TASK	IZBORI	OTOCI	PLAHTE
input	standard input		
output	standard output		
time limit	1 second	5 seconds	2 seconds
memory limit	128 MB		
points	**200**	**200**	**200**
	600		

It is election time. V voters attend the election, each casting their vote for one of N political parties. M officials will be elected into the parliament.

The conversion from votes to parliament seats is done using the D'Hondt method with a 5% threshold. More precisely, suppose that the parties are numbered 1 through N and that they receive V_1, V_2, ..., V_N votes. Parliament seats are allocated as follows:

1. All parties that receive strictly less than 5% of V votes are **erased** from the list of parties.
2. The parliament is initially empty i.e. every party has zero seats allocated.
3. For each party P, the quotient $Q_P=V_P/(S_P+1)$ is calculated, where V_P is the total number of votes received by party P, and S_P is the number of seats already allocated to party P.
4. The party with the largest quotient Q_P is allocated one seat. If multiple parties have the same largest quotient, the lower numbered party wins the seat.
5. Repeat steps 3 and 4 until the parliament is full.

The votes are being counted and only part of the V votes has been tallied. It is known how many votes each party has received so far.

Write a program that calculates for each party, among all possible outcomes of the election after all V votes are counted, the largest and smallest number of seats the party wins.

INPUT

The first line contains the integers V, N and M ($1 \leq V \leq 10{,}000{,}000$, $1 \leq N \leq 100$, $1 \leq M \leq 200$), the numbers of votes, parties and seats in the parliament.

The second line contains N integers – how many votes (of those that have been counted) each party got. The sum of these numbers will be at most V.

OUTPUT

On the first line output N integers separated by spaces – the largest number of seats each party can win.

On the second line output N integers separated by spaces – the smallest number of seats each party can win.

SCORING

For each test case, the two subtasks (two lines of output) are scored independently.

Solving the first subtask correctly is worth 20% of points.

Solving the second subtask correctly is worth 80% of points. It is necessary to output exactly N integers on the first line (even if they are completely wrong) for the second subtask to be graded.

EXAMPLES

input 20 4 5 4 3 6 1 **output** 3 3 3 2 1 0 1 0	**input** 100 3 5 30 20 10 **output** 4 3 3 1 1 0

In the first example, 14 votes have been tallied and 6 are yet to be counted. To illustrate one possible outcome, suppose that the first party receives 2 of those 6 votes, the second none, the third 1 vote and the fourth 3 votes. The parties' totals are 6, 3, 7 and 4 votes. All parties exceeded the 5% threshold. Seats are allocated as follows:

1. The quotients are initially 6/1, 3/1, 7/1 and 4/1; the largest is 7/1 so party 3 wins a seat.
2. The quotients are 6/1, 3/1, 7/2 and 4/1; the largest is 6/1 so party 1 wins a seat.
3. The quotients are 6/2, 3/1, 7/2 and 4/1; the largest is 4/1 so party 4 wins a seat.
4. The quotients are 6/2, 3/1, 7/2 and 4/2; the largest is 7/2 so party 3 wins a seat.
5. The quotients are 6/2, 3/1, 7/3 and 4/2; parties 1 and 2 are tied with quotients 6/2 and 3/1, but party 1 is lower numbered so it wins the last seat.

In this outcome, the numbers of seats won by the parties are 2, 0, 2 and 1. Since it is possible for the second party not to win any seats, the second number on the second line of output is zero.

Some time ago Mirko founded a new tourist agency named "Dreams of Ice". The agency purchased N icy islands near the South Pole and now offers excursions. Especially popular are the emperor penguins, which can be found in large numbers on the islands.

Mirko's agency has become a huge hit; so big that it is no longer cost-effective to use boats for the excursions. The agency will **build bridges** between islands and transport tourists by buses. Mirko wants to introduce a computer program to manage the bridge building process so that fewer mistakes are made.

The islands are numbered 1 through N. No two islands are initially connected by bridges. The initial number of penguins on each island is known. That number may change, but will always be between 0 and 1000 (inclusive).

Your program must handle the following three types of commands:

- "bridge A B" – an offer was received to build a bridge between islands A and B (A and B will be different). To limit costs, your program must accept the offer **only if there isn't already a way** to get from one island to the other using previously built bridges. If the offer is accepted, the program should output "yes", after which the bridge is built. If the offer is rejected, the program should output "no".
- "penguins A X" – the penguins on island A have been recounted and there are now X of them. This is an informative command and your program does not need to respond.
- "excursion A B" – a group of tourists wants an excursion from island A to island B. If the excursion is possible (it is possible to get from island A to B), the program should output the **total number of penguins** the tourists would see on the excursion (including islands A and B). Otherwise, your program should output "impossible".

Important note: your program must output responses to commands "bridge" and "excursion" immediately after they are received. The server program will not send the next command until your program responds to the previous one.

Another important note: for the server program to be able to read your program's responses, your program must flush the standard output after every response it outputs.

