TWGSS
OI Team

Cooperation Theory III

SCHEDULE

Date 18 February 2025

Time 14:55 - 18:55 | 4 hours

RULES & REMINDERS

- Teams are ordered by the number of **Solved problems**, followed by the **total penalty time**.
- All test cases in a problem must be passed to be considered **Solved (Accepted)**.
- A submission with the verdict **Wrong Answer**, **Time Limit Exceeded** or **Runtime Error** is considered **Rejected**.
- The **penalty time** for a solved problem is the number of minutes past the beginning of the contest that it is solved, plus the number of **Rejected** submissions multiplied by 20 minutes.
- The **total penalty time** is the sum of all solved problems' **penalty time**.
- You are recommended to attempt the problems using C++20. It is not guaranteed that the problems are solvable using other languages.
- Unless otherwise specified, your output will be automatically fixed as follows:
  - Trailing spaces in each line will be removed.
  - An end-of-line character will be added to the end of the output if not present.
- 64-bit integers may be required in some problems, use `long long` in C++ if needed.
- The problems are NOT necessarily arranged in ascending order of difficulty.
- All stories are hypothetically written. In case of coincidences, they are coincidences.

PROBLEMS

	Problem		Limit	
			Time	Memory
TP251	**A**	Anyday in the Sun	1s	256MB
	B	Best Challenge	1s	256MB
	C	Cauchy the Rock!	1s	256MB
	D	Decomposition Decomposition	1s	256MB
	E	Eratosthenes' Revenge	1s	4MB
	F	Finding Socks	1s	256MB
	G	Ghost Post	1s	256MB
	H	Heung Shing Railway	1s	256MB
	I	I agree with you	1s	256MB
	J	Jacket Production	1s	256MB
	K	King's Battle	1s	16MB
	L	Losing Heroine	1s	256MB

TP251 Anyday in the Sun

Limit: 1s | 256MB

A Sunny Day. An Exciting Day. The training has been backed to normal, you are so excited! Therefore, you decide to propose an imaginary problem based on that.

Imagine Haruhi (ハルヒ) is living in the JOI Kingdom *(not IOI Kingdom anymore)*. She decides to do a "tiny" experiment on up to the 10^6 measuring cylinders she owned. Initially, some water is poured into each measuring cylinder. The amount of water poured to different measuring cylinders may varies.

You are asked to report the total amount of water in all of the measuring cylinder to her at different points in time. Meanwhile, Haruhi may decide to add some amount of water to some of the test tubes as she wishes.

Obviously, it is impossible to keep track of all of the measuring cylinders. Therefore, you decide to cheat while still meeting Haruhi's requirement. You have prepared an extremely tall measuring cylinder, and you can measure the amount of water dried up at different moments.

With this cheat, you believe that you can simply implement a program to handle this automatically. Can you do so gracefully without angering Haruhi?

INPUT

The first line consists of two integers N and Q, representing the number of measuring cylinders and the number of events respectively.

The second line consists of N integers, the i^{th} integer A_i represents the initial water level of Haruhi's i^{th} measuring cylinder.

The j^{th} line of the next Q lines first consists of a character C_j, representing the type of the j^{th} event. If $C_j =$ `A`, two integers X_j and V_j follows, meaning V_j units of water are added to Haruhi's X_j^{th} measuring cylinder. If $C_j =$ `H`, an integer D_j follows, meaning the water level decreases by D_j units in the extremely tall measuring cylinder.

CONSTRAINTS

$1 \leq X_j \leq N \leq 10^6$

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

$0 \leq A_i, V_j, D_j \leq 10^9$

$C_j =$ `A` or `H`

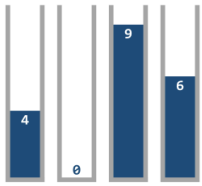
There exists at least one type `H` event

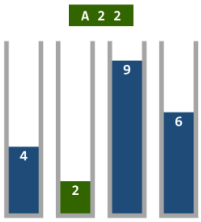
OUTPUT

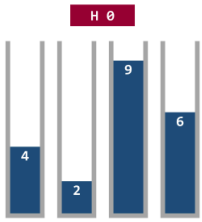
After each type `H` event, output the total units of water in all of Haruhi's measuring cylinders.

