Hong Kong Olympiad in Informatics 2017/18
Senior Group

Task Overview

ID	Name	Time Limit	Memory Limit	Subtasks
S181	Odd is Odd	1.000 s	256 MB	14 + 16 + 13 + 24 + 33
S182	Tournament	1.000 s	256 MB	7 + 13 + 20 + 25 + 10 + 25
S183	Desktop Icons	1.000 s	256 MB	12 + 15 + 16 + 21 + 12 + 24
S184	Bogo Translate	1.000 s	256 MB	18 + 25 + 31 + 26

Notice:
Unless otherwise specified, inputs and outputs shall follow the format below:

- One space between a number and another number or character in the same line.
- No space between characters in the same line.
- Each string shall be placed in its own separate line.
- Outputs will be automatically fixed as follows: Trailing spaces in each line will be removed and an end-of-line character will be added to the end of the output if not present. All other format errors will not be fixed.

C++ programmers should be aware that using C++ streams (`cin` / `cout`) may lead to I/O bottlenecks and substantially lower performance.

For some problems 64-bit integers may be required. In Pascal it is `int64`. In C/C++ it is `long long` and its token for `scanf` /`printf` is `%lld`.

All tasks are divided into subtasks. You need to pass all test cases in a subtask to get points.

S181 - ODD IS ODD

Time Limit: 1.000 s / Memory Limit: 256 MB

Alice feels that it is so odd to have dinner alone. Therefore, she invites her $N-1$ friends to her home and eat together. In other words, N people are going to eat together.

Before Alice's friends come, Alice cooked a lot of nuggets and distributed the nuggets on N plates. However, as Alice is not good at division, she just arbitrarily put some nuggets on each plate instead of distributing the nuggets evenly. More precisely, A_i nuggets are placed on the i^{th} plate.

After distributing the nuggets, Alice puts the plates on a round table in clockwise order. The $(i+1)^{th}$ plate is to the left of the i^{th} plate for $1 \leq i < N$ and the 1^{st} plate is to the left of the N^{th} plate.

However, after placing the plates, Alice notices that some plates contain an odd number of nuggets. She thinks that it is odd to eat an odd number of nuggets, so she decides to perform the following operations until every plate contains an even number of nuggets.

Each time, Alice may perform one of the two operations:

- Alice picks one nugget from plate i and puts it on the plate to the left of plate i. Alice needs C seconds to perform this operation.
- Alice picks one nugget from plate i and puts it on the plate to the right of plate i. Alice needs D seconds to perform this operation.

Now, Alice wants to know what is the minimum time required to make every plate contains an even number of nuggets. Note that it is acceptable for Alice to have a plate with no nuggets, as long as there does not exist a plate containing an odd number of nuggets.

INPUT

The first line contains three integers, N, C, D.
The second line contains N integers, $A_1, A_2, \ldots, A_N$.

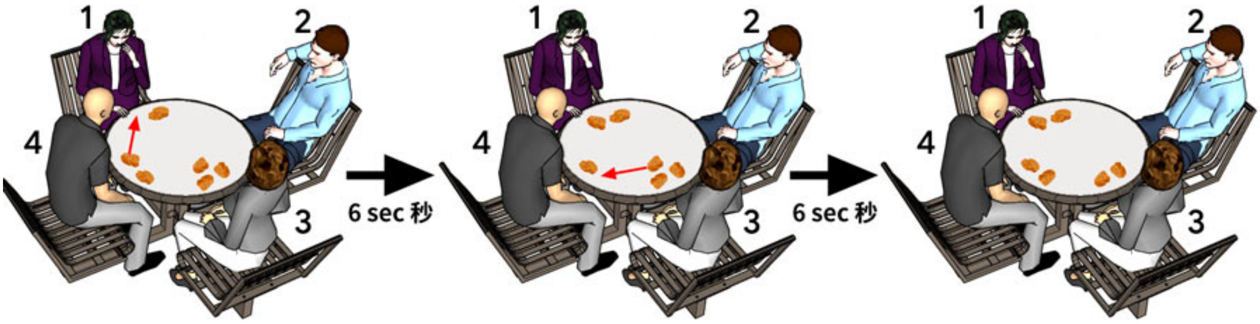
OUTPUT

Output a single integer in a line denoting the minimum time (in seconds) to make every plate contains an even number of nuggets. If it is impossible to do so, output `-1`.

