Problem A. Allocation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Your task is to implement the memory manager. The computer have 10^5 consecutive memory blocks, numbered from 1 to 10^5. Initially all the blocks are free.

The manager shall support those functions:

- `varname=alloc(size);` — finds the first *size* consecutive memory block and reserves them. The function sets value of *varname* variable to the address of the first reserved block. If there is no *size* consecutive blocks, then function reserves nothing and returns a value of 0.
- `free(varname);` — frees the memory allocated for variable *varname* by previously called function `alloc` and sets the value of the *varname* to 0. If no memory was allocated for the variable or the variable is freed, the function does nothing.
- `output(varname);` — prints the value of variable *varname*.

Each command ends with a semicolon (‘`;`’). Variables are strings composed of **exactly 4** lowercase English letters. All used variables are initially set to 0.

Input

The first line contains the integer n ($1 \leq n \leq 10^5$) — the number of function calls. In the i-th of the following n lines i-th function call is found. The function calls are formatted according with the problem statements without any extra blank characters. The total number of different variables will be less than or equal to 1000. At least one of the commands will be the command `output`. Size in the `alloc` command may vary between 100 and 10^5, inclusively.

Output

For each `output` call print the result of it at the new line.

Examples

standard input	standard output

3
data=alloc(239);
more=alloc(17);
output(more);
	240

5
long=alloc(50002);
file=alloc(50000);
output(long);
free(long);
output(file);
	1
	0

Problem B. Binary Trees

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 mebibytes

A binary tree is a hierarchical structure consisting of n nodes numbered 1 to n. Each node k of a tree can have a *left child* l_k and a *right child* r_k. If node m is a child of node k, then we say that k is the *parent* of node m. The node labeled number 1 is called the *root* of the binary tree. Root have no parent, while every other node have a unique parent.

Descendants of a node k are all nodes that can be reached by walking from parents to childs; as the one of consequences, it is true that all nodes (except root itself) are descendants of the root.

A *subtree* is a binary tree formed by a node k and all its descendants.

Lets assign to each node i the integer value v_i.

A binary tree is called the *binary search tree* if for each node k of the tree:

- If k has a left child l_k, then the value of node l_k and all its descendants is less than or equal to of node value k.
- If k has the right child r_k, then the value of node r_k and all its descendants is greater or equal of node value k.

Note that each subtree of the binary search tree must also be a binary search tree.

Given a binary tree. Your task is to process q queries are given. Each query indicates that the value of the node k_i is set to x_i. Answer to each query is the number of subtrees that are binary search trees.

Input

The first row of inputs contains the integers n and q – number of nodes in the tree and the number of queries ($1 \le n, q \le 2 \cdot 10^5$). i-th of the following n lines contains two integers l_i and r_o ($0 \le l_i, r_i \le n$) — the numbers of the left and right child of node i. If the node does not have left or right child, l_k or r_k will be zero.

The next line contains n integers $v_1, v_2, \ldots, v_n$ ($0 \le v_k \le 10^9$) — initial values, assigned to the nodes of the tree.

Then q lines follow, describing the queries. Each query contains two integers k_i and x_i ($1 \le k_i \le n$, $0 \le x_i \le 10^9$) number of node and the new value.

Output

Print q lines. i-th of those lines must contain one integer — number of subtrees that are binary search trees after i-th query.

Examples

standard input	standard output

6 5
2 3
4 0
5 6
0 0
0 0
0 0
4 1 3 2 2 5
3 3
2 2
3 5
5 4
6 1
	4
	5
	5
	6
	4

8 10
4 5
8 0
0 0
3 7
0 6
0 0
2 0
0 0
7 0 9 3 6 0 6 2
3 0
4 0
8 2
2 3
7 6
1 6
5 7
6 9
1 1
1 7
	3
	3
	3
	6
	6
	6
	6
	8
	7
	8

Problem C. Counting Bit Strings

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

Consider the *pattern* as the string of the length N consisting of characters '0' and '*'. Binary string of length N is accepted by the pattern if all characters '0' in the pattern correspond to characters '0' in the string (wildcard may correspond to either '0' or '1').

Count number of strings which is accepted by given pattern and have next additional property: each substring of length L contains no more than P '1'-s.

Input

First line of the input contain three integers N ($1 \le N \le 100$), L ($1 \le L \le 50$) and P ($0 \le P \le 5$) — length of the pattern and additional parameters, respectively. Second line of the input contains the pattern — string of length N, consisting of characters '*' and '0'.

