Hong Kong Olympiad in Informatics 2025/26
Senior Group

Task Overview

ID	Name	Time Limit	Memory Limit	Subtasks
S261	Tribal Occupation	1.000 s	256 MB	11 + 12 + 17 + 22 + 23 + 15
S262	Even the Rhythm	2.000 s	1024 MB	4 + 8 + 17 + 18 + 11 + 15 + 13 + 14
S263	Set	1.000 s	256 MB	7 + 12 + 16 + 21 + 12 + 32
S264	Triangular Territories	1.000 s	256 MB	6 + 7 + 18 + 11 + 15 + 24 + 12 + 7

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 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.

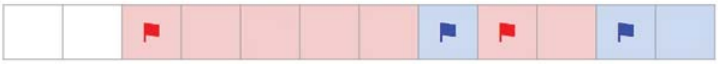
S261 - TRIBAL OCCUPATION

Time Limit: 1.000 s / Memory Limit: 256 MB

Two tribes, Red and Blue, are placing flags on a long line to claim territory.

The line is a grid with M cells, numbered 1 to M from left to right. There are N flags in total. The i-th flag has a color C_i (either Red or Blue) and a position P_i.

After all flags are placed, the tribes start to claim the grid cells as follows. For each flag, the owner of the flag, either the Red or Blue tribe, starts claiming the territory to the right of the flag, inclusive of the cell where the flag is placed on. A flag's territory stops at the cell before the next flag, while the last flag's territory extends up to cell M. The *total territory* of a tribe is the total number of cells claimed by the tribe.



You want to help the Red tribe get the most territory. To achieve that, you are allowed to move **at most one** red flag, which is done by picking an existing red flag on the grid, removing it and placing that flag in any cell (from 1 to M) that is **currently empty**. You can also choose to not move any flags.

Can you find the maximum possible total territory for the Red tribe after making your move optimally?

INPUT

The first line of input consists of two integers N and M, the number of flags and the size of the grid.

The next N lines each contain a character C_i followed by an integer P_i, the colour and the position of the flag, respectively. $C_i = \boxed{\texttt{R}}$ if the flag is Red, and $C_i = \boxed{\texttt{B}}$ if the flag is Blue.

It is guaranteed that each cell in the grid contains at most one flag, and the flags are given in ascending order of position. There is at least one flag coloured Red.

OUTPUT

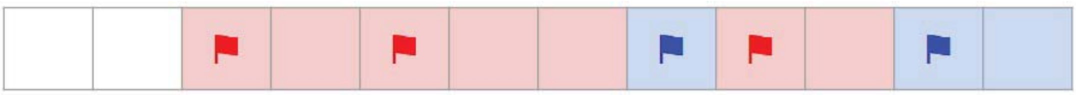
Output a single integer: The maximum possible total territory for the Red tribe.

SAMPLE TESTS

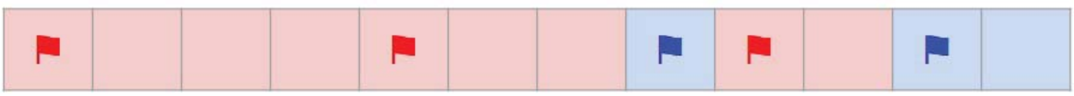
1

Input	Output
5 12 R 3 R 5 B 8 R 9 B 11	9

The grid cells originally occupied by the two tribes are shown in the following figure. The total territory of the Red tribe is 7.



The optimal strategy is to move the Red flag at position 3 to position 1. The total territory of the Red tribe becomes 9.



	Input	Output
2	4 9 B 3 R 5 B 8 R 9	5
3	1 10 R 6	10

You can move the single red flag to position 1.

SUBTASKS

For all cases:
$1 \le N \le 10^5$
$1 \le M \le 10^9$
C_i is either $\boxed{\texttt{R}}$ or $\boxed{\texttt{B}}$ for $1 \le i \le N$
$1 \le P_1 < P_2 < \cdots < P_N \le M$
There is at least one flag coloured Red.