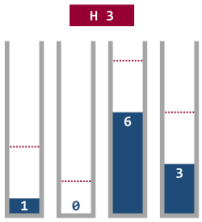
SAMPLE TESTS

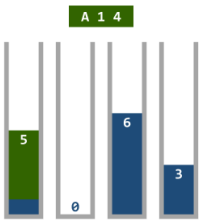
	Input	Output
1	4 6 4 0 9 6 A 2 2 H 0 H 3 A 1 4 H 5 H 3	21 10 1 0

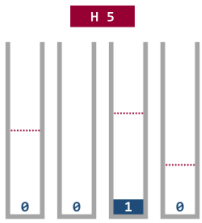


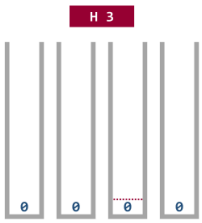












TP251 Best Challenge

Limit
1s | 256MB

"The person who finish the challenge first will get 10^9 dollars!"

Earlier, you received an invitation from Mr. Best to participate in his latest challenge! Upon arriving at the venue, you notice some massive blocks with numbers painted on them. In specific, there are four types of blocks with 1, 2, 3 and 4 printed on them respectively. There exists 4 **blocks of each type**, totalling up to 16 blocks.

"You have to arrange the blocks into a 4×4 grid. But there's a catch – numbers on each of the rows and columns must sum up to its target value!" More specifically, the sum of numbers on the i^{th} ($1 \leq i \leq 4$) row and column must be R_i and C_i respectively. It is guaranteed that solution exists for the given values.

Losing in the *Seat Game*™ a month ago, you can't wait to be the first one to finish Mr. Best's challenge and get the money you deserved.

INPUT

The first line consists of four integers R_1, R_2, R_3 and R_4, representing the targeted row sum of the 1^{st}, 2^{nd}, 3^{rd} and 4^{th} row respectively.

The second line consists of four integers C_1, C_2, C_3 and C_4, representing the targeted column sum of the 1^{st}, 2^{nd}, 3^{rd} and 4^{th} column respectively.

CONSTRAINTS

$4 \leq R_i, C_i \leq 16$

OUTPUT

Output 4 lines, each line should consist of 4 integers, each separated by a space, representing the solved grid.

If there exist multiple solutions, you may output any of them.

SAMPLE TESTS

	Input	Output
1	6 11 9 14 8 12 12 8	1 1 2 2 3 4 2 2 1 3 4 1 3 4 4 3

TP251 Cauchy the Rock!

Limit: 1s | 256MB

Cauchy the Rock is driving a car! He wants to go from his home to his university as quick as possible.

There are different terrains affecting the speed of the car. Specifically, there are N vertical axes, cutting the plane into different terrains. The x-coordinates of the vertical axes are $A_1, A_2, \ldots, A_N$ respectively. The speed of the car is specified as follows:

- For every point on the left of the axis $x = A_1$, the speed of the car is S_0 units per second.
- For $1 \leq i \leq N-1$, at every point between the axes $x = A_i$ and $x = A_{i+1}$, the speed of the car is S_i units per second.
- For every point on the right of the axis $x = A_N$, the speed of the car is S_N units per second.

Cauchy's home is at (X_H, Y_H) and he wants to travel to his university (X_U, Y_U). Can you find the minimum time needed to travel from his home to his university?

INPUT

The first line consists of an integer N, representing the number of vertical axes.

The second line consists of N integers, the i^{th} integer A_i represents the x-coordinate of the i^{th} vertical axis. Note that this line is empty if $N = 0$.

The third line consists of $N+1$ integers $S_0, S_1, S_2, \ldots, S_N$, the speed of the car in different strips of terrains.

The fourth line consists of two integers X_H and Y_H, describing the location of Cauchy's home.

The fifth line consists of two integers X_U and Y_U, describing the location of Cauchy's university.