Output

Print the answer to the problem modulo 998 244 353.

Examples

standard input	standard output
7 2 1 *******	34
10 2 1 ****0*****	104

Problem D. Deep Red

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 Mebibytes

The famous company HAL is invented the new supecomputer, called Deep Red. It is a fantastic machine — it has an infinite amount of memory and it processes all operations infinitely fast. Bytica, the senior programmer in HAL, has to write programs for this computer.

However, Deep Red is in pre-alpha testing stage, so it still has several problems. More precisely, there are three of them.

1. The processor of Deep Red has just two operations: it can only add or multiply two integers stored in its memory, and write the result into memory. Adding or multiplying an integer stored in memory by itself is also allowed. No other operations and programming constructions such as cycles, conditions and even writing a constant to memory are possible for now.

2. Although operations are performed correctly, the process of writing integers to the memory has a bug: while writing, the number may change, but at most by 1. So, if you calculate $2 \cdot 3$, then the result will be 6, but the result stored in memory can be 5, 6 or 7.

3. Deep Red has no input or output devices. After boot, its memory contains just two positive integers x and y. These numbers are the same each time the computer boots.

The first problem that Bytica wants to solve is to store some integer z in the memory. Of course, Bytica understands that getting precisely z can be impossible. So, she wants to write a program that produces a result as close as possible to z in the worst case.

The result of running a program is the last number stored by it in memory.

Input

Input file contains three integers x, y and z ($1 \le x, y, z \le 1000$, $z \ne x$, $z \ne y$).

Output

Write the distance to z that Bytica's code will get in the worst case.

Examples

standard input	standard output
`2 3 6`	`1`
`2 3 7`	`2`

Problem E. Edge Sets

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 mebibytes

Given the tree. Your job is to answer q queries.

Each query conststs of the subset S of the vertices, and you shall answer, how many edges belong to atleast one path between two of vertices from S.

Input

First line of the input contains one integer n ($2 \leq n \leq 10^6$) — number of the vertices of the tree. Then $n-1$ lines follow, each containing two distinct integers a and b ($1 \leq a, b \leq n$) — the vertices of the tree, connected by a edge. Then one integer q follows ($1 \leq q \leq 6 \cdot 10^4$) — the number of the queries. Then q queries follow. Each query starts with integer k_i ($2 \leq k_i \leq n$) — number of the vertices in the query, followed by k_i pairwise distinct integers between 1 and n, inclusively — indices of vertices in the query. You may assune that sum of all k_i for all queries does not exceed $6 \cdot 10^4$.

Output

For each query, print the answer on it.

Examples

standard input	standard output

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

```
10
1 2
2 5
4 2
6 3
3 1
6 7
9 7
8 6
8 10
5
2 4 5
3 4 5 6
4 2 5 7 10
4 4 5 9 10
9 1 2 3 4 6 7 8 9 10
```
	```
	2
	5
	7
	9
	8
	```

Problem F. Fractal Enumeration

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Given the part of the coordinate plane, delimited by the square with the bottom left corner placed at the point $(-2^n+1, -2^n+1)$ and the upper right corner at $(2^n-1, 2^n-1)$. All points inside this square such as both x- and y- coordinates are odd, painted in black and numbered by sequential integers between 1 and 4^n.

The points are numbered following the next rules:

- For i-th quardant $(1 \le i \le 3)$ all points in this quardrant are going earlier than points in quadrant $i+1$.
- At the next step we consider each quadrant as separate square and recursively apply the rule, mentioned above. As the coordinate axis for the new squares are used their axis of symmetry, parallel to existing coordinate axis.

Given integer k, your job is to calculate sum of the numbers of the black points such as $x + y = k$.

Input

First line of the input contains two integers n and k ($1 \le n \le 60$, $-2^{63} \le k \le 2^{63} - 1$. You may assume that for at least one black point $x + y = k$.

Examples

standard input	standard output
3 0	344
2 8	6
3 10	21

Problem G. Good Integers

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Consider the rules of *compatibility* on the non-zero digits in the next way:

- 1 is compatible only with 2, 9 is compatible only with 8, 4 is compatible only with 3.
- 2 is compatible with 1, 3 and 5; 5 is compatible with 2 and 6.
- Other digits (3,6,7,8) are compatible with two digits whose values differ from this one by 1.

The integer is considered to be `good` if it does not contain the zeroes in its decimal representation d and if for each i from 1 to $|d|-1$ inclusively i-th and $i+1$-th digits are compatible.

