Problem A: Bookkeeping

Filename: `bookkeeping`
Time limit: `2 seconds`

Jimbo finds himself unable to concentrate while doing a reading assignment for his literature class. As he stares at the bland, monospaced page, the letters start to look like they're popping out of the page. Jimbo wonders, if he were to construct a rectangle of a certain size and place it somewhere optimally on the page aligned with the letters such that it would have the maximum number of occurrences of a kind of letter, what would that number of occurrences be?

Input

The first line of input contains two integers, ***w*** and ***h*** ($1 \le w, h \le 1{,}000$), representing the width of the page in letters and the height of the page in letters, respectively. The second line contains two integers, ***a*** and ***b*** ($1 \le a \le w$; $1 \le b \le h$), representing the width of the rectangle in letters and the height of the rectangle in letters, respectively. Following this will be ***h*** lines, each containing a string of ***w*** lowercase letters, representing what the page looks like.

Output

Output 26 lines, one for each letter of the alphabet, with the i^{th} line containing an integer representing the maximum number of occurrences of the i^{th} letter of the alphabet within the rectangle if it were placed optimally on the page.

Samples

Input	Output
3 3 2 1 ddd fbc cba	1 1 1 2 0 1 0 0 0 0 0 0 0 0 0 0 0

	0 0 0 0 0 0 0 0 0
3 3 1 2 ddd fbc cba	1 2 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Problem B: Holes

Filename: `holes`
Time limit: `4 seconds`

Bessie likes to play a lot. Most of all she likes to play a game “Holes”. This is a game for one person with following rules:

There are ***N*** holes located in a single row and numbered from left to right with numbers from 1 to ***N***. Each hole has its own power (hole number *i* has the power a_i). If you throw a ball into hole *i* it will immediately jump to hole *i* + a_i, then it will jump out of it and so on. If there is no hole with such number, the ball will just jump out of the row. On each of the ***M*** moves the player can perform one of two actions:

- Set the power of the hole *a* to value *b*.
- Throw a ball into the hole *a* and count the number of jumps of a ball before it jumps out of the row and also write down the number of the hole from which it jumped out just before leaving the row.

Bessie is not good at math, so, as you have already guessed, you are to perform all computations.

Input

The first line contains two integers ***N*** and ***M*** ($1 \le N \le 10^5$, $1 \le M \le 10^5$) — the number of holes in a row and the number of moves. The second line contains ***N*** positive integers not exceeding ***N*** — initial values of holes power. The following ***M*** lines describe moves made by Bessie. Each of these line can be one of the two types:

- 0 *a b*
- 1 *a*

Type 0 means that it is required to set the power of hole *a* to *b*, and type 1 means that it is required to throw a ball into the *a*-th hole. Numbers *a* and *b* are positive integers do not exceeding ***N***.

Output

For each move of the type 1 output two space-separated numbers on a separate line — the number of the last hole the ball visited before leaving the row and the number of jumps it made.

Samples

Input	Output
8 5 1 1 1 1 1 2 8 2 1 1 0 1 3 1 1 0 3 4 1 2	8 7 8 5 7 3
4 2 1 3 2 1 0 2 2 1 2	4 2

Problem C: More Ice Cream for Anya

Filename: `moreicecream`
Time limit: `1 second`

Anya really enjoyed the ice cream tree her parents gave her last time, so when she graduated college, naturally her parents gave her another ice cream tree, rooted at node **1**.

You may remember from last time that each node in the tree has an ice cream tub with capacity k_i gallons.

However, Anya has grown much more sophisticated. To make her ice cream tree more fun (and to have the coolest ice cream tree in the neighbourhood) Anya also occasionally updates ice cream buckets.