SAMPLE TESTS

	Input	Output
1	4 6 9 1 0 3 2	12

One of the optimal solutions:



	Input	Output
2	4 3 2 1 1 1 1	4
3	3 3 2 2 3 4	-1
4	6 2 3 1 4 1 5 9 2	6

SUBTASKS

For all cases:
$2 \le N \le 10^5$
$0 \le A_i \le 18$
$1 \le C, D \le 10^4$

	Points	Constraints
1	14	$N = 3$
2	16	Exactly two integers in A_i are odd.
3	13	All A_i are odd.
4	24	$2 \le N \le 1000$
5	33	No additional constraints.

S182 - TOURNAMENT

Time Limit: 1.000 s / Memory Limit: 256 MB

Three teams, named Alpha, Beta and Gamma, participate in the annual Fantasy Football (FF) tournament. The tournament format is single round-robin, i.e. each pair of teams play exactly one match against one another.

In a match between team X and team Y, its result is represented by "X x - y Y", or equivalently, "Y y - x X", where x and y are non-negative integers. It means that team X scores x goals and concedes y goals, while team Y scores y goals and concedes x goals, in that match. Since FF is not exactly football, the values of x and y can be unexpectedly large.

If $x > y$, team X is considered to have won the match and gains W points, while team Y is considered to have lost the match and gains no points.
If $x = y$, both teams are considered to have drawn the match, and each of them gains D points.
If $x < y$, team X is considered to have lost the match and gains no points, while team Y is considered to have won the match and gains W points.

Now, for each team, the total points, the total number of goals scored and the total number of goals conceded are given, but the match results are lost. Your task is to find a list of match results that is consistent with the given data.

INPUT

The first line contains two integers, W and D, the number of points awarded for winning and drawing a match, respectively.

The second line contains three integers, P_A, S_A and C_A, the total points, the total number of goals scored and the total number of goals conceded of team Alpha, respectively.

The third line contains three integers, P_B, S_B and C_B, the total points, the total number of goals scored and the total number of goals conceded of team Beta, respectively.

The forth line contains three integers, P_G, S_G and C_G, the total points, the total number of goals scored and the total number of goals conceded of team Gamma, respectively.

OUTPUT

If there exists a list of match results that is consistent with the given data, output three lines. Each line should contain the result of one match, in the form of "X x - y Y" where x and y are the numbers of goals scored by team X and team Y in that match, respectively.

Otherwise, output `Impossible`.

If there are more than one such list, output any of them. Also, the results can be outputted in any order.

SAMPLE TESTS

	Input	Output
1	3 1 6 2 0 1 0 1 1 0 1	Alpha 1 - 0 Beta Alpha 1 - 0 Gamma Beta 0 - 0 Gamma
2	3 1 6 3 0 1 0 1 1 0 1	Impossible

SUBTASKS

For all cases:
$0 \le D \le W \le 5$
$0 \le P_A, P_B, P_G \le 2W$
$0 \le S_A, S_B, S_G, C_A, C_B, C_G \le 10^9$

	Points	Constraints
1	7	$S_A = S_B = S_G = C_A = C_B = C_G = 0$
2	13	$W > 2D$ $P_A = P_B = P_G = 2D$ $S_A = C_A$ $S_B = C_B$ $S_G = C_G$
3	20	$0 \le S_A, S_B, S_G, C_A, C_B, C_G \le 20$
4	25	$0 \le S_A, S_B, S_G, C_A, C_B, C_G \le 10^6$
5	10	$W = 0$
6	25	No additional constraints

S183 - DESKTOP ICONS

Time Limit: 1.000 s / Memory Limit: 256 MB

Alice likes trying out different operating systems. The appearances of the user interfaces on different operating systems are different. For example, the "Start menu" on Microsoft Windows is absent on macOS, and that is why some Microsoft Windows users get confused when switching to the macOS. Nonetheless, different user interfaces do have some common features. On Microsoft Windows, you typically see a desktop showing icons such as the "Recycle Bin", "My Computer" and various files; on macOS, you also see a desktop showing icons such as connected disks and various files. The wallpapers on both desktops may be customized as well.

