

Problem A. Average Rank

Input file: **standard input**
Output file: **standard output**
Time limit: 5 seconds
Memory limit: 256 megabytes

The National Weekly Escape Room Challenge (NWERC) is a long-running competition held in Eindhoven. Every week a new escape room is presented, and anyone who completes it in their first attempt gains one point.

At the end of each week, competitors are ranked by the total number of points accumulated so far, highest first. In the case of a tie, they share the same rank. In other words, the rank of a competitor is one more than the number of people with a strictly larger number of points.

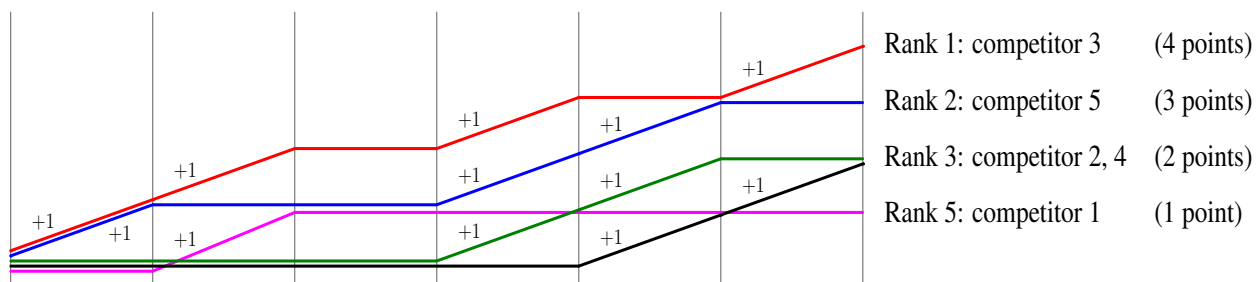


Illustration of Sample Input 3.

In total there have been n participants in the contest, and the contest has been going for w weeks. For each week you are given a list of the competitors that gained a point that week. Your task is to calculate the average rank during the w -week competition for each competitor.

The figure illustrates the score progression in the third sample.

Input

The input consists of:

- One line with two integers n and w ($1 \leq n, w \leq 3 \cdot 10^5$), the number of competitors and the number of weeks. The competitors are numbered from 1 to n .
- w lines (one for each week), each containing an integer k ($0 \leq k \leq n$) followed by k distinct integers $c_1, \dots, c_k$ ($1 \leq c_i \leq n$ for all i), indicating that the k competitors $c_1, \dots, c_k$ each gained a point that week.

The total number of points awarded is at most 1 million.

Output

Output n lines, the i th of which contains the average rank of the i th competitor during the w -week competition. Your answers should have an absolute or relative error of at most 10^{-6} .

Examples

standard input	standard output
3 2	1.000000000
2 1 2	1.500000000
2 1 3	2.500000000
3 1	1.000000000
0	1.000000000
	1.000000000
5 6	3.166666667
2 3 5	3.333333333
2 3 1	1.000000000
0	3.833333333
3 3 5 2	1.666666667
3 5 4 2	
2 3 4	

Problem B. Balanced Cut

Input file: standard input
Output file: standard output
Time limit: 4 seconds
Memory limit: 256 megabytes

Professor Anna van Lier is preparing to give a lecture on balanced binary search trees. Recall that these are binary trees with two properties:

- Balanced tree: For every node, the height of its left subtree and the height of its right subtree differ by at most 1. For instance in the figure, the left and right subtrees of node 7 have heights 2 and 1, respectively. If a node does not have a left (or right) subtree then that subtree is considered to have height 0.
- Search tree: Each node has a value. The value of a node is greater than all the values in the left subtree of the node, and smaller than all the values in the right subtree of the node. For instance in the figure, the left subtree of node 7 contains the values 4, 5 and 6 which are all smaller than 7.

Anna got a picture of such a tree from a colleague. This tree has n nodes with the values 1 to n . However, it turns out to be too big to fit on her slides so she would like to make it smaller. In particular, she would like to erase some nodes from the tree such that it has exactly k remaining nodes. Whenever she erases a node, she also erases the subtrees of that node. Of course, the resulting tree must still be a balanced binary search tree.

