Hong Kong Olympiad in Informatics
Team Formation Test 2016

Time allowed: 5 hours

Task Overview

ID	Task	Subtasks	Score
T161	Apple Race	$8 + 15 + 8 + 19 + 30 + 20$	100
T162	Avoiding the Crows	$12 + 26 + 14 + 11 + 37$	100
T163	Model Answer II	Partial score available	100
T164	Robos' Feast II	$11 + 23 + 10 + 15 + 41$	100

For all tasks:
CPU time limit: 1 second
Memory limit: 256 MB

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.
- No trailing space(s) in each line.
- No empty lines, except that the input and output should end with the endline character.

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

For some problems 64-bit integers may be required. In Pascal it is int64. In C/C++ it is long long int.

C/C++ programmers should use `"%lld"` for 64-bit integers I/O.

Apple Race

Problem

RB is known as the best competitive programmer with the most exotic diet which is eating salt. Last time, he was satisfied by eating (possibly polluted) salt. This time, he is participating a reality show called “Apple Race”. Here is the description of the race:
The race is held on a perfectly straight road of length N. Competitors start at point 0 and they have to reach point N.

At every point of integral coordinate other than 0, there is a tree by the road. Therefore, there are N trees in total.
There are two types of trees: short and tall. Climbing a short tree requires 1 unit of energy while climbing a tall tree requires 2 units of energy.
On the top of each tree, there is exactly one type of apple which can be described by an integer in $[1, K]$.
After climbing to the top of the tree, competitors can pick one apple from the tree. You may assume that the apples will never run out.
Note that competitors may choose not to climb a tree, which requires no energy.

Competitors only run forward. That is, if one is located at point i, he/she can only go to point $(i + 1)$. There are K checkpoints on the road. The i^{th} checkpoint is located at P_i $(1 \leq P_1 \leq P_2 \leq ... \leq P_K \leq N)$. When a competitor passes checkpoint i, he/she should have collected at least i **different types** of apples, or he/she would lose immediately. Competitors may pick an apple from the tree located at P_i before passing checkpoint i.

RB wants to win the race by reaching point N with the minimum amount of energy spent. As RB wants to taste salty apples very much, please help RB find the winning plan!

Input

The first line of input contains two integers N and K.
The next N lines of input each contains one character (`'S'`: short; `'T'`: tall) and an integer (in $[1, K]$) which specifies the type of apples on the tree at point i.
The next line contains K integers $P_1, P_2, ..., P_K$.

Output

If RB cannot win the race, output `"Lose"` (without quotes).
Otherwise, output according to the following format:
In the first line output two integers, E (the amount of energy used) and C (the number of trees climbed).
In the next line output C integers, the positions of the trees that RB should climb. The positions should be output in increasing order.
If there are several answers of minimum energy spent, output anyone of them.

Sample test

Input	Output
5 2 T 1 T 2 T 1 S 1 T 1 2 5	3 2 2 4

Input	Output
6 3 T 1 T 2 T 3 S 1 S 2 S 3 1 2 3	6 3 1 2 3

Input	Output
6 4 T 1 T 2 T 4 S 1 S 2 S 3 3 5 5 6	5 4 3 4 5 6

Input	Output
6 4 T 1 T 2 T 3 S 1 S 2 S 3 4 5 6 6	Lose

Subtasks

Subtask 1 (8 points)
All trees are short, $1 \leq K \leq 2000$

Subtask 2 (15 points)
$N = K + 1$
$1 \leq K \leq 2000$

Subtask 3 (8 points)
$K = 2$

Subtask 4 (19 points)
$1 \leq K \leq 10$

Subtask 5 (30 points)
$1 \leq K \leq 2000$

Subtask 6 (20 points)
No additional constraints

For all test cases, $1 \leq N \leq 200000$, $1 \leq K \leq 200000$.

Avoiding the Crows

Problem

HKOI-TFT 2016 is going to be held today! One of the enthusiastic contestants, Alice, woke up early in the morning. However, instead of coding again, She decided to relax now so that she will not be nervous and perform badly in the contest.