Construct any bijection between good integers and the strings, consisting of the characters '`A`' and '`B`' only, More specifically, given a AB-string of length s, construct a good integer consisting of no more than $s+10$ digits, encoding the string.

Afterwards, given the good integer, restore the original string.

Interaction Protocol

In this problem, your solution will be run twice on each test.

During the first run, the solution encodes the AB-string. The first line contains the word "`first`". The second line contains an integer s: length of the AB-string ($1 \le s \le 3 \cdot 10^5$). The third line contains a string of length s, consisting of the characters '`A`' and '`B`'.

On the only line of the output, print any good number of length no more than $s+10$.

During the second run, the solution restores the AB-string. The first line contains the word "`second`". The second line contains an integer n, the length of the decimal representation of the good number given. The third line contains a string of length n — decimal representation of the good number. The sequence and the integer is the one printed during the first run.

On the only line of the output, print the restored AB-string.

During each run, each line of input including the last one is terminated by a newline.

Examples

On each test, the input during the second run depends on the solution's output during the first run.

Two runs of some solution on the first test are shown below.

standard input	standard output
first 6 AABBAA	12325
second 5 12325	AABBAA

Problem H. Hoodies

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

There are n schoolpupils in the some summer competitive programming camp. Before the camp starts, the contestants got the hoodies from the administration. Some contestants got green hoody and some got violet hoody, which caused the the rivalry between the green and violet team.

At the beginning of the final contest, all contestants have 0 points and the color of the hoody weared by a contestant is known to everyone. During the contest, q submits happen. After the i-th submit, the score of competitor c_i changes by p_i points.

Each competitor in a green hoody calculates his or her *unhappiness* as the number of competitors in a violet hoody, who, at that point, have strictly more points than him. Calculate and print the sum of the unhappiness of contestants in green hoody after every change in points.

Input

The first line of the input contains two integers n ($1 \le n \le 10^5$ and q ($1 \le q \le 3 \cdot 10^5$) — number of contestants and number of sumits in the log.

The next line contains a string of length n that describes the color of each competitor's hoody. Each character in that word is one of the capital letters 'G' or 'V' indicating the color of the i-th competitor's hoody (green or violet, respectively). You may assume that there is at least one contestant in green hoody and at leaste one contestant in violet hoody.

Then the log of the submits follows: each of q entries describes one submit and contains two integers c_i ($1 \le c_i \le n$) and p_i ($1 \le p_i \le 3 \cdot 10^5$). The maximum possible score in the contest for one contestant is $3 \cdot 10^5$.

Output

In the i-th of the q rows of output, print the total unhappiness after i-th submit.

Examples

standard input	standard output
3 3 VGG 1 15 2 25 3 10	2 1 1
4 6 VGVG 3 72 3 33 2 101 1 13 4 84 1 102	2 2 2 3 2 4

Problem I. Integer Points

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 Mebibytes

Given three integers p, q and n, find the number of points on the Cartesian plane with integer coordinates satisfying

$$0 \le x \le n \text{ and } 0 \le y \le x \cdot \sqrt{\frac{p}{q}}.$$

Input

On the first line of input there is a single integer t, the number of test cases ($1 \le t \le 100\,000$). After that, t lines follow, each containing one test case. Each test case consists of three integers p, q and n separated by single spaces ($1 \le p,\ q \le 100$, $0 \le n \le 10^9$).

Output

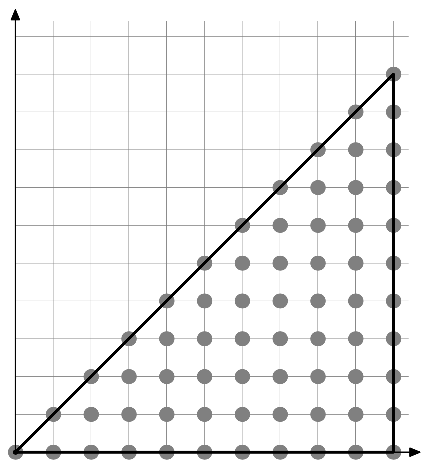
For each test case, your solution should output the number of points with integer coordinates satisfying the constraints on a line by itself.

