.

For each integer i ($1 \leq i \leq N-1$), the integers u_i and v_i represent the vertex numbers of the endpoints of the i -th edge.

The given graph forms a tree.

The sum of N over all test cases does not exceed 5×10^5 .

Output

For each test case, output the minimum possible total cost required to achieve the goal.

Sample Input	Sample Output
3 6 1 2 2 3 3 4 2 5 5 6 6 6 3 1 4 5 1 3 1 2 3 10 1 5 4 8 9 1 3 5 3 2 7 3 1 6 10 3 1 4	5 5 12

For the first test case, after adding an edge connecting vertices 4 and 6, you can start at vertex 1 and visit the vertices in the order $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 5$, for a total cost of 5.

Problem G

Longest Path on DAG

Time Limit: 2.0 seconds

This is an interactive problem.

You are given a positive integer N and a sequence $S = (S_1, S_2, \dots, S_N)$ of length N whose elements are all either 0 or 1.

There is a directed bipartite graph G with N vertices numbered $1, 2, \dots, N$. Each vertex of G is colored: vertex v is white if $S_v = 0$ and black if $S_v = 1$. Every directed edge (u, v) in G satisfies both of the following conditions:

- $u < v$.
- Vertices u and v have different colors.

The judge has G hidden, and you are not explicitly given any information about its edges. Instead, you may send the following two types of queries to the judge, at most 15000 queries in total:

- **in u V** : Ask the judge about a black vertex u and a set V of zero or more white vertices. The judge answers whether there exists a vertex v in V such that the edge (v, u) exists.
- **out u V** : Ask the judge about a black vertex u and a set V of zero or more white vertices. The judge answers whether there exists a vertex v in V such that the edge (u, v) exists.

Output any one of the longest paths in G .

Note that the hidden graph G is fixed before the interaction and remains unchanged.

Interaction

First, an integer N ($2 \leq N \leq 500$) is given from standard input. The next line contains N integers $S_1, S_2, \dots, S_N$, each of which is 0 or 1, separated by spaces.

N
 $S_1 S_2 \dots S_N$

After reading the input, you may make at most 15000 **in queries** and **out queries** in total, as described below.

in query

First, choose a black vertex u and a set V of zero or more white vertices. Then, construct a string $T = T_1 T_2 \dots T_N$ of length N consisting only of 0s and 1s such that T_v is 1 if and only if $v \in V$. Print the query to standard output in the following format:

in u T

The judge determines whether there exists a vertex v in V such that the edge (v, u) exists. Let r be 1 if such a vertex exists and 0 otherwise. The value r is given from standard input in the following format:

r

out query

First, choose a black vertex u and a set V of zero or more white vertices. Then, construct a string $T = T_1T_2\ldots T_N$ of length N consisting only of 0s and 1s such that T_v is 1 if and only if $v \in V$. Print the query to standard output in the following format:

out u T

The judge determines whether there exists a vertex v in V such that the edge (u, v) exists. Let r be 1 if such a vertex exists and 0 otherwise. The value r is given from standard input in the following format:

r

Once you have determined a longest path, let $v_1, v_2, \dots, v_k$ be the vertex numbers in the order in which they appear on the path, and print your answer to standard output in the following format:

! k v_1 v_2 $\dots$ v_k

Terminate your program immediately after outputting the answer.

Notes on Interactive Judging

If your output does not conform to the specified format, or if your program terminates without outputting an answer, the verdict is undefined. Terminate your program immediately after outputting the answer. Otherwise, the verdict is undefined.

Whenever you output something, end it with a newline and flush standard output.

If you make more than 15000 queries, the verdict is undefined. Note that outputting the answer does not count as a query.

Sample Interaction

Read	Write
5	
1 0 0 1 0	out 1 01101
1	in 4 01100
1	in 4 01000
0	out 4 01001
1	! 4 1 3 4 5

In this sample interaction, the hidden graph has the edge set $E(G) = \{(1, 2), (1, 3), (1, 5), (3, 4), (4, 5)\}$. This information is not given to the program.

Problem H

Warp

Time Limit: 2.0 seconds

There is a circle with circumference L , where L is an even integer. Positions on the circle are measured counterclockwise and represented by real numbers in the range $[0, L)$.

There are N points on this circle. The i -th point is located at position A_i . The positions are not necessarily distinct.

Initially, you can choose any position on the circle as your starting position.

After that, you can move along the circle either clockwise or counterclockwise at a speed of 1 unit per second. You can change your direction at any time, without consuming any time.

Furthermore, you can use a **magic warp** at any time, as many times as you like. A magic warp takes no time and moves you to the exact opposite of your current position on the circle. More precisely, if you are at position x ($0 \leq x < L$), a warp takes 0 seconds, and your position after the warp becomes

$$\begin{cases} x + L/2 & (0 \leq x < L/2), \\ x - L/2 & (L/2 \leq x < L). \end{cases}$$

A point is considered visited whenever your position coincides with its position. Find the minimum time, in seconds, required to visit all N points at least once.