Therefore, Alice decided to take a walk in the park near her home. The park can be considered as a $N \times M$ grid. The top-left, top-right, bottom-left and bottom-right corner of the grid are denoted as $(1, 1)$, $(1, M)$, $(N, 1)$ and (N, M) respectively. The entrance of the park is located at the top-left corner, whereas the exit of the park is located at the bottom-right corner. Alice is standing at the entrance initially and she wants to reach the exit of the park. In each of her step, she can only walk towards four directions (up, down, left, right) by 1 cell. In other words, if Alice is standing at (x, y), she can walk to $(x - 1, y)$, $(x + 1, y)$, $(x, y - 1)$, $(x, y + 1)$. Undoubtedly, she cannot walk out of the park.

However, Alice doesn't want to take a path from the entrance to the exit causally as there are totally K crows living in some cells in the grid. Alice is superstitious and she believes if she walks too close to the crows, she will be unlucky and fail in the TFT. Therefore, she is going to select her path wisely. Note that cells on her selected path might contain crows as well.

Alice gives a value to each path from cell $(1, 1)$ to cell (N, M) according to the following standard. Firstly, she denotes *cell value* as the Manhattan distance between a cell and any of its closest crow(s). *Path value* is the minimum cell value among the cells visited by Alice along the path.

More precisely, assume the position of the i^{th} crow is (r_i, c_i). The cell value of (x, y) (denoted as $v_{x,y}$) equals to $\min\{|r_i - x| + |c_i - y|\}$ where $1 \le i \le K$. Assume also that Alice will visit the cell (R_j, C_j) during walking along the path of length L. Then the path value will be $\min\{v_{R_j,C_j}\}$ where $1 \le j \le L$.

Please help Alice to find the maximum path value among all paths from the entrance to the exit of the park.

Input

The first line contains 3 integers N, M, K, representing the dimensions of the park and the total number of crows. The following K lines each contains 2 integers, r_i and c_i, representing the position of a crow. There can be more than one crow living in the same cell.

Output

Output the maximum path value among all paths from the entrance to the exit of the park in a single line.

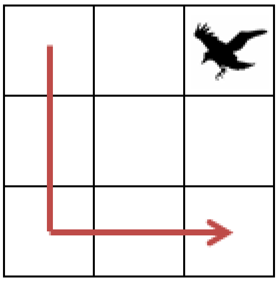
Sample test

Input	Output
3 3 1 1 3	2

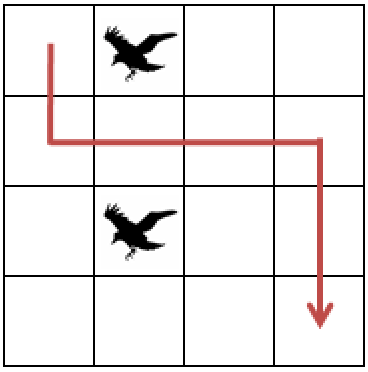
Input	Output
4 4 2 1 2 3 2	1

Explanation

The optimal sequence of moves for Sample 1:



One of the optimal sequences of moves for Sample 2:



Subtasks

Subtask 1 (12 points)
$1 \le N, M \le 10^9, K = 1$

Subtask 2 (26 points)
$1 \le N, M, K \le 200$

Subtask 3 (14 points)
$1 \le N, M, K \le 1000$

Subtask 4 (11 points)
$1 \le N, M, K \le 2000$

Subtask 5 (37 points)
$1 \le N, M \le 10^9, 1 \le K \le 2000$

For all test cases, $1 \le r_i \le N, 1 \le c_i \le M$.

Model Answer II

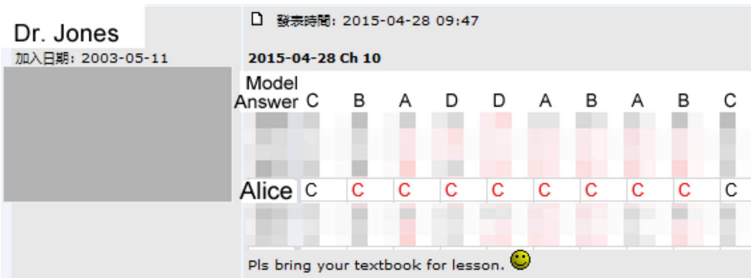
Problem

Last year, Dr. Jones has joined the Byteland Academy as a professor. Recall that he has a robot called Robo, which can scan and mark examination papers for him.