For pedagogical purposes, Anna would like the node values in her final tree to be small. Therefore, she wants the list of the k remaining node values to be the lexicographically smallest possible. For example, she would prefer a tree containing values 2, 5, 9 over a tree containing values 2, 6, 7.

As Anna is far too busy doing more important things, the task of finding which nodes to erase falls upon one of her teaching assistants, i.e., you.

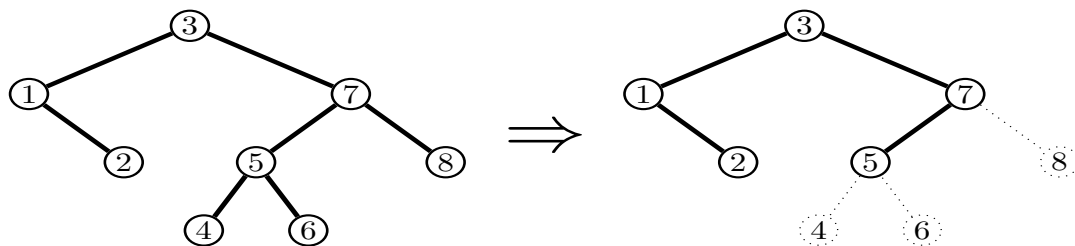


Illustration of Sample Input 2 and its solution.

Input

The input consists of:

- One line with two integers n and k ($2 \leq n \leq 5 \cdot 10^5$, $1 \leq k \leq n - 1$), the number of nodes in the tree and the number of nodes to keep.
- n lines, the i th of which contains an integer p_i ($1 \leq p_i \leq n$ or $p_i = -1$), the parent of the node with value i or -1 if the node with value i is the root.

It is guaranteed that the given tree is a balanced binary search tree.

Output

Output a single line with a binary string of length n . The i th character should be '1' if the node with value i should be kept, and '0' if it should be erased.

Examples

standard input	standard output
3 1 2 -1 2	010
8 5 3 1 -1 5 7 5 3 7	11101010

Problem C. Canvas Line

Input file: **standard input**
Output file: **standard output**
Time limit: 1 second
Memory limit: 256 megabytes

Your friend Charmion asked you to hang some canvases out to dry on a straight washing line for an art project she has been working on. The canvases are artfully arranged such that none of them overlap, although they may touch along the edges. For stability, each canvas must be held by two pegs, but because the canvases are very rigid, they can be held from anywhere.

Each canvas is an integral number of centimetres wide (at least 10 cm). Each peg is slightly less than 1 cm wide. Canvases and pegs are all placed at integral centimetre positions along the line.

Unnecessary things touching any canvas is a smudge risk, thus every canvas should be held by exactly two pegs, no more and no less. Given all of the pegs that are already attached to the line, place as few as possible additional pegs as necessary to hold all of the canvases.

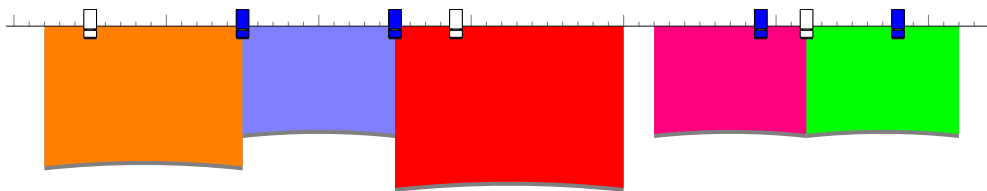


Illustration of a solution to Sample Input 2. Pre-existing pegs are marked in white.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^3$), the number of canvases on the line.
- n lines, the i th of which contains two integers ℓ_i and r_i ($0 \leq \ell_i < r_i \leq 10^9$ and $\ell_i + 10 \leq r_i$), the positions of the left and the right end of the i th canvas in centimetres.
- One line with an integer p ($0 \leq p \leq 2 \cdot 10^3$), the number of pegs already used.
- One line with p integers $x_1, \dots, x_p$ ($0 \leq x_i < x_{i+1} \leq 10^9$ for each i), the position of each existing peg in centimetres.