	Points	Constraints
1	11	$N = 2$ $C_1 = \boxed{\texttt{R}}$ $C_2 = \boxed{\texttt{B}}$
2	12	$N \ge 2$ $C_1 = \boxed{\texttt{R}}$ $C_2 = \cdots = C_N = \boxed{\texttt{B}}$
3	17	$N \ge 2$ $C_1 = \boxed{\texttt{R}}$ $C_2 = \boxed{\texttt{R}}$
4	22	$N \le 100$ $M \le 500$
5	23	$N \le 2000$
6	15	No additional constraints

S262 - EVEN THE RHYTHM

Time Limit: 2.000 s / Memory Limit: 1024 MB

Bob's band is composing a new polyrhythmic track, and Bob needs to design the accompanying drum pattern.

The track consists of N bars. The i-th bar has a duration of A_i milliseconds.

To maintain a steady beat, Bob requires that the drum hits within any single bar be evenly spaced in time. For each bar i, he must choose a spacing X_i such that X_i is a **positive factor** of the duration A_i. Consequently, the number of notes played in bar i is exactly $\frac{A_i}{X_i}$.

However, to prevent exhaustion during the performance, Bob wants to play at most $K \geq N$ notes across the entire track. This means the chosen spacings must satisfy the condition $\frac{A_1}{X_1} + \frac{A_2}{X_2} + \cdots + \frac{A_N}{X_N} \leq K$.

Finally, for the track to sound cohesive, the spacings shouldn't fluctuate too wildly between different bars. Bob defines the *instability* of a way of choosing spacings $X_1, X_2, \ldots, X_N$ as the difference between the largest and smallest spacings used across all bars, which is $\max(X_i) - \min(X_i)$. Bob wants to minimize the instability while satisfying the conditions above.

Bob is a great drummer but he is not that good at programming. Can you help him design the track? If there are multiple spacings that achieve the minimal instability, you may output any of them. Note that under the constraints of this problem, there is always at least one valid way of choosing the spacings.

INPUT

The first line contains two integers N and K, the number of bars and the maximum number of notes.

The second line contains N integers $A_1, A_2, \ldots, A_N$, the duration of each bar in milliseconds.

OUTPUT

On the first line, output a single integer, the minimal instability you can achieve.

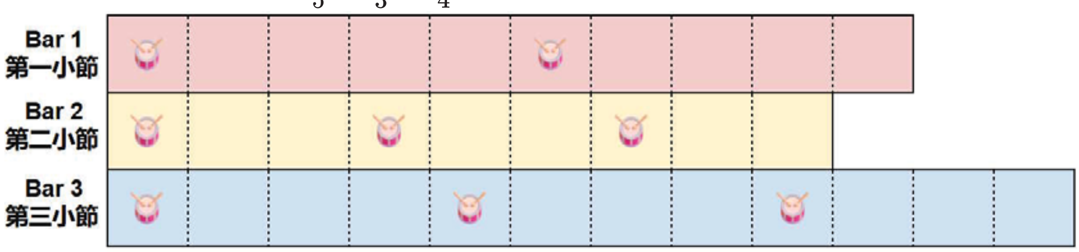
On the second line, output N integers $X_1, X_2, \ldots, X_N$, the spacings of the designed track.

If there are multiple spacings that achieve the minimal instability, you may output any of them.

SAMPLE TESTS

	Input	Output
1	`3 10` `10 9 12`	`2` `5 3 4`

The total amount of notes is $\frac{10}{5} + \frac{9}{3} + \frac{12}{4} = 8 \leq 10$, and the instability is $5 - 3 = 2$.



Note that $X = \{5, 3, 3\}$ is also a valid answer.

	Input	Output
2	2 3 1 2	0 1 1

The total amount of notes is $\frac{1}{1} + \frac{2}{1} = 3 \leq 3$, and the instability is $1 - 1 = 0$.