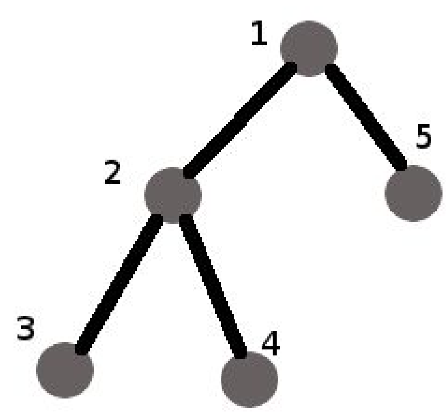
When it comes time to enjoy some ice cream, Anya again wants to select some ice cream tub from a subtree. The ***i***th time she picks ice cream she picks it like so:

1. For all ice cream tubs in the subtree, Anya counts the number of one bits in the binary representation of the tub's capacity.
2. Of all the tubs in that subtree, Anya highlights all of the tubs whose capacities have the largest number of one bits in their binary representations (of course, all capacities are positive integers).
3. Of the highlighted ice cream tubs, Anya selects the one with the highest capacity, because the only thing Anya likes more than one bits is LOTS of ice cream.

As said earlier, Anya also might change things up every now and then, replacing an ice cream tub of capacity k_i with one of capacity c_i.

Help determine if Anya will have a good summer by answering ***Q*** queries.

Remember: Subtrees are numbered in the order they are visited when traversing the tree from the root (also known as a preorder traversal). The root of the tree is the ***1st*** subtree. Refer to the image below for an example. If two or more nodes have the same parent, the earlier subtree is the one rooted at the node with the smaller index ***i*** as indicated in the input.



Input

- The first line contains two integers, ***N*** and ***Q*** ($1 \le N,Q \le 10^5$), the number of nodes in Anya's ice cream tree and how many queries will follow, respectively. The following line contains ***N*** integers, the i^{th} of which is the initial capacity, k_i ($1 \le k_i \le 10^9$) of the i^{th} ($1 \le i \le N$) ice cream tub. Next is a line containing ***N*** - 1 integers.The i^{th} ($1 \le i \le N-1$) of these integers is p_{i+1}, the parent of node ***i***+1.

- Then come the Q queries. Each query is on its own line. First is an integer t_i, the type of the ith query.
  - If the query type is **1**, following t_i is r_i ($1 \le r_i \le N$), meaning that node r_i is the root of the subtree relevant to the query. For this query you must find the capacity of the ice cream tub, in the subtree rooted at r_i, that Anya picks.
  - If the query is type **2**, following t_i are the integers r_i ($1 \le r_i \le N$) and c_i ($1 \le c_i \le 10^9$), the node whose tub to change and the new capacity of that tub, respectively.

Output

For queries of type 1, output the capacity of the ice cream tub Anya selects.

Samples

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td><pre>4 4
1 2 8 5
1 2 1
1 2
1 1
2 2 7
1 2</pre></td><td><pre>8
5
7</pre></td></tr>
<tr><td><pre>4 4
3 6 9 12
1 2 3
1 1
2 1 7
1 1
1 2</pre></td><td><pre>12
7
12</pre></td></tr>
</table>

Problem D: Test

Filename: `test`
Time Limit: `2 seconds`

Sometimes it is hard to prepare tests for programming problems. Test Student is preparing tests for a new problem about strings — input data to his problem is one string. Test Student has 3 wrong solutions to this problem. The first gives the wrong answer if the input data contains the substring ***A***, the second enters an infinite loop if the input data contains the substring ***B***, and the third requires too much memory if the input data contains the substring ***C***. Bob wants these solutions to fail single test. What is the minimal length of test, which couldn't be passed by all three Bob's solutions?

Input

The input consists of three space separated strings on one line: ***A*** ($1 \leq |A| \leq 10^5$), ***B*** ($1 \leq |B| \leq 10^5$), and ***C*** ($1 \leq |C| \leq 10^5$). All three strings consist of only lowercase Latin letters.

Output

Output one number — the minimal length of any string, containing ***A***, ***B*** and ***C*** as substrings.

Samples

Input	Output

`ab`
`bc`
`cd`
	`4`

`abacaba`
`abaaba`
`x`
	`11`

Sample Explanations:

Sample 1: "abcd" is length 4. Sample 2: "abaabacabax" is length 11.