Example

standard input	standard output

4
1 1 10
2 1 5
1 2 4
8 2 3
	66
	25
	10
	16

Illustration



Problem J. Jury Troubles

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 mebibytes

The qualification round of annual match in figure skating between teams of two famous universities: Camford and Oxbridge is just finished. There are n skaters representing Camford and m skaters representing Oxbridge. k jury members are going to decide who goes to the finals and who leaves competition.

Each jury member submits a vote with two skaters: skater whom the jury member wants to see in finals and other skater, who should leave the competition after the qualification round. Because jury members are from same universities, they are aligned, so those 2 skaters never represent the same university.

The jury member is *satisfied*, if after the announcement of the qualification results, first skater from her/his vote is qualified to the finals, while the second skater is not qualified.

Your task is to create the list of **non-qualified skaters** to satisfy maximum possible number of jury members.

Input

The first line of the input contains one integer n $(1 \le n \le 250)$ — the number of skaters from Camford team. Then n lines follow, each containing the name of the member from Camford team. Name is the non-empty sting of English characters consisting of no more than 10 letters.

Then one line follow containing one integer m $(1 \le m \le 250)$ — the number of skaters from Oxbridge team. Then in m lines the names of the Oxbridge team members are listed in the same format.

Then one line follow containing one integer k $(1 \le k \le 1000)$ — the number of jury members. Then k lines follow, each containing two names — a name of skater, supposed to qualify, and name of skater, supposed to lose. You may assume that each of those names is listed in the list of one team and that for each jury member those teams are distinct. All the names of the skaters from both teams are pairwise distinct.

Output

In the first line, print the integer q that represents the number of contestants who are leaving the competition after the qualification round. In the following q lines print the names of these competitors in any order. If there are multiple possible lists, output any of them.

Examples

standard input	standard output
2 john stanley 1 steven 3 john steven stanley steven steven stanley	1 steven
3 king queen rook 3 knight bishop pawn 5 king pawn bishop king pawn rook queen pawn rook bishop	2 bishop pawn

Problem K. Kate The Robot

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 Mebibytes

Consider labyrinths of size 10×10 square cells. Each cell is either empty or filled with an impenetrable wall. One of the labyrinth cells is chosen as the starting position, another one as the destination.

Kate The Robot is then placed in the labyrinth at the starting position. Kate The Robot can make steps; a single step moves the Robot to an adjacent cell in one of the four cardinal directions. The Robot cannot step into a wall and cannot leave the labyrinth.

Kate The Robot has a program to execute. The program is just a string consisting of the following commands: make a single step to the North (‘`N`’), to the West (‘`W`’), to the South (‘`S`’) or to the East (‘`E`’). The commands are executed one after another in the order they appear in the program. For each command, if it is possible to make a step from the Robot’s current position in the given direction, she makes that step. Otherwise (if there is a wall in that direction, or the step would make the Robot leave the labyrinth) the Robot just ignores the command.

Kate’s task is to pass the labyrinth, that is, to move to the destination cell. If at some point of time the Robot moves to the destination cell, her task is fulfilled. If the Robot has executed all commands given in the program but has not yet been to the destination cell, the Robot fails to complete the task.

The measure of Robot’s accomplishments is the penalty defined as follows. If the task is fulfilled, the penalty is the number of commands executed before the Robot moves to the destination cell for the first time (here, we count both the commands that resulted in making a step and the commands that were ignored). If the Robot fails to pass the labyrinth, the penalty is set to the constant 20 000.

Your task is to write a program for the Robot so that for the certain type of random labyrinths described below, she performs well *on average*, that is, the average penalty does not exceed the given threshold t. Your program should consist of no more than 10 000 commands.

Your program will be checked in the following way. There will be 20 tests. In each test, the input for your solution contains just one integer t. There are also 1000 pre-generated random labyrinths associated with each test, for a total of 20 000 labyrinths; they are however kept secret. After your solution writes a program for the Robot to the output, the checking program executes it on the 1000 pre-generated labyrinths associated with the respective test. If the average penalty for these labyrinths does not exceed the given threshold, your solution passes the test; otherwise, the outcome for the test is “`Wrong Answer`”.

The pre-generated labyrinths are generated by the following algorithm. First, 20 distinct cells to mark on the 10×10 grid are picked randomly. After that follow 500 iterations of trying to put a wall. On each iteration, a random cell on the 10×10 grid is picked. If that cell is empty and not marked, a new wall is put on it. After that, if all the marked cells are still reachable from each other, the new wall is kept; otherwise, it is removed to make the cell empty again.