CONSTRAINTS

$0 \leq N \leq 50000$

$0 \leq A_1 < A_2 < \cdots < A_N \leq 10^6$

$1 \leq S_i \leq 10^6$ for $0 \leq i \leq N$

$0 \leq X_H, Y_H, X_U, Y_U \leq 10^6$

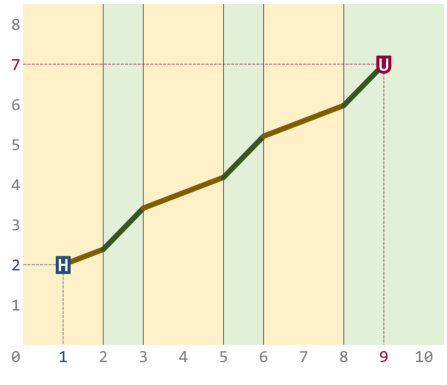
OUTPUT

Output a floating-point number on the first and only line, the minimum number of seconds needed for Cauchy to go from his home to his university.

Your answer will be accepted if its absolute or relative error is less than 10^{-6}.

SAMPLE TESTS

	Input	Output
1	5 2 3 5 6 8 1 2 1 2 1 2 1 2 9 7	7.505759110498



TP251 Decomposition Decomposition

Limit: 1s | 256MB

In the wonderful universe of *Squarespace*, there are N square tiles. Suddenly, you receive a mysterious offer, asking you to split the tiles to **minimum** number of groups, such that each group can form a square with integer side length. In other words, each group should consist of a^2 tiles, where a is a integer. The side lengths can be repeated.

What is that mysterious offer for? A discount code for *Squarespace*! Although you don't need that at all, you still decide to try solving this problem for the solve count!

INPUT

The first line consists of an integer N, representing the number of tiles.

CONSTRAINTS

$1 \leq N \leq 10^5$

OUTPUT

On the first line, output a single integer K, the number of groups formed.

On the second line, output K integers, each separated by a space. The i^{th} integer A_i should be the side length of the i^{th} square formed.

You can output the side lengths in any order. If there are multiple solutions, you may output any of them.

SAMPLE TESTS

	Input	Output
1	5	2 1 2

5 tiles need to be split into 2 groups minimally, one 1×1 and one 2×2.

	Input	Output
2	2022	3 43 13 2

TP251 Eratosthenes' Revenge

Limit: 1s | 4MB

Note the unusual memory limit.

You are currently teaching a lesson on Eratosthenes' Sieve (愛氏篩)! Unfortunately, you are not good at Greek, so you don't actually know how to pronounce his name. Therefore, you decide to refer him as Mr. E instead.

Upon saying that, Eratosthenes suddenly appears out of nowhere. *"How dare you don't know how to pronounce my name!"* exclaimed Eratosthenes *(translated from Greek, of course)*. Eratosthenes decides to copyright his sieve so you can no longer use it in any situation!

"Fine, let me invent my own sieve!" you said.

To challenge you, Eratosthenes writes down a problem on the blackboard. He decides an integer N between 2 and 10^9 in secret, then for all the numbers $N, N+1, \ldots, N+29$, he write down `1` if the number is prime and `0` if the number of not a prime (of course, using his sieve). You are asked to find the original number N.

Unfortunately, you find this problem isn't really do-able, as multiple numbers easily result to the same sequence written on the blackboard (e.g. 20250222 and 2022)! Therefore, you decide to ask Eratosthenes for more hints. Eratosthenes recognise this as an issue also, so he decides to give you more information – for each non-prime integer, its smallest prime factor is also given to you.

Can you prove your sieving ability to Eratosthenes to counteract his copyright claim and cause *a series of unfortunate events for Mr. E (E先生連環不幸事件)*, or admit that Eratosthenes' Sieve™ is really useful and apologise to Eratosthenes?

INPUT

The first and only line consists of thirty integers, the sequence written on the blackboard. The i^{th} ($0 \le i \le 29$) integer is 1 if $N+i$ is a prime, and is its smallest prime factor if $N+i$ is not a prime.

OUTPUT

Output an integer on the first and only line, **any possible** N that generates the sequence written on the blackboard.

Your N must still satisfy $2 \le N \le 10^9$. It is guaranteed that such N always exists for all test cases.