Problem E: Extra Set

Filename: `extraset`
Time limit: `2 seconds`

Anya loves playing the game Set. The game consists of 81 cards. Each card has a picture with four attributes: number, color, shape and shading and there are three possible values for each attribute (number - {1, 2, 3}, color - {red, green, purple}, shape - {oval, diamond squiggle}, shading - {none, lines, solid}).

In the game, several cards are laid out and the first person to claim a set, wins that set. A set is three cards such that for each attribute, all three cards in the set either share the attribute or are all different for that attribute. For example, the cards [1, red, oval, lines], [2, red, oval, solid], [3, red, none] form a set because they have three different numbers, the same color, the same shape, and three different shadings.

Anya already regularly beats Arup, but Arup would like to train Anya to be even better in the game. He's thought of a harder version of the game where each card has k attributes, each of which has three possible values taken from the set {0, 1, 2}. Of the possible 3^k distinct cards, consider a set of n of these cards laid out. Arup would like Anya to be able to figure out how many combinations of three cards out of the n cards laid out form sets (a set of three cards where each attribute is either shared or different). Write a program to help Anya ace Arup's challenge!

Given the number of attributes for a Set card game, as well as the description of several cards from the game, determine the number of combinations of three cards from the given cards that form sets.

Input

The first line of input for will contain two space separated positive integers: $\boldsymbol{k}$ ($3 \le \boldsymbol{k} \le 19$), representing the number of attributes for the cards in the game, and $\boldsymbol{n}$ ($3 \le \boldsymbol{n} \le 1500$), representing the number of cards laid out for the game. The cards for the game follow, one per line. In particular, the i^{th} of these lines will contain $\boldsymbol{k}$ space separated integers $\boldsymbol{c_{i,1}}, \boldsymbol{c_{i,2}}, \boldsymbol{c_{i,3}}, \ldots, \boldsymbol{c_{i,k}}$ ($0 \le \boldsymbol{c_{i,j}} \le 2$), where $c_{i,j}$ represents the $\boldsymbol{j}^{th}$ attribute value of the $\boldsymbol{i}^{th}$ card. It is guaranteed that each card in a single game will be unique.

Output

On single line, output a single integer on a line by itself: the number of different combinations of three cards that form a set.

Samples

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

Problem F: Gauss's Detention

Filename: `gauss`
Time limit: `1 second`

Gauss and his class misbehaved, so his teacher gave the class the task of adding the first 100 integers. Gauss thwarted his teacher's intention by discovering a formula for the sum and avoided doing any addition! Now, his teacher wants revenge! He's figured out that Gauss doesn't like adding large numbers. In fact, if he's adding two numbers a and b, it takes him $a+b$ ms. Thus, his teacher has decided to give Gauss a random string of numbers with no pattern, to add. For example, if Gauss had to add 137, 213, 98 and 49, he could add 137+213 = 350, then add 98 + 49 = 147 and finally add 350 + 147 = 497. The total amount of time this would take him if he added the numbers in this order is 350 ms + 147 ms + 497 ms = 994 ms. It turns out, it would have been better if he added 98 + 49 = 147, 147 + 137 = 284 and 284 + 213 = 497, which would have taken him 147 ms + 284 ms + 497 ms = 928 ms. Given a list of numbers Gauss has been asked to add, determine the minimum amount of time it will take him in ms to calculate the sum, assuming he adds up the numbers in the optimal order.

Input

The first line of input will contain a single positive integer $\boldsymbol{n}$ ($\boldsymbol{n} \leq 30000$), representing the number of positive integers Gauss has to add for the input case.

The second line of each input case will contain the $\boldsymbol{n}$ space separated integers:
$\boldsymbol{a}_i$ ($1 \leq i \leq \boldsymbol{n}$, $1 \leq \boldsymbol{a}_i \leq 4000$) that Gauss must add for the input case.

Output

For each input case, on a line by itself, output the minimum amount of time in ms that it will take Gauss to complete the addition posed to him.