- In C++, use the command `cout << flush;`
- In C, use `fflush(stdout);`
- In pascal, use `flush(output);`

INPUT

The first line contains the integer N ($1 \leq N \leq 30000$), the number of islands.

The second line contains N integers between 0 and 1000, the initial number of penguins on each of the islands.

The third line contains an integer Q ($1 \leq Q \leq 300000$), the number of commands.

Q commands follow, each on its own line. As noted above, after receiving a command "bridge" or "excursion", your program will not receive another command until it has responded to the previous one.

OUTPUT

Output the responses to commands "bridge" and "excursion", each on its own line.

SCORING

In test cases worth 50% of points, the command "penguins" will not appear. In these test cases N will be odd, while in all other cases N will be even.

EXAMPLES

input

```
5
4 2 4 5 6
10
excursion 1 1
excursion 1 2
bridge 1 2
excursion 1 2
bridge 3 4
bridge 3 5
excursion 4 5
bridge 1 3
excursion 2 4
excursion 2 5
```

output

```
4
impossible
yes
6
yes
yes
15
yes
15
16
```

input

```
6
1 2 3 4 5 6
10
bridge 1 2
bridge 2 3
bridge 4 5
excursion 1 3
excursion 1 5
bridge 3 4
excursion 1 5
penguins 3 10
excursion 1 3
bridge 1 5
```

output

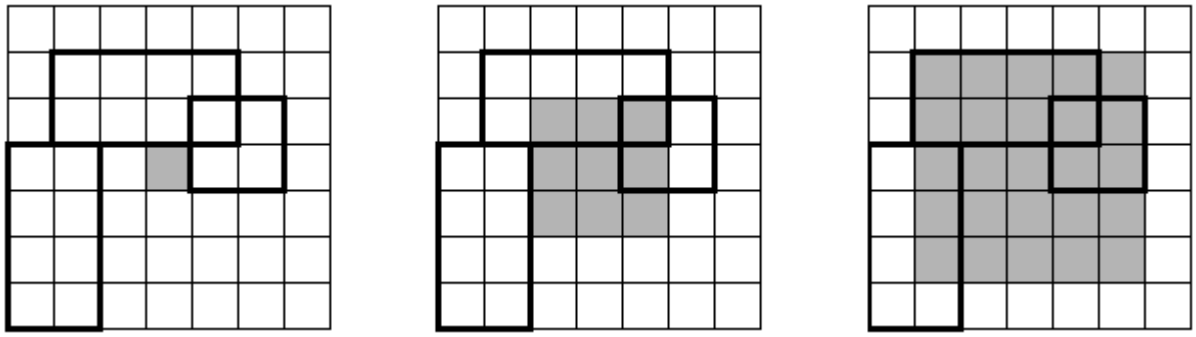
```
yes
yes
yes
6
impossible
yes
15
13
no
```

Mirko washed his sheets and is on his way to hang them to dry in front of his house. However, strong winds have pulled the clothesline out of the ground so Mirko temporarily laid out the sheets on the grass.

The grass field can be modeled by an infinite square grid, where every **unit square** is represented by a pair of coordinates. Sheets are rectangles in the grid with sides parallel to the coordinate axes. Sheets may overlap.

In an effort to put his clothesline back up, Mirko slammed a pole into the ground at coordinates (0, 0). An entirely unexpected turn of events followed. Oil sprung from the ground and the shock caused Mirko to faint. While Mirko lay unconscious, the oil is spreading, staining his sheets.

Time is measured from the moment the oil starts spreading – at time zero only square (0, 0) is covered in oil. The oil is spreading at a speed of **one square per second in all eight directions**, as shown in the figure below. When oil enters a square, it stains that square of fabric on **all** sheets covering the square.



The grass field from the first example after zero, one and two seconds.

Write a program that, given M points in time, calculates the **total area of stained fabric** on all sheets for each of the given time points.

INPUT

The first line contains an integer N ($1 \leq N \leq 100000$), the number of sheets.

Each of the following N lines contains four integers x_1, y_1, x_2 and y_2 ($-1\,000\,000 \leq x_1 \leq x_2 \leq 1\,000\,000$), ($-1\,000\,000 \leq y_1 \leq y_2 \leq 1\,000\,000$). The coordinates ($x_1$, y_1) and (x_2, y_2) represent diagonally opposite corners of a sheet (again, these coordinates are unit squares, not points on the plane). None of the sheets will cover the square (0, 0).

The next line contains an integer M ($1 \leq M \leq 100000$), the number of points in time.

The next line contains M integers between 0 and 1 000 000, the time points. They will be given in strictly ascending order.

OUTPUT

For each of the time points, output on a separate line the total area of stained fabric on all sheets, in the order the time points are given in the input.

EXAMPLES

input	input	input
3 -2 1 1 2 1 0 2 1 -3 -3 -2 0 2 1 2	4 5 1 8 4 -8 1 -5 4 -10 2 10 3 6 0 8 10 6 1 2 3 4 7 9	1 1 1 1000000 1000000 3 100 10000 1000000
output	**output**	**output**
5 15	0 5 14 18 70 100	10000 100000000 1000000000000