SAMPLE TESTS

	Input	Output
1	1 1 2 1 2 1 2 3 2 1 2 1 2 3 2 1 2 1 2 3 2 1 2 5 2 3 2 1 2 1	2
2	2 7 2 3 2 1 2 1 2 3 2 1229 2 5 2 3 2 1 2 11 2 3 2 5 2 17 2 3 2 7	20250222
3	2 7 2 3 2 1 2 1 2 3 2 19 2 5 2 3 2 1 2 13 2 3 2 5 2 23 2 3 2 7	2022

TP251 Finding Socks

Limit: 1s | 256MB

The website CodeImpact recently held a Think-Box Round, and the prize is a pair of branded socks! Daniel likes that pair of socks very much.

One day, Daniel finds out that he can't find a matching pair of socks! It would be embarrassing for him to wear a different socks on left and right foot *(as of his daily outfit is not embarrassing enough)*. Therefore, he decides to do some investigation to find out a matching pair of socks.

There are in total $N+1$ socks of N different types. There are only one sock in each type, except his branded socks, which appears in pair. *(Now you know why he likes that branded socks very much, because it's the only pair thay he can wear outside.)*

It is way too hard to identify the pair of socks manually. Therefore, Daniel has a genius plan, using the *Socks Pairing Detector (SPD)*. By throwing some socks into it and pressing the "Run" button, the SPD will detect whether or not there exists a pair of matching socks in it.

The socks are now arbitrary labelled as 1 to $N+1$. Can you find out the matching pair of socks using the SPD? However, the SPD consumes much electricity when it runs. *(Some says as much as a particle accelarator!)* Therefore, the number of runs is limited to 100. Otherwise, the hostel will receive an expensive electric bill and terminate Daniel's residency!

IMPLEMENTATION

You should implement the function `finding_socks(int N)`:

- N: The number of different kinds of socks.
- This function will be called exactly once.
- You may call the function `run(int[] A)` at most 100 times.
- You should call the function `answer(int x, int y)` exactly once before you terminates.

You are allowed to call the following functions:

`bool run(int[] A)`

- A: The set of indices of socks you want to query.
- This function returns `true` if there exist two socks of the specified indices that are of the same kind. Otherwise, this function returns `false`.
- There should not be duplicated indices in A.
- You may call this function at most 100 times.

`void answer(int x, int y)`

- Give your guess as the sock indexed x and the sock indexed y are of the same kind.
- The given indices x and y should be different.
- You should call this function exactly once, and terminates immediately afterwards.

CONSTRAINTS

$1 \leq N \leq 100$

TEMPLATE

```
#include "socks.h"

void finding_socks(int N) {
    std::vector<int> A;
    A.push_back(1);
    A.push_back(2);
    bool res = run(A);
    if (res) {
        answer(1, 2);
    } else {
        answer(2, 3);
    }
}
```

SAMPLE GRADER

The sample grader reads the input in the following format:
Line 1: N
Line 2: A_1 A_2 ... A_{N+1}

If your solution is correct, the output will be in the following format:
Line 1: `OK`
Line 2: K, where K is the number of runs

If your solution is wrong, the output will be in the following format depending on the reason:
Line 1: `WA`
Line 2: Simple description of the reason for the wrong answer

Compilation Command: `bash compile_cpp.sh`
Run Command: `./socks`

SAMPLE RUN

Judge	User
finding_socks(5)	
	run({1, 2, 3})
false	
	run({2, 3, 6})
true	
	run({2, 6})
true	
	answer(2, 6)

SAMPLE TESTS

	Input	Output
1	5 1 2 4 5 3 2	OK 3

This test corresponds to the Sample Run.