Samples

Input	Output
3 1 2 4	10
4 137 213 98 49	928
8 1 4 9 16 25 36 49 64	512

Problem G: Phil and Plinko

Filename: `plinko`
Time limit: `4 seconds`

The carnival's in town, and Phil the Philosopher has gone to play some games on the midway. He decides to play a game similar to Plinko. Phil has philosophized that, unlike most carnival games, this one can be won deterministically as long as he makes the right choices.

The Plinko game is made up of a grid of several dials and blocks that control the path of a small ball. When a ball falls onto a dial, depending on what direction the dial is facing, it can travel to any of the three spaces closest to it on the row below. The ball can not travel into any space that contains a block, and it will come to rest if it's directed towards one. Phil is allowed to change the direction of up to K of the dials on the grid before dropping a ball onto any dial on the grid.

Each of the spaces on the grid containing a dial have a score. If the ball travels through a space before falling off the grid or coming to rest, that space's value is added to Phil's score. If Phil turns the dials optimally and drops the ball in the right place, what is the best score he can get?

Input

The first line of input will contain three positive integers: $\boldsymbol{N}$, $\boldsymbol{M}$ and $\boldsymbol{K}$ ($\boldsymbol{N},\boldsymbol{M} \leq 10^5$, $\boldsymbol{N} \cdot \boldsymbol{M} \leq 10^5$, $\boldsymbol{K} \leq 200$), representing the number of rows in the grid, the number of columns in the grid, and the maximum number of dials Phil can change. Following this are $\boldsymbol{N}$ lines of $\boldsymbol{M}$ characters representing the original state of the Plinko grid. The characters **/**, **|**, and **** correspond to dials causing the ball to move down/left, straight down, and down/right respectively. **#** represents a block. Following this are $\boldsymbol{N}$ lines, each containing $\boldsymbol{M}$ space separated integers $\boldsymbol{a}_{i,j}$ ($0 \leq \boldsymbol{a}_{i,j} \leq 10^4$), representing the score of the dial in the j^{th} cell of the i^{th} row.

Output

On a line by itself, output a single integer: the highest score Phil can get.

Samples

<table>
<tr><th>Input</th><th>Output</th><th>Explanation</th></tr>
<tr><td>4 4 2
\//|
#\||
\#\/
|//\
3 6 4 3
0 4 5 3
9 0 7 3
4 7 4 6</td><td>26</td><td>\//| \|/|
#\|| #/||
\#\/   \#\/
|//\ |//\

The best path for the ball is highlighted above. The right side shows the grid after changes to 2 dials.</td></tr>
</table>

Problem H: Purple Rain

Filename: `purplerain`
Time limit: `4 seconds`

Purple rain falls in the magic kingdom of Linearland which is a straight, thin peninsula. On close observation however, Prof. Nelson Rogers finds that actually it is a mix of Red and Blue drops. In his zeal, he records the location of each of the raindrops to fall with its corresponding color in different locations along the peninsula. He wants to answer the following question: which section of Linearland had the least purple rain? That is, which section had the greatest difference between red rain and blue rain?

After some thought, he decides to model the problem as follows: Divide the peninsula into ***n*** sections and describe it as a sequence of ***R*** or ***B*** values depending on whether the rainfall in that section is primarily red or blue. Then, find the part consisting of consecutive sections where the absolute difference of the count of ***R***s and ***B***s is maximized.

Input

The input will consist of a single line with a string ***s*** ($1 \le |s| \le 100{,}000$), where every character in ***s*** is either a capital ***R*** or a capital ***B***. This string describes the peninsula, from west to east.

Output

Output two integers, indicating the start and end of the part of the peninsula which maximizes the difference between ***R***s and ***B***s. The first character of ***s*** is at position 1, and the last is at position ***n***. Output the smaller index first. If there are multiple parts that feature the same maximal absolute difference, print the one with the smallest starting position. If there are multiple such parts starting at that same smallest starting position, print the shortest of those.

Samples

Input	Output
`BBRRBRRBRB`	`3 7`
`BBRBBRRB`	`1 5`