Canvases are given from left to right and may touch only at edges, that is $r_i \leq \ell_{i+1}$ for each i .

Output

If the canvases can be secured, output the smallest number of extra pegs needed to secure all of the canvases while touching each exactly twice. On the next line output the integer positions of all of the new pegs.

Otherwise, output “impossible”.

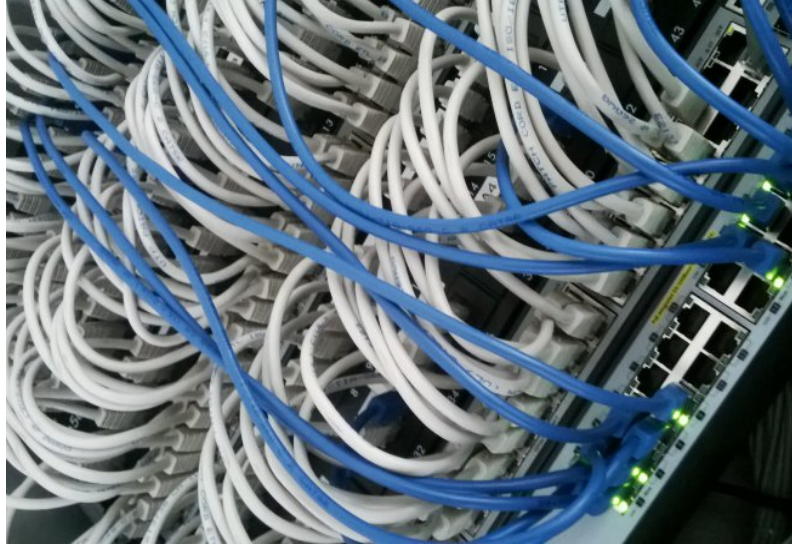
If there are multiple optimal solutions, you may output any one of them.

Examples

standard input	standard output
4 0 18 18 28 28 40 49 60 4 6 12 35 60	3 28 27 59
5 2 15 15 25 25 40 42 52 52 62 3 5 29 52	4 15 25 51 62
3 0 60 60 120 120 140 4 20 60 80 120	impossible

Problem D. Disposable Switches

Input file: standard input
Output file: standard output
Time limit: 4 seconds
Memory limit: 256 megabytes



Picture by Ted Sakshaug on Flickr, cc by

Having recently been hired by Netwerc Industries as a network engineer, your first task is to assess their large and dated office network. After mapping out the entire network, which consists of network switches and cables between them, you get a hunch that, not only are some of the switches redundant, some of them are not used at all! Before presenting this to your boss, you decide to make a program to test your claims.

The data you have gathered consists of the map of the network, as well as the length of each cable. While you do not know the exact time it takes to send a network packet across each cable, you know that it can be calculated as $\ell/v + c$, where

- ℓ is the length of the cable,
- v is the propagation speed of the cable, and
- c is the overhead of transmitting the packet on and off the cable.

You have not been able to measure v and c . The only thing you know about them is that they are real numbers satisfying $v > 0$ and $c \geq 0$, and that they are the same for all cables. You also know that when a network packet is being transmitted from one switch to another, the network routing algorithms will ensure that the packet takes an optimal path—a path that minimises the total transit time.

Given the map of the network and the length of each cable, determine which switches could never possibly be part of an optimal path when transmitting a network packet from switch 1 to switch n , no matter what the values of v and c are.

Input

The input consists of:

- One line with two integers n, m ($2 \leq n \leq 2000$, $1 \leq m \leq 10^4$), the number of switches and the number of cables in the network. The switches are numbered from 1 to n .

- m lines, each with three integers a , b and ℓ ($1 \leq a, b \leq n$, $1 \leq \ell \leq 10^9$), representing a network cable of length ℓ connecting switches a and b .

There is at most one cable connecting a given pair of switches, and no cable connects a switch to itself. It is guaranteed that there exists a path between every pair of switches.