Alice is one of Dr. Jones' students. One day, Dr. Jones made a sudden exam in class. The exam consists of some multiple choice questions from the textbook. Each question has 4 options `A`, `B`, `C`, and `D`. One point will be given for each correct answer and zero points for wrong and invalid answers (no or multiple options are selected). Unfortunately, Alice forgot to bring her textbook, so she had no idea what the questions were.

"When in doubt, try C," Dr. Jones said.

So Alice put C in every answer. Two days later, Dr. Jones posted the scores:



Alice was very sad because it was the first time she failed in an exam.

Today Dr. Jones makes another sudden exam. The test has the same format as the previous one except that this time there are 100,000 questions. The forgettable Alice forgets to bring her textbook again! The desperate Alice is begging Dr. Jones to give her some hints.

"Fine. The hint is ... The answers repeat every 10 questions. In other words, questions 1, 11, 21, ... have the same answer, questions 2, 12, 22, ... have the same answer, and so on."
"That's not helpful, " Alice interrupts, "because the expected score is still 25%!"
"Let me finish first. " Dr. Jones continues. "Here is my marking robot Robo. As you can see, there are 6 buttons `A`, `B`, `C`, `D`, `X`, and `Score`. Enter your answers from question 1 to 100,000 in order using the buttons `A`, `B`, `C`, `D` and `X`. `X` is for invalid answer."
"It won't tell you your score unless you press the `Score` button. You can press the `Score` button to see your current score any time you want, and you can use it more than once. But your final score for the test depends on the number of correct answers (more is better) and the number of times you press the `Score` button (fewer is better)."
"Hey, Bob! You don't have your textbook too? Sit next to Alice and do the exam together." Dr. Jones suggests you.

Task

Write an interactive program to operate Robo. You should implement a procedure `exam` that make calls to grader procedure `answer` and function `score`.

Your procedure `exam()`

Pascal: `procedure exam()`

C/C++: `void exam()`

The grader will call your procedure `exam` once to start the exam. In your `exam` procedure, make exactly 100,000 calls to grader procedure `answer`. Between calls to `answer` you can call grader function `score` to obtain the current score (number of correct answers so far). The procedure should return when you finish the exam.

Grader procedure `answer(option)`

Pascal: `procedure answer(option: char)`

C/C++: `void answer(char option)`

Use this procedure to enter your answer for the next question. Character argument `option` must be one of the capital letters `A`, `B`, `C`, `D`, and `X`. The procedure has no return value.

Grader function `score()`

Pascal: `function score(): longint`

C/C++: `int score()`

Use this function to make Robo display the number of correct answers so far. The function has no parameters. It returns an integer containing your score. Note that there is no need for you to call this function after the last question.

Total CPU time taken by the grader is negligible.

Scoring

You will receive "Wrong Answer" (0 score) if you press the `Score` button for more than 40 times, or if you get fewer than 25,000 (25%) correct answers.

Otherwise, the score x you get for a test depends on the number of `Score` presses p, and number of correct answers s. A calculator is available on the Online Judge.

$$x = \min(100, 36 \times \sqrt[(p+1)^{0.8}]{\tan\frac{(s+80000)\pi}{362500}} - 28)$$

The points you get for this task is the minimum score of all tests, rounded to the nearest 3 d.p.

Hint: It is not necessary to press the `Score` button for the same number of times for every test.
If $p \geq 4$, then $x < 100$ for all $s \leq 100000$.