Input

The input consists of a single test case of the following format.

```
N L
A1 A2 ... AN
```

The integer N represents the number of points on the circle ($1 \leq N \leq 2 \times 10^5$). The even integer L represents the circumference of the circle ($2 \leq L \leq 10^9$). Each integer A_i satisfies $0 \leq A_i < L$ and indicates the location of the i -th point on the circle ($i = 1, 2, \dots, N$).

Output

Output the minimum time required to visit all N points. It can be proved that the answer is an integer.

Sample Input 1

```
4 10
0 2 8 9
```

Sample Output 1

```
3
```

Sample Input 2

```
1 10
9
```

Sample Output 2

```
0
```

Sample Input 3

```
7 100
39 57 57 1 99 70 37
```

Sample Output 3

```
33
```

In Sample Input 1, it is possible to visit all 4 points in 3 seconds as follows.

- First, you start at position 2, visiting the second point.
- Move to position 3, taking 1 second.
- Use a magic warp and move to position 8, taking 0 seconds. You also visit the third point.
- Move to position 9, taking 1 second. You also visit the fourth point.
- Move to position 0, taking 1 second. You also visit the first point.

Problem I

Buttons

Time Limit: 4.0 seconds

Consider a sequence $X = (X_1, X_2, \dots, X_N)$ of length N . Initially, all elements of X are zero. You are also given a sequence $P = (P_1, P_2, \dots, P_N)$ of N nonnegative integers.

There are M buttons. Pressing the i -th button once increases X_j by 1 for every integer j satisfying $L_i \leq j \leq R_i$. Each button may be pressed any number of times, including zero.

The buttons must be pressed so that, for every $i = 1, 2, \dots, N$, the final value of X_i is at most K .

Find the maximum possible value of

$$\sum_{i=1}^N P_i X_i$$

among all valid ways of pressing the buttons.

Input

The input contains one or more test cases. The first line of the input contains an integer t ($1 \leq t \leq 10^5$), which is the number of test cases. The descriptions of the t test cases follow, each in the following format.

```
N M K
P1 P2 ... PN
L1 R1
⋮
LM RM
```

The first line of each test case contains three integers N , M , and K , representing the length of the sequence X , the number of buttons, and the upper bound for each element of X , respectively. They satisfy $1 \leq N, M, K \leq 2 \times 10^5$.

The second line contains N integers $P_1, P_2, \dots, P_N$. For each i ($1 \leq i \leq N$), the integer P_i represents the weight of X_i and satisfies $0 \leq P_i \leq 10^7$.

Each of the following M lines contains two integers L_j and R_j ($1 \leq j \leq M$). They satisfy $1 \leq L_j \leq R_j \leq N$ and represent the interval affected by the j -th button.

The sum of N , M , and K over all test cases does not exceed 6×10^5 each.

Output

Output t lines. For each test case, output the maximum possible value of $\sum_{i=1}^N P_i X_i$ among all valid ways of pressing the buttons, in a separate line.

Sample Input	Sample Output
3 4 3 2 3 1 4 2 1 2 2 3 4 4 1 3 5 7 1 1 1 1 1 1 12 7 1 5 0 8 2 7 1 6 3 9 4 0 10 1 3 2 5 4 6 6 8 9 12 3 7 8 10	14 35 50

In the first test case, pressing the second and third buttons twice each results in $X = (0, 2, 2, 2)$. The resulting sum is then $3 \times 0 + 1 \times 2 + 4 \times 2 + 2 \times 2 = 14$, which is the maximum possible value.

Problem J

Take Many ABA

Time Limit: 3.0 seconds

For a string T consisting of A and B, define $f(T)$ as the maximum number of times the following operation can be performed.

Operation: Remove ABA that appears as a (not necessarily contiguous) subsequence of T , and close up the resulting gaps.

You are given a string S consisting of A, B, and ?. Find the sum of $f(T)$ over all strings T obtained by replacing each ? in S with either A or B, modulo 998244353.

Input

The input is given in the following format:

S

S is a string consisting of A, B, and ?, and its length is between 3 and 1000, inclusive.

Output

Print the answer.

Sample Input 1

A?B?

Sample Output 1

2

Sample Input 2

?B?A?B?

Sample Output 2

15

Sample Input 3

????????????????

Sample Output 3

4595728

The possible strings T and their corresponding values of $f(T)$ for the first sample are as follows:

- $T = \text{AABA}$: $f(T) = 1$.
- $T = \text{AABB}$: $f(T) = 0$.
- $T = \text{ABBA}$: $f(T) = 1$.
- $T = \text{ABBB}$: $f(T) = 0$.

Therefore, the answer is $1 + 0 + 1 + 0 = 2$.

Problem K

Empty Intersection

Time Limit: 6.0 seconds

You are given two integer sequences $A = (A_1, \dots, A_N)$ and $B = (B_1, \dots, B_N)$, and two sequences of positive integers $V = (V_1, \dots, V_N)$ and $W = (W_1, \dots, W_N)$. All four sequences have length N .

Choose two (possibly empty) intervals of indices, $[L, R)$ for A and $[U, D)$ for B . Here, L, R, U, D are integers satisfying $1 \leq L \leq R \leq N + 1$ and $1 \leq U \leq D \leq N + 1$.