Output

First output a line with an integer k , the number of switches that could never possibly be part of an optimal path when a packet is transmitted from switch 1 to switch n . Then output a line with k integers, the indices of the k unused switches, in increasing order.

Examples

standard input	standard output
7 8 1 2 2 1 3 1 1 4 3 2 6 1 2 7 2 3 5 1 4 7 2 5 7 1	2 4 6
5 6 1 2 2 2 3 2 3 5 2 1 4 3 4 5 3 1 5 6	0
5 6 1 2 2 2 3 1 3 5 2 1 4 3 4 5 3 1 5 6	1 4

Problem E. Expeditious Cubing

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes



The classic Rubik's cube

Picture by Keqs on Wikimedia Commons, cc-by sa

Your friend Claire has dragged you along to a speedcubing event that is happening in Eindhoven. These events are all about solving the Rubik's cube and similar twisty puzzles as quickly as possible. The attendants of the event can enter into various competitions based on the type and size of the puzzle, and there are even special competitions where the puzzles need to be solved one-handed or blindfolded.

Claire is competing in the most popular competition: speedsolving the $3 \times 3 \times 3$ Rubik's cube, pictured on the right. Each contestant needs to solve the cube five times, each time with a different random scramble. After all solves are completed, the best and the worst times are discarded and the final score is the average of the remaining three times. The contestant with the smallest final score wins.

Claire has done well in the competition so far and is among the contenders for the overall victory. All the other contestants have already finished their five solves, but Claire has one solve remaining. By looking at the final scores of the other contestants, she has deduced her own target final score. As long as her final score is less than or equal to this target score, she will be declared the overall winner. Is it possible for her to win the competition, and if so, what is the worst time she can have on her last solve in order to do so?

Input

The input consists of:

- One line with four real numbers t_1 , t_2 , t_3 and t_4 , the times Claire got on her first four solves.
- One line with a real number t , Claire's target final score, the worst final score she can have in order to be declared the overall winner.

Each number is between 1 and 20, inclusive, and is given with exactly two decimal places.

Output

If it is not possible for Claire to win the event, output “**impossible**”. If she will win regardless of the time she gets on her last solve, output “**infinite**”. Otherwise, output the worst time she can have on her last solve in order to be declared the overall winner. Output the number to exactly two decimal places.

Examples

standard input	standard output
6.38 7.20 6.95 8.11 7.53	infinite
6.38 7.20 6.95 8.11 6.99	6.82
6.38 7.20 6.95 8.11 6.45	impossible

Problem F. Firetrucks Are Red

Input file: standard input
Output file: standard output
Time limit: 5 seconds
Memory limit: 256 megabytes



Lily is fascinated by numbers. She believes the whole world revolves around them, and that everything is connected by numbers. Her friends, Alice, Bob, Charlie and Diane, are not convinced. But she gives them an example:

“Alice lives in house number 25 on her street, but that is exactly Bob’s age. Bob is born on June 4th, and Charlie was his parents’ fourth child. Finally, Diane has five fingers on her left hand, which happens to be the same as the number of toes that Bob has on his right foot!”

This shows that her friends are all connected—either directly or indirectly—by numbers. But she still has to convince her family as well as her coworkers.

Given a group of n individuals, and a set of numbers that describe each individual, help Lily come up with a proof that shows that everyone in this group is either directly or indirectly connected by numbers, or determine that this is not possible.

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 2 \cdot 10^5$), the number of individuals in the group. The individuals are numbered from 1 to n .
- n lines, describing the individuals in the group.

The i th such line starts with an integer m_i ($1 \leq m_i \leq 2 \cdot 10^5$), the number of numbers that describe individual i .

The remainder of the line has m_i distinct integers $d_{i,1}, \dots, d_{i,m_i}$ ($1 \leq d_{i,j} \leq 10^9$ for each j), the set of numbers that describe individual i .

It is guaranteed that the sum over all m_i is at most $2 \cdot 10^5$.