TP251 Ghost Post

Limit
1s | 256MB

"Coach Wong, do you think Leader Lo should serve for another term? (黃教練，你覺得盧隊長連任好不好啊？)"
"Of course! (好啊！)"
"Supporting Leader Lo so early, won't it give the impression that he was appointed? (現在那麼早你們就說支持盧隊長，會不會給人印象就是內定了、欽點了？)"
"There is no appointing. It's still according to OI Team's tradition, according to the preference of past Leaders… (沒有任何的意思。還是按照隊伍的傳統、按照前隊長的意願……)"
"Coach Wong, but… (黃教練，但是……)"
"After all, you guys are still too young. You understand what I mean, right? You guys have one strength, being faster than anyone else to give up in contests. But the code you write is too simple, sometimes naïve! Do you understand? (我覺得你們啊，畢竟還 too young，明白這意思吧。你們有一個好，無論參加甚麼比賽，你們比其他的同學啊，放棄得還快。但是呢，寫來寫去的程式啊，都 too simple、sometimes naïve！懂了沒有啊？識得唔識得啊？)"

Getting criticisms for appointing Leaders in OI Team, Coach Wong decide to reform the system for deciding posts in the Team, using nothing but… fate!

There are N OI Team members and posts respectively, such that each member would take up exactly one post. There is a paper of width N and height $2N+2$, with $(1,0)$ representing the top-left cell and $(N,2N+1)$ representing the bottom-right cell.

To begin, Coach Wong would write the names of all N OI Team members on the top row of the paper, with the name of i^{th} member at $(i,0)$. Then, N OI Team posts are written on the bottom row of the paper, with the i^{th} role at $(i,2N+1)$. Finally, N vertical lines are drawn connecting the names and roles, with the i^{th} vertical line spans from $(i,1)$ to $(i,2N)$.

After doing the above setup, Coach Wong would "randomly" draw some horizontal line segments onto the paper. There are some requirements on the horizontal lines drawn:

- Any horizontal line segments are drawn should be length 1, connecting two vertical lines. In other words, it should span from (r,c) to $(r,c+1)$, where $1 \le r \le 2N$ and $1 \le c \le N-1$.
- There should not be multiple horizontal line segments overlapping with each other, i.e. connecting the same pair of points.
- A point cannot simultaneously connect to its left and its right by horizontal line segments. In other words, if there already exists a horizontal line connecting (r,c) to $(r,c+1)$, it is invalid to have another horizontal line connecting $(r,c-1)$ to (r,c) or $(r,c+1)$ to $(r,c+2)$.

Once finished, the game of "Ghost Leg" is played to determine the post of each OI Team member. The rules are as follows:

- Start from the position where the name is found.
- Repeatedly move downwards. Upon reaching any horizontal line, move across the horizontal line.
- The post at the final position is the post of that OI Team member.

"Of course our decision is also important! (當然我們的決定權也是很重要的！)" Coach Wong already has a post preference for each of the N members! He wants the i^{th} member to get the P_i^{th} post. However, Coach Wong does not know how to "randomly" draw such a Ghost Leg diagram to achieve all of his post preferences!

"2025: Make It Happen! (2025，一定要得！)" Can you help Coach Wong to arrange such a Ghost Leg diagram?

INPUT

The first line consists of an integer N, representing the number of OI Team members and posts.

The second line consists of N integers, the i^{th} integer A_i represents the post preference of the i^{th} member.

CONSTRAINTS

$1 \leq N \leq 2000$

Each of $1, 2 \ldots, N$ appears exactly once in $A_1, A_2, \ldots, A_N$

OUTPUT

On the first line, output an integer K, the number of horizontal line segments to be drawn.

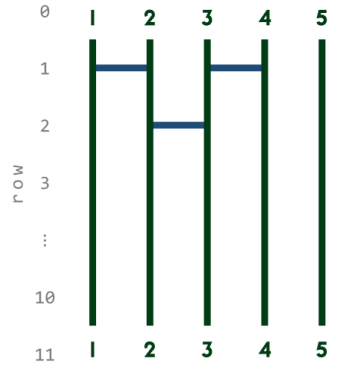
On the i^{th} line of the next K lines, output two integers R_i and C_i, meaning the i^{th} horizontal line segment is drawn between point (R_i, C_i) and point $(R_i, C_i + 1)$. The requirements of drawing horizontal lines must be satisfied.

You may output the line segments in any order. If there are multiple solutions, you may output anyone of them.

It is guaranteed that solution exists for all test cases.