The desktop feature is very common among user interfaces. The user interface on the operating system that Alice is now trying out also shows a desktop on which various icons can be placed. The operating system is called BWOS, named after the user interface being black and white only. Since the icons on the desktop and the wallpaper are also in black and white, sometimes an icon may be invisible. For example, when a user places a black desktop icon in a black region of the wallpaper, the icon will be invisible.

To avoid invisible icons, users may choose an appropriate wallpaper and place the icons in appropriate locations. In a good desktop icon placement, important icons should be made visible and invisible icons should be unimportant.

More specifically, the desktop is organized as a rectangular grid of R rows and C columns. Each grid cell is a unit square slot in which at most one desktop icon could be placed. The desktop slot located at the j^{th} column in the i^{th} row is denoted as (i, j), so the desktop slot at the top-left corner is denoted as $(1, 1)$ and the the desktop slot at the bottom-right corner is denoted as (R, C).

There are K wallpapers which can be chosen, numbered from 1 to K. Each wallpaper consists of R rows and C columns of tiles, each of which is a unit square colored either black or white. The denotation of the wallpaper tiles is the same as the denotation of the desktop slots, and each wallpaper tile corresponds to one desktop slot. Therefore when a wallpaper is chosen, the background color of the desktop slot (i, j) will be the color of the wallpaper tile (i, j) of that wallpaper.

There are N desktop icons, each of which is colored either black or white and is placed in an empty slot on the desktop. No two desktop icons can be placed in the same desktop slot. A desktop icon is visible if its color is different from the background color of the desktop slot it is placed, and it is invisible otherwise. For example, when a white desktop icon is placed in a desktop slot with black background, the icon will be visible; if it is placed in a desktop slot with white background, it will be invisible.

Unless N equals $R \times C$, there will be at least one empty desktop slot on the desktop. Users may rearrange the desktop icons by repeatedly dragging a desktop icon to an empty desktop slot, but not to a desktop slot already occupied by another desktop icon because that would cause BWOS to crash immediately. Only one desktop icon will be moved in one drag. If N equals $R \times C$, all desktop slots are occupied and the desktop icons cannot be rearranged.

Each desktop icon is assigned an importance value. Alice wants to choose a wallpaper and rearrange the desktop icons to maximize the sum of importance values of all visible desktop icons. She does not care about which wallpaper is used, but she does not want to drag for more than 6400 times. Can you help her?

INPUT

The first line contains three integers N, R and C.
The i^{th} of the next N lines contains four information r_i, c_i, d_i and v_i. The information r_i, c_i and v_i are integers and the information d_i is the character `B` (stands for black) or `W` (stands for white). It means that the color and importance value of the desktop icon located at desktop slot (r_i, c_i) are d_i and v_i respectively.
It is guaranteed that $1 \leq r_i \leq R$, $1 \leq c_i \leq C$, and no two desktop icons occupy the same desktop slot.

The next line contains an integer K.
Then comes K blocks of input, the i^{th} block corresponds to wallpaper i. Each of these blocks consists of R lines each contains a string of C characters. Each of these characters is either `B` (stands for black) or `W` (stands for white). The j^{th} character on the i^{th} of these R lines indicates the color of the wallpaper tile (i, j).

OUTPUT

On the first line output three integers S, W and M, which are the maximum sum of importance values of visible desktop icons, the number of the chosen wallpaper and the number of drags performed respectively. Note that $1 \leq W \leq K$ and $0 \leq M \leq 6400$.
Then output M lines. The i^{th} of these M lines contains four integers $r1_i$, $c1_i$, $r2_i$ and $c2_i$, meaning that in the i^{th} drag the desktop icon located at desktop slot $(r1_i,\ c1_i)$ is moved to desktop slot $(r2_i,\ c2_i)$. Dragging to the original location (i.e., $r1_i = r2_i$ and $c1_i = c2_i$) is allowed.