Output

Output a proof in the form of $n - 1$ lines, each of which contains three integers p , q and r , where p and q are distinct individuals that are both described by the number r . Using only these relations, it must be possible to show that any pair of individuals in the group are connected either directly or indirectly.

If no such proof exists, output “impossible”. If there are multiple proofs, you may output any one of them.

Examples

standard input	standard output
6 2 17 10 1 5 2 10 22 3 17 22 9 2 17 8 3 9 22 16	impossible
6 2 17 10 2 5 10 2 10 22 3 17 22 9 2 17 8 3 9 22 16	1 4 17 4 3 22 3 2 10 4 6 22 1 5 17

Problem G. Gnoll Hypothesis

Input file: **standard input**
Output file: **standard output**
Time limit: **2 seconds**
Memory limit: **256 megabytes**

You are a huge fan of the RPG The Eldest Scrolls: Earthrim and know all of the game's internal mechanics. For instance, when spawning a new monster, each of the n different types of monsters in the game has some fixed probability of appearing, and you know exactly what this probability distribution over monster types is.

However, in the latest update the developers seem to have changed the way monsters are spawned. After some testing and reverse engineering, you realise that instead of spawning all n types of monsters, each spawn location only has a spawn pool of k monster types. These spawn pools are chosen randomly at the start of the game, independently for each spawn location, with every monster type having the same chance of being chosen for the spawn pool. And apparently a developer was lazy with adjusting the spawn chances. Instead of normalising the spawn chances of the k chosen types, the developer decided that if a type of monster is not chosen, its spawn chance is added to the next chosen monster type in the list of types (and if monster types at the end of the list are not chosen, their spawn chances are added to the first chosen monster type in the list). For example, the figure shows a small example with $n = 5$ monsters, a possible random choice of $k = 3$ of those monsters, and the resulting spawn probabilities for those 3 monsters.

After the update, some monster types seem to appear less often than before, and some more often (for instance now there seem to be Gnolls all over the place). You believe that the new spawning logic may be responsible for this by having changed the effective spawn chances of the monsters. In order to test this hypothesis, you decide to compute these effective spawn chances after the update.

1	1%	1	24%
2	25%	2	25%
3	39%	3	0%
4	12%	4	51%
5	23%	5	0%

Sample Input 1 and one possible spawn pool with adjusted spawn chances.

Input

The input consists of:

- One line with two integers n and k ($1 \leq k \leq n \leq 500$), the number of different types of monsters and the number of monsters that are randomly chosen for the spawn pool of each spawn location.
- One line with n real numbers $s_1, s_2, \dots, s_n$ ($s_i \geq 0$ for each i , $\sum_{j=1}^n s_j = 100$), where s_i is the spawn chance in percent for the i th type in the list of monster types. Every real number has at most six digits after the decimal point.

Output

Output a single line containing n real numbers, the effective spawn chance in percent of each type of monster. The i th number in your output should correspond to the i th type of monster. Your answers should have an absolute or relative error of at most 10^{-6} .

Examples

standard input	standard output
5 3 1 25 39 12 23	8.7 17.6 31 21.4 21.3
3 2 2.019 87.51234 10.46866	4.835553333333 59.01456 36.14988666667

Problem H. Height Profile

Input file: **standard input**
Output file: **standard output**
Time limit: 6 seconds
Memory limit: 256 megabytes

The cycling classic Amstel Gold Race is held annually in the nearby Dutch province of Limburg. It features frequent short climbs giving contestants little time to recover in between. For instance the last 42 kilometres of the race has 8 steep climbs of average length roughly 1 kilometre each.

The incline grade for each climb is given in percent. An incline grade of 100% means that for every 1 horizontal metre travelled, you also travel 1 vertical metre upwards. An incline grade of 0% is perfectly flat. In general, an incline grade of $p\%$ has $p/100$ vertical metres for each horizontal metre. Note that the incline grade can also be negative.

You know the height of the road at integer horizontal kilometres from the start of the race, and for simplicity we model the heights as being piecewise-linear in between these points. In other words, the incline grade is assumed to be constant between 0 and 1 km, and between 1 and 2 km, and so on. For example, in the figure, the height at the horizontal distance 2.5 kilometres is exactly 20 metres.