	Input	Output
3	6 20 2 3 5 7 11 13	10 2 3 5 7 11 1
4	6 20 1 3 5 7 9 11	6 1 3 5 7 3 1

SUBTASKS

For all cases:
$2 \leq N \leq 2 \times 10^5$
$1 \leq K \leq 10^{11}$
$1 \leq A_i \leq 5 \times 10^5$
$N \leq K \leq A_1 + A_2 + \cdots + A_N$

	Points	Constraints
1	4	$N = 2$ $A_1 = 1$ $A_2 = 2$
2	8	$N = 2$ $A_1 = 1$
3	17	$N \leq 2000$ $A_i \leq 2000$ for all $1 \leq i \leq N$ $A_1 \leq A_2 \leq \cdots \leq A_N$ A_i is prime for all $1 \leq i \leq N$
4	18	$N \leq 100$ $A_i \leq 100$ for all $1 \leq i \leq N$
5	11	$N \leq 2000$ $A_i \leq 2000$ for all $1 \leq i \leq N$
6	15	$N \leq 5 \times 10^4$ $A_i \leq 10^5$ for all $1 \leq i \leq N$ $A_1 = 1$
7	13	$N \leq 5 \times 10^4$ $A_i \leq 10^5$ for all $1 \leq i \leq N$
8	14	No additional constraints

S263 - SET

Time Limit: 1.000 s / Memory Limit: 256 MB

Have you played the card game *Set* before? In *Set*, a **card** consists of L distinct **features**, such as shading, number of shapes, etc, where each **feature** has **three possibilities**. For instance, the possibilities of shading can be open, striped or solid, while there can be one, two or three shapes on a card.

Formally, we number the features from 1 to L and label the possibilities of a feature as $\boxed{\text{A}}$, $\boxed{\text{B}}$ and $\boxed{\text{C}}$. Each card is represented by a length-L string consisting of $\boxed{\text{A}}$, $\boxed{\text{B}}$ and $\boxed{\text{C}}$, where the j-th ($1 \le j \le L$) character represents the j-th feature of the card.

A **complete deck** consists of 3^L cards, where each card with unique combination of features appears exactly once. For example, a complete deck of $L = 2$ has $3^2 = 9$ cards, which are represented by $\boxed{\text{AA}}$, $\boxed{\text{AB}}$, $\boxed{\text{AC}}$, $\boxed{\text{BA}}$, $\boxed{\text{BB}}$, $\boxed{\text{BC}}$, $\boxed{\text{CA}}$, $\boxed{\text{CB}}$ and $\boxed{\text{CC}}$ respectively.

During a game of *Set*, several cards are placed on a table, and players compete to find a **valid** set of three cards as fast as possible. A set of three cards is **valid** if and only if for each feature, the three cards are either all same or all different. For example,

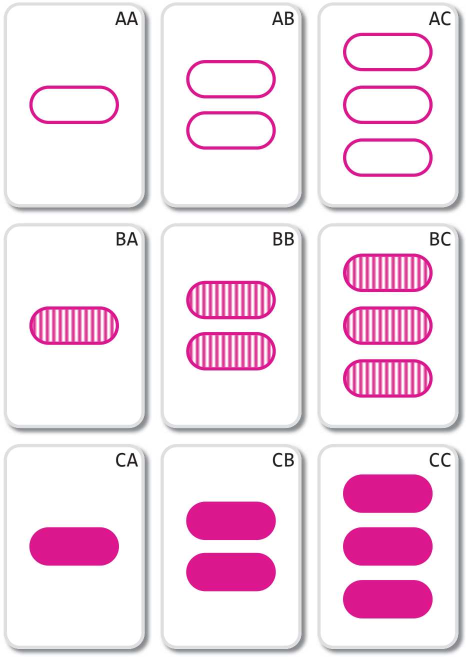


Figure 1: A complete deck of $L = 2$.