Sample run

Suppose the answers are `CBADDABABC...`

Function call	Return value	Explanation
`answer(A)` for 10 times		
`score()`	`3`	3 out of 10 questions have answer `A`
`answer(B)` for 10 times		
`score()`	`6`	3 out of 10 questions have answer `B`
`answer(C)` for 10 times		
`score()`	`8`	2 out of 10 questions have answer `C`
`answer(D)` for 10 times		
`score()`	`10`	2 out of 10 questions have answer `D`
`answer(X)` for 10 times		You decided to try button `X`, which is for "invalid answer".
`score()`	`10`	Of course the score does not increase.
`answer(A)` for 99950 times		
(`exam()` returns)		You pressed the `Score` button 5 times and you got $10 + 9995 \times 3 = 29995$ out of 100,000 questions correct. For this test, $p = 5$, $s = 29995$. Therefore the score $x = 11.062$.

Robos' Feast II

Problem

Owing to your generous help last year, about 80% of the Robos were able to know the sum of tastiness of the oil bottles they receive.

RB is the best human competitive programmer. Last year during the Annual Robos' Festival, he was invited to Byteland to join a Robo-human programming competition. Despite the Robos' best effort, RB won convincingly. As a result, he was awarded red, blue, orange, and purple (R, B, O, P) oil bottles. Clearly, the Robos neglected the dietary differences between Robos and humans!

"I don't consume oil," RB said, "I only consume salt."

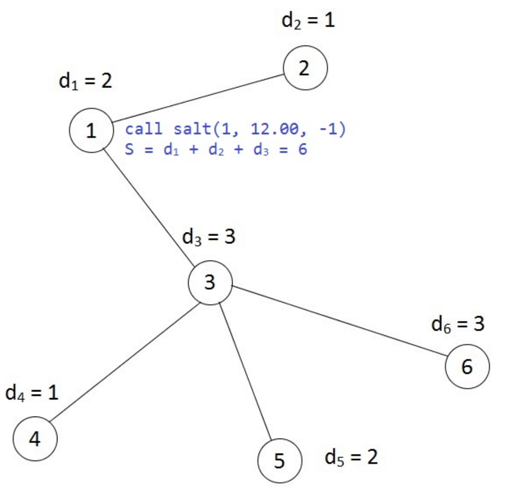
Thus, RB returned home empty-handed. Still, RB was inspired by the Annual Robos' Festival and decided to hold one in Harkerland, where he lives. Harkerland consists of N cities and $N-1$ bidirectional roads. Each road connects a pair of cities. The cities are numbered from 1 to N. It is possible to reach one city from another using one or several roads. In graph theoretic terms, Harkerland is a tree.

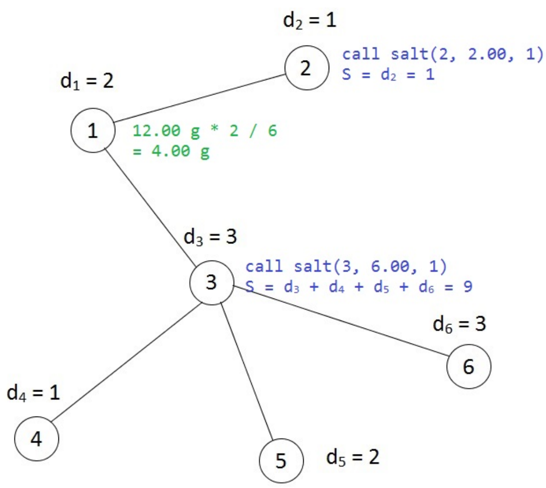
RB wanted to distribute biscuits instead of bottled oil. In order to make biscuits, RB needed to prepare a huge amount of salt and distribute them to the cities. The population density index of city i is d_i, where d_i is an integer ranging from 1 to 10.

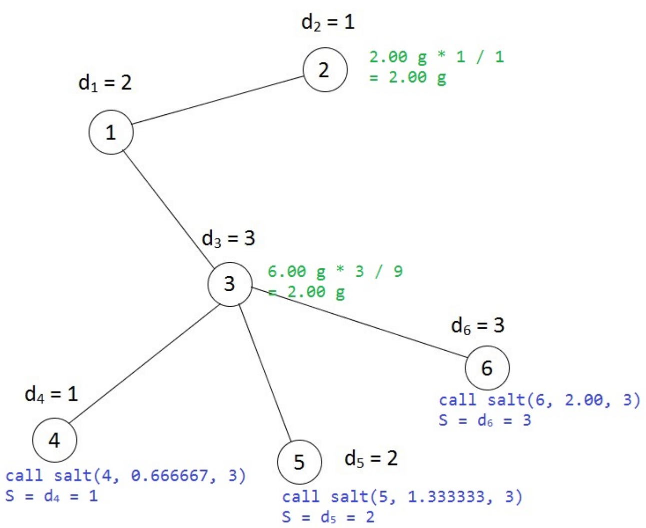
The salt would be given out in M batches and distributed according to a specific algorithm. For each batch, T_i grams of salt would be given out, and a city C_i would be selected. Then, the function `salt(Ci, Ti, -1)` would be called. `salt(C, T, P)` is defined as follows:

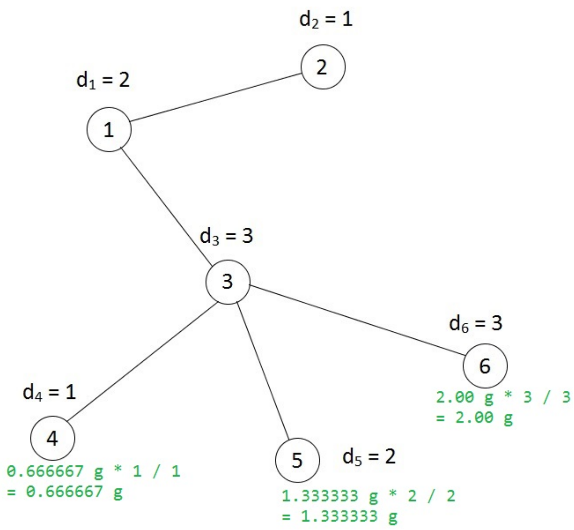
```
// Two cities are said to be neighbours if and only if there is a road connecting them
salt(C, T, P){
  S = dC + sum(di | C and i are neighbours, and i != P)
  give T * (dC / S) grams of salt to city C
  for all nodes i such that C and i are neighbours, and i != P
    call salt(i, T * di / S, C)
}
```

For example, given the graph below, calling `salt(1, 12.00, -1)` would have the following effects:









Today is three days after the first Harkerland Festival. Residents in several cities complain to RB that they did not receive enough biscuits. In order to investigate their complaints, RB decides to find out the amount of salt (in grams) each city received. RB solves this task without difficulty. Can you do the same?

Input

The first line of input contains two integers N and Q.
The second line of input contains N integers $d_1, d_2, ..., d_N$, the population density indices of the N cities.
The next $N - 1$ lines of input each contains a pair of integers A_i and B_i, meaning that the i^{th} road connects cities A_i and B_i.
The next Q lines of input each contains a pair of numbers C_i and T_i, the parameters of the i^{th} call of the salt distribution algorithm. T_i will be given with exactly two decimal places.

Output

Output N lines of non-negative real numbers. In line i, output the total amount of salt (in grams) received by city i.
Let your answer for the i^{th} city be X_i, and the judge's be Y_i. Your answer will be accepted if $\frac{|X_i - Y_i|}{max(1, Y_i)} < 10^{-6}$ for $i = 1, 2, ..., N$.

Sample test

Input	Output
6 1	4.000000
2 1 3 1 2 3	2.000000
1 2	2.000000
1 3	0.666667
3 4	1.333333
3 5	2.000000
3 6	
1 12.00	

Input	Output
6 2	5.333333
2 1 3 1 2 3	2.666667
1 2	5.000000
1 3	1.666667
3 4	3.333333
3 5	5.000000
3 6	
1 12.00	
3 11.00	

Explanation

For the first sample, refer to the task description.

Subtasks

Subtask 1 (11 points)
$1 \leq N, Q \leq 1000$
The i^{th} road connects cities i and $i+1$.

Subtask 2 (23 points)
$1 \leq N, Q \leq 1000$

Subtask 3 (10 points)
The i^{th} road connects cities i and $i+1$.

Subtask 4 (15 points)
The i^{th} road connects cities 1 and $i+1$.

Subtask 5 (41 points)
No additional constraints

For all test cases, $1 \leq N, Q \leq 200000, 0 < T_i \leq 10^9$.