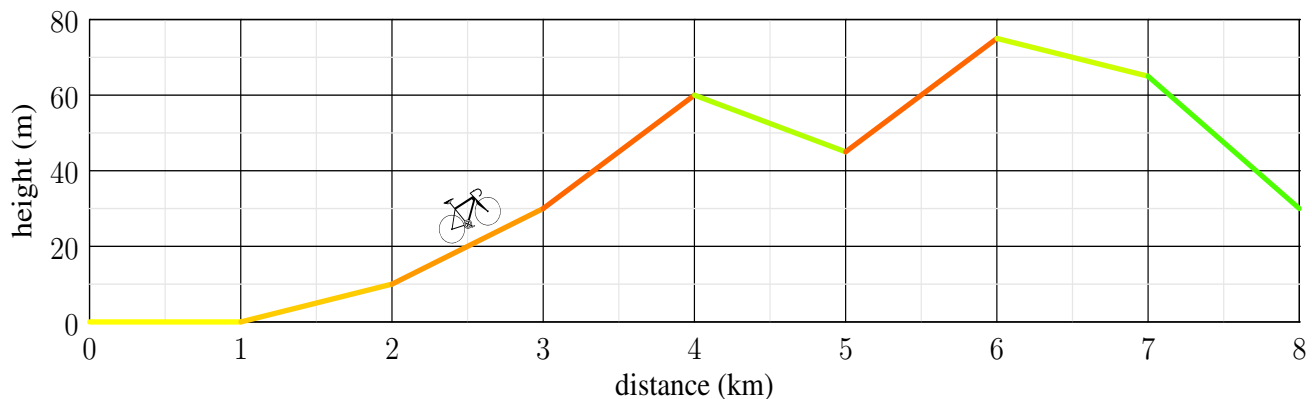


Illustration of Sample Input 1. The incline grades of the first 3 horizontal kilometres are in order 0%, 1% and 2%. The last kilometer has an incline grade of -3.5% .

The Amstel Gold Race cannot be compared with some hilly stages of the Tour de France or the Giro d'Italia, but you are still interested in a comparison. For each one-day race, whether it is the Amstel Gold Race or one of the stages of the Tour de France, you would like to know the length of the longest horizontal interval with at least a given incline grade. Actually, you would like to know the answer for multiple incline grades.

The incline grade of a horizontal interval is measured between the two endpoints. For example, the incline grade between kilometers 1 and 4 of the race in the figure is 2%, as 60 vertical metres are gained in 2 kilometres. This horizontal interval is also the longest one with an incline grade of at least 2%.

Input

The input consists of:

- One line with two integers n and k ($2 \leq n \leq 10^5, 1 \leq k \leq 50$), the horizontal length of the race in kilometres and the number of incline grades you have chosen.
- One line with $n + 1$ integers $h_0, h_1, \dots, h_n$ ($0 \leq h_i \leq 10^9$), where h_i is the height of the route in metres i horizontal kilometres from the start.
- k lines each containing a real number g ($-100 \leq g \leq 100$ and g has exactly one digit after the decimal point), an incline grade you care about.

Output

For each of the k given incline grades, in the same order as in the input, output the length in kilometres of the longest horizontal interval with at least this incline grade. If no suitable interval exists, output “-1”.

Your answers should have an absolute or relative error of at most 10^{-6} .

Examples

standard input	standard output
8 2 0 0 10 30 60 45 75 65 30 2.0 3.1	3 -1
2 2 0 30 30 3.0 2.0	1 1.5

Problem I. Inverted Deck

Input file: `standard input`
Output file: `standard output`
Time limit: 5 seconds
Memory limit: 256 megabytes



Elf by LadyofHats via Wikimedia Commons, public domain

As a huge fan of the popular collectible card game *Numinous Wilds: the Elven Reign Chronicles* (NWERC), you have a large collection of cards which you carefully organise by their rarity. One day you notice that someone has touched your collection, and that some of the cards are now out of order. The most natural suspect, of course, is your little brother Billy, who was absolutely 100% forbidden from playing with your cards. After a few minutes of interrogation Billy confesses that he indeed took a few consecutive cards from the middle of the stack, but he swears that he put them back in exactly the same order as they were. You suspect that Billy, being so young, may have simply mistakenly reversed the order of the cards that he took. Now you want to check your theory and decide if you can find the batch of cards that Billy took to play with.