- The set of cards $\boxed{\text{AB}}$, $\boxed{\text{BB}}$ and $\boxed{\text{CB}}$ is valid, since they are all different in feature 1 and all same in feature 2.
- The set of cards $\boxed{\text{AA}}$, $\boxed{\text{AB}}$ and $\boxed{\text{BC}}$ is invalid, since they are neither all same nor all different in feature 1.
- The set of three identical cards is always valid, since each of the L features of the three cards are the same.

Bob loves playing *Set* very much, but since he is too strong, no one wants to play against him anymore! Therefore, he decides to invent a single-player mode of *Set*, which works as follows:

- First draw N cards (where N is a multiple of 3) and place them onto the table. The drawn cards may contain duplicates.
- Then, one can unpack **any number of (possibly zero) complete decks**, and add all the cards in the decks onto the table.
- Next, one may repeatedly choose any three cards that form a **valid** set of three, and remove them from the table.
- The game is won when all the cards are removed from the table.

Bob has just won a game of *Set* in single player mode - and decides to challenge you! Given the initial N cards that Bob has drawn, can you win the game? Since you would have to clean up the table afterwards, you don't want to unpack too many complete decks (Please refer to the output section).

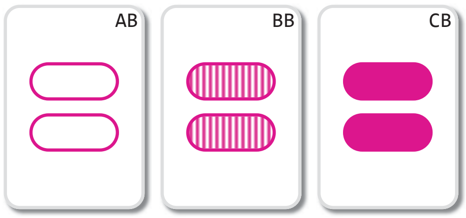


Figure 2a: A valid set of cards.

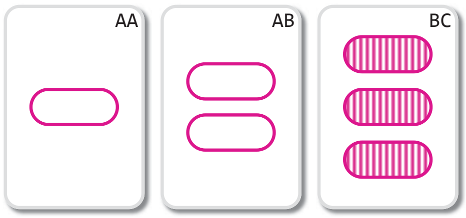


Figure 2b: An invalid set of cards.

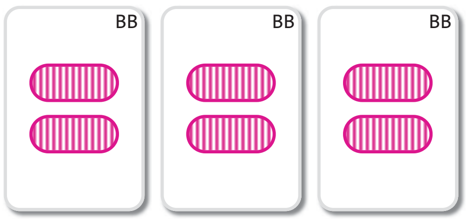


Figure 2c: Set of three identical cards is always valid.

Notes

Since Bob is unsure if everyone knows a related (**may or may not be useful**) fact from the HKOI Heat Event, out of fairness, he let you know that if we represent the three possibilities `A`, `B` and `C` of a feature as 1, 2 and 3, then three cards show all same or all different in that feature if and only if the sum is divisible by 3.

This is because the cases where the feature are all same or all different are `AAA`, `BBB`, `CCC`, `ABC`, `ACB`, `BAC`, `BCA`, `CAB` and `CBA`, exactly covers every case where the sum is divisible by 3. For instance, for the scenario `CCC`, the corresponding sum is $3+3+3=9$, which is divisible by 3.

INPUT

The first line contains two integers N and L, the number of cards and the number of features on each card, respectively.

Each of the next N lines contains a length-L string S_i ($1 \leq i \leq N$), consists of `A`, `B` and `C` only, describing the i-th card. The j-th ($1 \leq j \leq L$) character of S_i represents the j-th feature of the i-th card, where `A`, `B` and `C` represents the three possibilities of that feature.

OUTPUT

On the first line, output two non-negative integers M and K, the number of extra complete decks you used and the number of sets you made, respectively.

On the next K lines, each output a valid set of three cards. Each card should be represented by a length-L string, which consists of `A`, `B` and `C` only. Each card must be used exactly once.

Note that K must not exceed 2×10^5. In other words, the total number of cards $N + M \times 3^L = 3K$ must not exceed 6×10^5.

If there are multiple ways to win the game, you can output any of them. Please note that it is not necessary to minimise K as long as $K \leq 2 \times 10^5$.

It is guaranteed that the game can be won **in the given test cases**. It can be shown that under the constraints of the problem, if a game can be won, there exists a way to win with K not exceeding 2×10^5.