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

SCORING

Within a subtask:

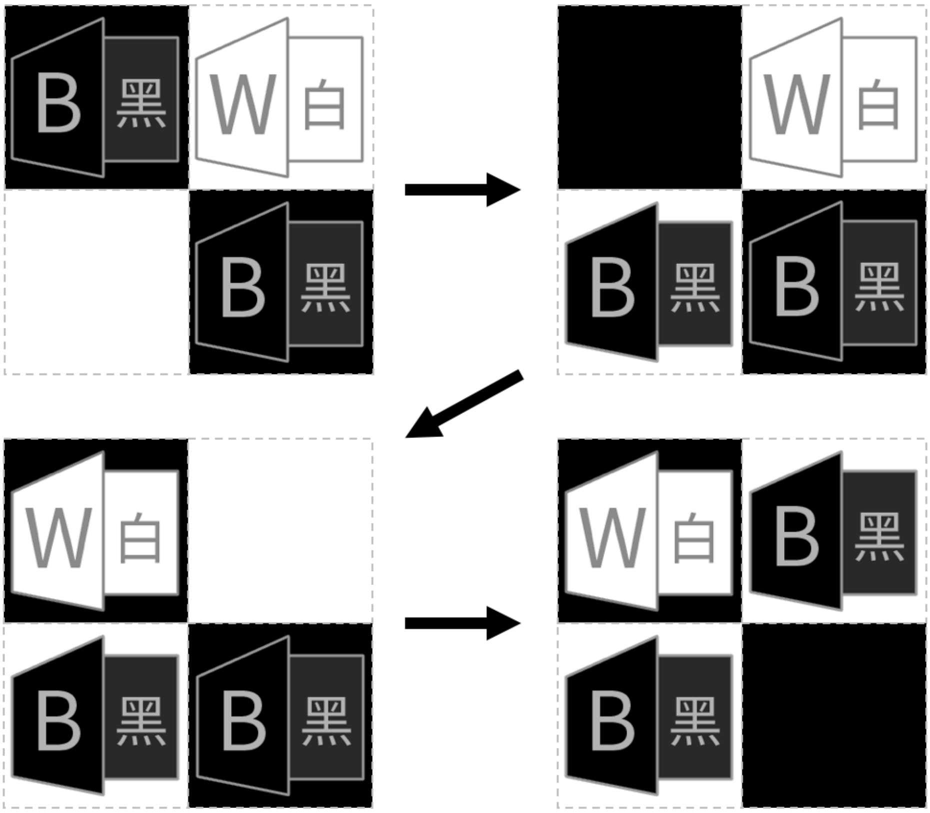
- If for each and every test case, your program outputs the correct maximum sum of important values of visible desktop icons, correct wallpaper number and correct dragging operations, you score 100% in the subtask.
- Otherwise if for each and every test case, your program outputs the correct maximum sum of important values of visible desktop icons, any wallpaper number and any dragging operations that would not crash BWOS in correct format with $1 \leq W \leq K$ and $0 \leq M \leq 6400$, you score 40% in the subtask.
- Otherwise, you lose all points in the subtask.

SAMPLE TESTS

	Input	Output
1	1 2 2 1 1 W 10 1 WW WW	0 1 0
2	1 2 2 1 1 W 10 2 WW WW WW BW	10 2 1 1 1 2 1

3		
	3 2 2 1 1 B 10 1 2 W 10 2 2 B 10 1 BW WB	30 1 3 1 1 2 1 1 2 1 1 2 2 1 2

The figure below illustrates the solution:



4		
	3 2 2 1 1 B 10 1 2 W 10 2 2 B 10 1 BW WB	30 1 0

This sample only scores 40% of the points

SUBTASKS

For all cases:
$1 \leq N \leq R \times C$
$1 \leq R, C \leq 80$
$0 \leq v_i \leq 10^5$
$1 \leq K \leq 100$

	Points	Constraints
1	12	$R, C \leq 40$ $K = 1$ All wallpaper tiles are of the same color
2	15	$R, C \leq 40$ $K = 1$
3	16	$R, C \leq 40$ $N = R \times C$
4	21	$R, C \leq 40$ $v_i = 1$
5	12	$R, C \leq 40$
6	24	No additional constraints

S184 - BOGO TRANSLATE

Time Limit: 1.000 s / Memory Limit: 256 MB

Charlie is an intern at Bogo. He is developing a new feature "Bogo Translate". Due to his poor coding skills, he asks for your help.

Being an intern project, the system can only translate from language A into language B. To perform translation, the system should:

1. Convert each word in the sentence in language A into language B individually. Words in a sentence are separated by exactly 1 space. Each word consists of lowercase English letters only.
2. Rearrange the words according to the *sentence structure* difference.