SAMPLE TESTS

	Input	Output
1	5 3 1 4 2 5	3 1 1 2 2 1 3



TP251 Heung Shing Railway

Limit
1s | 256MB

You are currently visiting Heung Shing! The layout of Heung Shing can be seen as a rectangular grid of N rows and M columns, with the top left cell being $(1, 1)$ and the bottom right cell being (N, M).

The only railway company Heung Shing Railway (HR) in Heung Shing has three systems, HR All, HR Odd and HR Even, offering services in different ways:

- **HR All**: It provides service at all cells. It allows you to go **horizontally or vertically**, that is, from (r, c) to any of $(r + 1, c)$, $(r - 1, c)$, $(r, c + 1)$ or $(r, c - 1)$. The cost per distance is $\$All$.
- **HR Odd**: It only provides service at all **odd** cells (that is, all cells (r, c) such that $r + c$ is odd). It allows you to go **diagonally**, that is, from (r, c) to any of $(r + 1, c + 1)$, $(r - 1, c + 1)$, $(r + 1, c - 1)$ or $(r - 1, c - 1)$. The cost per distance is $\$Odd$.
- **HR Even**: It only provides service at all **even** cells (that is, all cells (r, c) such that $r + c$ is even). It allows you to go **diagonally**, that is, from (r, c) to any of $(r + 1, c + 1)$, $(r - 1, c + 1)$, $(r + 1, c - 1)$ or $(r - 1, c - 1)$. The cost per distance is $\$Even$.

You have just finished your lunch at HS Cafe at (R_0, C_0). You have already planned Q locations to visit in a particular order, with the i^{th} location at (R_i, C_i). Already spending $\$10^{18}$ in Heung Shing, you don't have much money left! Therefore you want to minimise the travelling cost to fulfil your travel plan.

"I can just walk!" Technically yes, but no, you must take the railways. As a *legendary coding grandmaster tourist*, you decide to write a program to find the minimum cost for each of the Q railway trips.

INPUT

The first line consists of two integers N and M, representing the number of rows and columns in Heung Shing respectively.

The second line consists of three integers All, Odd and $Even$, representing the cost to travel per distance on HR All, HR Odd and HR Even respectively.

The third line consists of two integers R_0 and C_0, describing your initial location.

The fourth line consists of an integer Q, representing the number of locations in your travel plan.

The i^{th} line of the next Q lines consists of two integers R_i and C_i, describing the i^{th} location in your travel plan.

CONSTRAINTS

$1 \leq N, M \leq 10^8$

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

$1 \leq All, Odd, Even \leq 10^9$

$1 \leq R_i \leq N$ for $0 \leq i \leq Q$

$1 \leq C_i \leq M$ for $0 \leq i \leq Q$

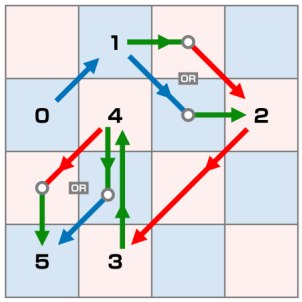
OUTPUT

Output Q lines.

On the i^{th} line, output an integer, the minimum cost for travelling from the previous location (R_{i-1}, C_{i-1}) to the next location (R_i, C_i).

SAMPLE TESTS

	Input	Output
1	4 4 1 1 1 2 1 5 1 2 2 4 4 2 2 2 4 1	1 2 2 2 2

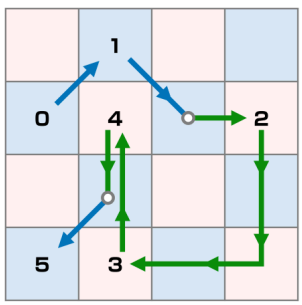


HR ALL
$ 1

HR ODD
$ 1

HR EVEN
$ 1

2	4 4 1 2 3 2 1 5 1 2 2 4 4 2 2 2 4 1	2 3 4 2 3



HR ALL
$ 1

HR ODD
$ 2

HR EVEN
$ 3

TP251 I agree with you

Limit
1s | 256MB

You ask Duncan whether he agrees with you. He responded you, but not as *directly*. His message can be summarised by a string S:

- The string starts with `I`.
- Then, some (possibly none) `do not` follows.
- After that, either `agree` or `disagree` follows.
- Finally, the string ends with `with you.`.