SAMPLE TESTS

	Input	Output
1	6 2 AB AB AC AC BB CC	1 5 AA BB CC AB AB AB AC AC AC BA BB BC CA CB CC

We can unpack a complete deck, which consists of $3^2 = 9$ cards, represented by `AA`, `AB`, `AC`, `BA`, `BB`, `BC`, `CA`, `CB` and `CC` respectively. Then, we can put the $6 + 9 = 15$ cards into 5 valid sets of three.

	Input	Output
2	6 1 A B A C A A	0 2 A A A A B C
3	6 1 A B A C A A	1 3 A A A A B C A B C

It is not necessary to unpack the minimum number of complete decks.

SUBTASKS

For all cases:
$3 \le N \le 60000$
N is a multiple of 3
$1 \le L \le 9$
The game can be won in all the test cases

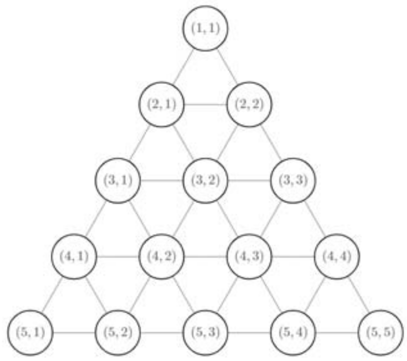
	Points	Constraints
1	7	$N = 3^L$ The initial cards form a complete deck
2	12	$L = 1$
3	16	$L = 2$ Exactly 1 card starts with B Exactly 1 card starts with C
4	21	$L = 2$
5	12	$L \le 4$
6	32	No additional constraints

S264 - TRIANGULAR TERRITORIES

Time Limit: 1.000 s / Memory Limit: 256 MB

Two tribes, Red and Blue, are placing flags to claim territory again. But this time, instead of a long line, they are claiming territory on a triangular grid.

There are N rows on the triangular grid. The rows are numbered from 1 to N from top to bottom, so the row at the top is row 1, and the row at the bottom is row N. For all $1 \le i \le N$, on row i, there are exactly i cells, numbered from 1 to i from left to right. We denote cell j on row i as cell (i, j).

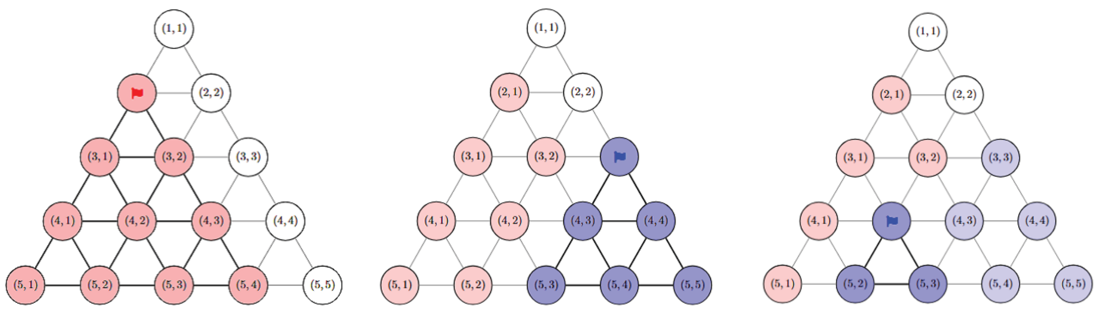


(A triangular grid with $N = 5$ rows)

Each tribe claims the grid cells as follows: place a flag of the color of the tribe (either red or blue) on any cell on the triangular grid. Then, all cells on the sub-triangle starting at the flag's cell and ending at the bottom row are claimed by the tribe. Note that if a cell on the sub-triangle is previously claimed by another tribe, the cell will be claimed by the owner of the newly placed flag and will no longer be claimed by the previous tribe. Formally, once a flag is placed on cell (u, v), all cells (i, j) satisfying $u \le i \le N$ and $v \le j \le v + (i - u)$ are claimed by the owner of the flag, regardless of whether cell (i, j) is claimed by any tribes before. Note that flags can be placed on cells claimed by another tribe, and more than one flag (either of the same or different color) can be placed on the same cell.