Once the walls are placed, it is time to choose the starting position and the destination. For each pair of cells reachable from each other, the distance is calculated as the length of the shortest path between them; here, the length of a path is the number of steps in that path. A pair of cells with the longest distance between them is picked randomly. One of the cells of the pair becomes the starting position, and the other becomes the destination.

In the algorithm above, all random picks are independent and have uniform distribution. All labyrinths are generated randomly and independently. Note that your solution does not need to perform well for labyrinths other than generated by this algorithm.

Input

On the first line of input there is a single integer t, the threshold ($1000 \le t \le 1500$). There are exactly 20 tests. It is guaranteed that in test number n, the threshold will be $t = \max(1000, 1550 - n \cdot 50)$. Thus the threshold is 1500 in test 1, 1450 in test 2, 1400 in test 3, . . ., 1050 in test 10 and 1000 in tests 11–20. The first 10 tests might show how well your program does. The last 10 tests are to ensure your program does well indeed.

Note that your solution does not need to use the number given in the input, it is there just for convenience.

Output

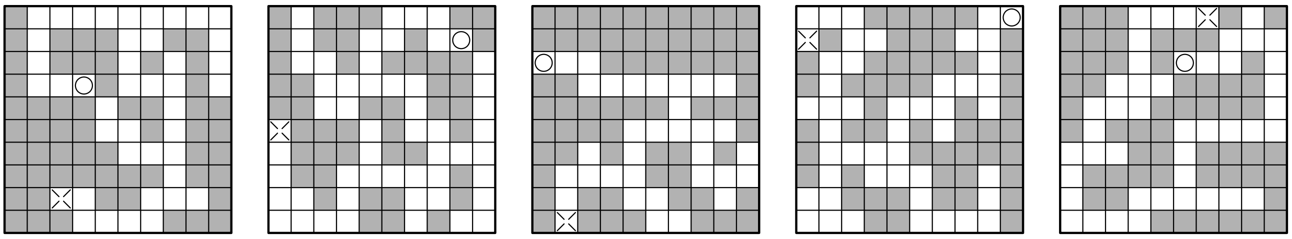
For each test, your solution should output the program for the Robot on a line by itself. Its length should not exceed 10 000 characters. Each character should be either '`N`' for moving to the North, '`W`' for moving to the West, '`S`' for moving to the South, or '`E`' for moving to the East.

In case your solution needs a significant amount of time to perform well, consider precalculating the answer on your local computer and submitting just a program that prints it to the output.

Example

There are no sample tests in this problem.

To illustrate the labyrinth generation algorithm, below are several random labyrinths generated by it.



Problem L. Longest Arc

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

There are n disjunct circles in the coordinate plane. Their convex shell consists of alternating straight segments and circular arcs. Determine the length of the longest of these circular arcs on the edge.

Input

The first line of the input consists of one integer n $(2 \le n \le 1000)$ — the number or circles. j-th of the following n lines contains three integers x_j, y_j and r_j $(-500 \le x_i, y_j \le 500,\ 1 \le r_j \le 500)$ — coordinates of the center and radius of j-th circle. You may assume that no two circles share a common point.

Output

Print the required length of the longest circular arc at the edge of the convex shell with absolute error 10^{-5} or better.

Examples

standard input	standard output

3
1 0 5
1 24 5
1 13 1
	15.707963267949

5
-1 43 2
29 -1 6
9 29 14
33 23 6
5 13 2
	12.469123355359

Problem M. Minimize the Similarity

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Consider the set of N pairwise distinct non-empty words. Each words consists of lowercase English letters. Define the *similarity* between two words as length of their longest common prefix. For example, the similarity between words "`byteland`" and "`bytedance`" is equal to 4, and similarity between words "`ccpc`" and "`icpc`" is equal to 0.

Your task is to order given set in a way such the sum of similarities of all pairs of neighboring words is minimized. If there is more than one such order, choose one lexicographically smallest (first ordering is lexicographically smaller than second if the first word that differs in those orderings, is lexicographically smaller for the first ordering).

Input

First line of the input contains one integer N — number of the words. Each of following N lines contains one non-empty string, consisting of lowercase English letters. You may assume that summary length of all strings does not exceed $4 \cdot 10^5$.

Output

Print the list of words from the input, ordered accordingly to the problem statements, each input string on the new line.

Examples

standard input	standard output
3 bytedance camp contest	camp bytedance contest
4 appeal applause appendix algorithm	appeal algorithm appendix applause
3 z zz zzz	zz z zzz