The system has a database of N pairs of words $(\mathrm{Word}A_i, \mathrm{Word}B_i)$, $i = 1, 2, \ldots, N$. $\mathrm{Word}A_i$ is a word in language A while $\mathrm{Word}B_i$ is a word in language B. Note that $\mathrm{Word}A_i$ are distinct while $\mathrm{Word}B_i$ may not be distinct. The same database is used for all translation tasks.

In each translation task, the system should convert every appearance of $\mathrm{Word}A_i$ in the sentence into $\mathrm{Word}B_i$. Some words may not be found in $\mathrm{Word}A$ as they might be proper nouns. Those words should not be converted.

After that, the system shall rearrange the words according to the difference in sentence structures specific to the sentence to be translated. The sentence structures are represented by two strings: $\mathrm{Patt}A$ and $\mathrm{Patt}B$. If the number of words in the sentence is x, then they both consist of x distinct uppercase English letters. $\mathrm{Patt}B$ is a permutation of $\mathrm{Patt}A$. If $\mathrm{Patt}A[i] = PattB[j]$ then the i^{th} word shall become the j^{th} word in the translated sentence.

Example:

- $\mathrm{Word}A_1 =$ `charlie`, $\mathrm{Word}A_2 =$ `i`, $\mathrm{Word}A_3 =$ `am`
- $\mathrm{Word}B_1 =$ `charli`, $\mathrm{Word}B_2 =$ `watashiwa`, $\mathrm{Word}B_3 =$ `desu`
- Sentence: `i am charlie`
- After converting the words individually: `watashiwa desu charlie`
- Sentence structures: $\mathrm{Patt}A =$ `SVO`, $\mathrm{Patt}B =$ `SOV`
- After rearranging the words: `watashiwa charlie desu`

The efficiency of the system is vital for Charlie to receive a return offer. Please help him implement Bogo Translate, and to translate M sentences from language A to language B.

INPUT

The first line consists of a single integer N, the number of pair of words in the translation database.

Then N pairs of lines follow, the first line in the i^{th} pair is $\mathrm{Word}A_i$ and second line in the i^{th} pair is $\mathrm{Word}B_i$. All $\mathrm{Word}A_i$ are distinct.

The next line consists of a single integer M, the number of translation tasks.

M translation tasks follows. Each translation task is described by 3 lines. The first line is the sentence in language A. The second line contains a string $\mathrm{Patt}A$ and the third line contains a string $\mathrm{Patt}B$.

OUTPUT

Your output should consist of M lines. The i^{th} line should contain sentence translated into language B in the i^{th} translation task, using the method described above.

SAMPLE TESTS

1

Input	Output
4 d m r f m s f l 1 d r m d m d m r m f f m r f ABCDEFGHIJKLMN ABCDEFGHIJKLMN	m f s m s m s f s l l s f l

This sample is applicable to both subtask 1 and 2.

2

Input	Output
5 dont m i ngo love zungji you nei like zungji 2 i love hkoi SVO SVO you dont like noip SNVO SNVO	ngo zungji hkoi nei m zungji noip

This sample is applicable to Subtask 2.

3		
	0 5 charlie loves bogo MVF FVM single A A isnt the task cool WXYZ XYWZ to be or not to be ABCDEF EFCDAB senior is easier than junior SIETJ JIETS	bogo loves charlie single the task isnt cool to be or not to be junior is easier than senior

This sample is applicable to Subtask 3.

4		
	5 charlieli chariri i watashiwa am desu coding koodinguga like sukidesu 3 charlieli G G i am charlieli XYZ XZY i like coding SVO SOV	chariri watashiwa chariri desu watashiwa koodinguga sukidesu

SUBTASKS

For all cases:
$0 \leq N \leq 300$
$1 \leq M \leq 10000$
Every word in the database $(\text{Word}A_i, \text{Word}B_i)$ and the words in the sentences consist of at least 1 and at most 15 lowercase letters.
There are at least 1 and at most 26 words in each sentence.
The total number of words in all M sentences do not exceed 10000.
The sentences in the test cases may not be grammatically correct and even may not be written in real languages.

	Points	Constraints
1	18	Every word in the database $(\text{Word}A_i, \text{Word}B_i)$ and the words in the sentences have exactly 1 lowercase letter.
2	25	$\text{Patt}A = \text{Patt}B$ for each translation task.
3	31	$N = 0$
4	26	No additional constraints