Consider the following example of a sequence of flag placements on a triangular grid of size $N = 5$. The configuration of the grid after each flag placement is shown.

1. A red flag is placed on cell $(2, 1)$;2. A blue flag is placed on cell $(3, 3)$;3. A blue flag is placed on cell $(4, 2)$.



The above sequence of flag placements results in the Red tribe claiming 5 cells and the Blue tribe claiming 8 cells.

The tribes have been placing flags and claiming territories over the past few months. Currently, there are exactly X cells on the grid which belong to the Red tribe, and exactly Y cells on the grid which belong to the Blue tribe. It is recorded that the Red tribe has placed at most A red flags while the Blue tribe has placed at most B blue flags. However, the positions of the flags, as well as the placement order of the flags, are unknown.

Your task is to recover a sequence of flag placements on the grid, such that the sequence consists of at most A red flags and at most B blue flags, and it results in exactly X cells being claimed by the Red tribe and exactly Y cells being claimed by the Blue tribe on the triangular grid. It is also possible that the records are wrong and it is impossible to construct such sequence, in that case, report so.

INPUT

The only line of the input consists of five integers, N, A, B, X, Y.

OUTPUT

If it is impossible to construct a sequence satisfying the constraints, output `-1`.

Otherwise, the first line of the output consists of an integer K, the length of the sequence of flag placements. K must satisfy $0 \le K \le A + B$.

The next K lines of the output each describes a flag placement. The i-th of these lines should contain a character c_i and two integers x_i, y_i, meaning that the i-th flag placed has color c_i and is placed on cell (x_i, y_i). c_i must be either `R` for red or `B` for blue. (x_i, y_i) must be a cell inside the grid, i.e. $1 \le y_i \le x_i \le N$ must hold.

The number of i where $c_i =$ `R` should be at most A, and the number of i where $c_i =$ `B` should be at most B.

The sequence of flag placements should result in exactly X cells being claimed by the Red tribe and exactly Y cells being claimed by the Blue tribe.

If there are multiple sequences that satisfy the constraints above, you may output any of them. Note that you do not have to minimize the length of the sequence.

SAMPLE TESTS

	Input	Output
1	5 2 2 5 8	3 R 2 1 B 3 3 B 4 2

The output corresponds to the sequence of flag placements shown in the problem statement.
This sample test satisfies the constraints of subtasks 5, 7 and 8.

	Input	Output
2	5 2 2 5 8	4 R 3 3 R 2 1 B 4 3 B 3 1

This is another valid sequence of flag placements for sample test 1.

	Input	Output
3	6 1 3 6 5	4 B 6 5 R 3 1 B 6 3 B 5 1

This sample test satisfies the constraints of subtasks 6, 7 and 8.

	Input	Output

4	`6 9 4 2 0`	`4`
		`B 6 2`
		`B 6 1`
		`R 6 2`
		`R 6 1`

This sample test satisfies the constraints of subtasks 5, 7 and 8.

	Input	Output
5	`4 1 1 4 5`	`-1`

This sample test satisfies the constraints of subtasks 4, 7 and 8.

SUBTASKS

For all cases:

$1 \le N \le 500$

$0 \le A, B, X, Y \le \frac{N(N+1)}{2}$

$1 \le A + B \le \frac{N(N+1)}{2}$

$1 \le X + Y \le \frac{N(N+1)}{2}$

	Points	Constraints
1	6	$A = 0$ $B = \frac{N(N+1)}{2}$
2	7	$A = 0$ $B = 1$
3	18	$A = 0$ $B \ge 2$
4	11	$A = B = 1$
5	15	$A, B \ge 2$
6	24	$A = 1$ $B \ge 3$
7	12	$N \le 100$
8	7	No additional constraints