Each part of S is separated by a space.

Using your English common sense, can you determine whether or not Duncan agrees with you?

INPUT

The first and only line contains a string S, Duncan's message.

CONSTRAINTS

$1 \leq \text{length of } S \leq 1000$

OUTPUT

Output `agree` or `disagree` on the first and only line, depending on the meaning of S.

SAMPLE TESTS

	Input	Output
1	I agree with you.	agree
2	I do not disagree with you.	agree
3	I do not do not do not agree with you.	disagree

TP251 Jacket Production

Limit: 1s | 256MB

The team is once again designing the team jacket. However, this time is not a physical jacket – but a virtual jacket!

The jacket already has a working design *by no other than the extremely talented ImPoster Maker*. As a virtual jacket, it is in the form of an ASCII Art.

Getting the *"Team Jacket Name Inputter"* role in the Ghost Leg post distribution, your job is, as the name suggested, to put members' name onto the jacket. There is a field for putting names on it, denoted by a contiguous `X`s. Your work is to put the name in the input field, while centring it. If there are multiple centres, put the name slightly to the left. You may refer to Sample Tests for clarification.

"I'll get the job done!" Somehow having 10^9 members joining the team this year, you decide to write a program to help you putting names on the virtual jackets.

INPUT

The first line consists of a string S, representing the member's name to be put on the jacket.

The second line consists of an integer H, representing the height of the team jacket design.

The i^{th} line of the next H line consists of a string D_i, representing the i^{th} line of the jacket design.

CONSTRAINTS

S, D_i consists of printable ASCII characters (including spaces)

$1 \leq \text{length of } S, D_i \leq 50$

$1 \leq H \leq 50$

There exists exactly one segment of contiguously appearing `X`s in all D_i

OUTPUT

Output `Invalid` on the first and only line if the name S is too long for the input field (contiguous `X`s in the jacket design).

Otherwise, output H lines, the jacket design with name replacing the input field on it, following the requirements.

SAMPLE TESTS

	Input	Output

1	Felix
	6
	  /^^(_____)^^\
	 / /| TWGSS |\ \
	/ / | O I | \ \
	O | | O
	  | XXXXX |
	  ------- 	
		/^^(_____)^^\
		 / /| TWGSS |\ \
		/ / | O I | \ \
		O | | O
		  | Felix |
		  ------- 	

2	Felix
	6
	  /^^(_____)^^\
	 / /| TWGSS |\ \
	/ / | O I | \ \
	O | | O
	  | XXXX |
	  ------- 	
		Invalid

3	Unless we can't
	6
	  /^^(_____)^^\
	 / /| |\ \
	/ / |We'll get the code working| \ \
	O | | O
	  |XXXXXXXXXXXXXXXXXXXXXXXX|
	  ------- 	
		/^^(_____)^^\
		 / /| |\ \
		/ / |We'll get the code working| \ \
		O | | O
		  | Unless we can't |
		  ------- 	

4	Lo 2 | XXXTak Court | | lou6 dak1 wai4 |	| Lo Tak Court | | lou6 dak1 wai4 |
5	X 3 xxxxxxxxx xxxXXXxxx xxxxxxxxx	xxxxxxxxx xxx X xxx xxxxxxxxx

TP251 King's Battle

Limit
1s | 16MB

Please note the unusual memory limit.

Once upon a time, there were two mighty kingdoms, of which their kings were King Charles and King Mason. It was a peaceful world until they both fall in love with a lady *(the name was censored and lost, but later historians speculated her name starts with A and have 5 characters)*. To decide who could bring lady A home, they agreed to a duel with Q matches. King Charles had N holy knights while King Mason had N high priests. In order to make the matches fair, they agreed to the following rules:

- In match j, some L_j and R_j are chosen. The L_j to R_j (inclusive) holy knights of King Charles and the L_j to R_j (inclusive) high priests join the match.
- Then, each pair of holy knight and high priest fight against each other (a total of $(R_j - L_j + 1)^2$ battles). Either wins the battle if his ability is higher than his opponent. If both are with the same ability, then the battle is tied.