Let X and Y be the sets of values occurring in these intervals, respectively:

$$X = \{A_i \mid L \leq i < R\},$$

$$Y = \{B_j \mid U \leq j < D\}.$$

The chosen intervals must satisfy $X \cap Y = \emptyset$. In other words, no value may occur in both intervals.

If $L = R$, the interval of A is empty and $X = \emptyset$. Likewise, if $U = D$, the interval of B is empty and $Y = \emptyset$.

The score of the two chosen intervals is

$$\sum_{i=L}^{R-1} V_i + \sum_{j=U}^{D-1} W_j.$$

Find the maximum possible score among all valid choices.

Input

The input contains one or more test cases. The first line of the input contains an integer T ($1 \leq T \leq 10^4$), which is the number of test cases. The descriptions of the T test cases follow, each in the format below.

```

N
A1 A2 ... AN
B1 B2 ... BN
V1 V2 ... VN
W1 W2 ... WN

```

The first line contains a single integer N , the common length of the four sequences ($1 \leq N \leq 2 \times 10^5$).

The following four lines contain N integers each and describe the sequences A , B , V , and W , respectively.

The integers A_i and B_i satisfy $1 \leq A_i, B_i \leq N$ for every i ($1 \leq i \leq N$).

The integers V_i and W_i satisfy $1 \leq V_i, W_i \leq 10^9$ for every i ($1 \leq i \leq N$).

The sum of N over all test cases does not exceed 6×10^5 .

Output

Output T lines. The i -th line should contain the maximum possible score for the i -th test case.

Sample Input	Sample Output
4	13
3	15
1 2 3	33
3 2 1	11
3 4 5	
6 2 1	
3	
1 1 1	
1 1 1	
2 3 4	
5 5 5	
4	
1 2 1 3	
4 2 4 3	
5 6 5 1	
4 10 4 10	
5	
3 1 2 4 3	
1 1 2 2 4	
1 3 3 2 1	
2 2 3 1 1	

In the first test case, we can choose the interval $[1, 3)$ of A and the interval $[1, 2)$ of B . More precisely, we can set $L = 1$, $R = 3$, $U = 1$, and $D = 2$. Their sets of values are $\{1, 2\}$ and $\{3\}$, respectively, and they are disjoint. The score is

$$V_1 + V_2 + W_1 = 3 + 4 + 6 = 13,$$

which is the maximum possible score.

In the second test case, we can choose an empty interval of A and the entire interval $[1, 4)$ of B . More precisely, $L = 2$, $R = 2$, $U = 1$, and $D = 4$.

Problem L

Edge Exchanges

Time Limit: 2.0 seconds

You are given two trees, A and B , each with N vertices. Both trees have the same labeled vertex set $\{1, 2, \dots, N\}$.

For each i ($1 \leq i \leq N - 1$), the i -th edge of A connects vertices u_i and v_i . For each j ($1 \leq j \leq N - 1$), the j -th edge of B connects vertices x_j and y_j .

For each $i = 1, 2, \dots, N - 1$, solve the following problem.

Find an integer j with $1 \leq j \leq N - 1$ such that the following exchange leaves both A and B as trees:

- Remove the edge connecting u_i and v_i from A , and remove the edge connecting x_j and y_j from B .
- Then, add an edge connecting x_j and y_j to A , and add an edge connecting u_i and v_i to B .

Each exchange is considered independently, starting from the original trees A and B .

It can be proved that at least one such integer j exists for every i under the given constraints.

Input

The input consists of a single test case in the following format.

```
N
u1 v1
u2 v2
⋮
u_{N-1} v_{N-1}
x1 y1
x2 y2
⋮
x_{N-1} y_{N-1}
```

The integer N is the number of vertices of both A and B ($2 \leq N \leq 2 \times 10^5$).

The following $N - 1$ lines describe the edges of A . For each $i = 1, \dots, N - 1$, the integers u_i and v_i ($1 \leq u_i, v_i \leq N$) mean that the i -th edge connects vertices u_i and v_i . It is guaranteed that A is a tree.

The next $N - 1$ lines describe the edges of B . For each $j = 1, \dots, N - 1$, the integers x_j and y_j ($1 \leq x_j, y_j \leq N$) mean that the j -th edge connects vertices x_j and y_j . It is guaranteed that B is a tree.

Output

Output $N - 1$ lines. For each $i = 1, 2, \dots, N - 1$, the i -th line should contain an integer j satisfying the condition in the problem statement for the i -th edge of A .

If there are multiple possible answers, you may output any of them.

Sample Input	Sample Output
4 1 3 2 4 3 4 1 2 2 3 2 4	1 3 2

In the first line of the sample output, $j = 1$ is chosen for $i = 1$. Exchanging the edge $(1, 3)$ of A with the edge $(1, 2)$ of B leaves both A and B as trees.

For $i = 2$, both the edge of A and the selected edge of B are both $(2, 4)$, so exchanging them does not change either tree. Other valid outputs are also possible.