It was such a epic duel that the details were remembered by every witness and was spread out by the wandering minstrel. Unfortunately, the verdicts of the matches are lost in the history.

Your role being a historian is to recover the verdict in each of the Q matches.

INPUT

The first line consists of two integers N and Q, representing the number of holy knights or high priests and the number of matches respectively.

The second line consists of N integers, the i^{th} integer C_i represents the ability of the i^{th} King Charles' holy knight.

The third line consists of N integers, the i^{th} integer M_i represents the ability of the i^{th} King Mason's high priest.

The j^{th} line of the following Q lines consists of two integers L_j and R_j, describing the j^{th} match.

CONSTRAINTS

$1 \leq N, Q \leq 2 \times 10^4$

$1 \leq C_i, M_i \leq 10^5$

$1 \leq L_j \leq R_j \leq N$

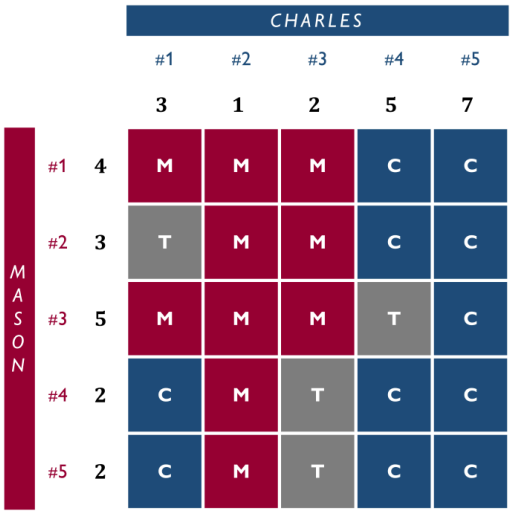
OUTPUT

Output Q lines.

On the j^{th} line, output three integers, each separated by a space, the number of battles King Charles won, the number of battles King Mason won and the number of tied battles respectively.

SAMPLE TESTS

	Input	Output
1	5 3 3 1 2 5 7 4 3 5 2 2 4 4 2 5 2 3	1 0 0 7 6 3 0 4 0



TP251 Losing Heroine

Limit
1s | 256MB

Yanami Anna is the losing heroine (負けヒロイン)! Now, she is in a game where she needs to compete against N opponents successively. The i^{th} opponent has the strength of A_i. Anna needs to face them from the 1^{st} one to the N^{th} one, following the order.

For each match, she would win if her strength is higher than the opponent. Otherwise, if her strength is less than or equal to opponent's strength, she loses the match *(well, she is the losing heroine)*. As the old saying *"failure makes you stronger"*, her strength will then increase by the opponent's strength after the loss.

Plots are always unpredictable, Q possible situations may happen. For the j^{th} situation, Anna would have the initial strength X_j. Can you calculate how many times Anna loses in each of the situations, such that the unpredictable plots can become somewhat predictable?

INPUT

The first line consists of two integers N and Q, representing the number of opponents and the number of situations respectively.

The second line consists of N integers, the i^{th} integer A_i represents the strength of the i^{th} opponents that Anna is going to face.

The j^{th} line of the next Q lines consists of an integer X_j, representing the initial strength of Anna in the j^{th} situation.

CONSTRAINTS

$1 \leq N, Q \leq 3 \times 10^5$

$1 \leq A_i, X_j \leq 10^9$

OUTPUT

Output Q lines.

On the j^{th} line, output an integer, the number of times Anna loses in the j^{th} situation.

SAMPLE TESTS

Input	Output
9 5 3 1 4 1 5 9 2 6 5 1 2 3 5 10	3 2 2 1 0

1

In the 1^{st} situation, Anna has the initial strength 1. In the matches,

- She loses to the 1^{st} opponent. Her strength then becomes $1 + 3 = 4$.
- She wins against the 2^{nd} opponent.
- She loses to the 3^{rd} opponent. Her strength then becomes $4 + 4 = 8$.
- She wins against the 4^{th} and 5^{th} opponent.
- She loses to the 6^{th} opponent. Her strength then becomes $8 + 9 = 17$.
- She wins against the 7^{th} to 9^{th} opponent.