Is it possible to restore the order of the cards into non-decreasing order of their rarity by reversing just one contiguous batch of cards?

Input

The input consists of:

- One line containing an integer n ($1 \leq n \leq 10^6$), the number of cards in your collection.
- One line containing n integers $v_1, \dots, v_n$ ($1 \leq v_i \leq 10^9$ for all i), the current order of the cards' rarity values.

Output

If the cards can be sorted by reversing exactly one contiguous subsequence of the list, then output the 1-based start and end indices of such a subsequence. Otherwise, output **impossible**. If there are multiple valid solutions you may output any one of them.

Examples

standard input	standard output
7 10 13 19 19 15 14 20	3 6
6 9 1 8 2 7 3	impossible
3 1 2 3	1 1

Problem J. Jackdaws And Crows

Input file: standard input
Output file: standard output
Time limit: 6 seconds
Memory limit: 256 megabytes



A Clattering of Jackdaws by Dun.can on flickr, cc by

Nick is a bird watcher and often visits the forum “CrowFinders” to discuss his hobby with like-minded people. CrowFinders has a voting system where users can upvote or downvote comments, which increases or decreases their score by 1. This means that each comment can end up with any integer score (including negative scores). Once when Nick was browsing a heated discussion about the classification of jackdaws as crows, he found something very pleasing: a chain of comments that alternated between positive and negative scores. But a few days later, he found that the comment chain was no longer alternating. Now Nick wants to make it alternating again.

A comment chain is alternating if the scores $s_1, s_2, \dots, s_n$ of the comments all are non-zero, and every pair of adjacent scores s_i, s_{i+1} have opposite signs. In particular, a single comment with a non-zero score or even a comment chain without any comment is an alternating comment chain.

There are two operations Nick can do to make the comment chain alternating:

1. Create a fake account and upvote/downvote some of the comments. This increases/decreases their respective scores by 1. Each fake account can only upvote/downvote each comment at most once, but it can vote on any subset of the comments. It takes c seconds to create an account and use it to vote (regardless of how many comments are upvoted/downvoted).
2. Report one specific comment to remove it from the chain. Thinking of convincing reasons for the report takes r seconds. (Nick is an excellent arguer, so once the report is filed, the comment is guaranteed to be removed.)

Nick can apply these operations in any order, any number of times. How fast can he make the comment chain alternating?

For example, consider Sample Input 1 below, where the scores in the comment chain are 8, 8, 2, -2, and it takes Nick 10 seconds to create an account and 50 seconds to file a report for one comment. In this case

it is optimal to first create 3 fake accounts and use them to upvote the fourth comment and downvote the third, followed by reporting the first comment. This results in the scores 8, -1, 1, which is an alternating chain. The time used for this is 80 seconds.

Input

The input consists of:

- One line with three integers n , c , and r ($1 \leq n \leq 5 \cdot 10^5$, $1 \leq c, r \leq 10^9$), the number of comments in the chain, the time it takes to create a fake account and the time it takes to report one comment respectively.
- One line with n integers $s_1, \dots, s_n$ ($-10^9 \leq s_i \leq 10^9$ for all i), the current score of each comment in the chain.

Output

Output the smallest time to make the comment chain alternating by applying the operations above.

Examples

standard input	standard output
4 10 50 8 8 2 -2	80
6 100 33 5 -13 0 0 -12 0	132

Problem K. Kitesurfing

Input file: standard input
Output file: standard output
Time limit: 4 seconds
Memory limit: 256 megabytes

Nora the kitesurfer is taking part in a race across the Frisian islands, a very long and thin archipelago in the north of the Netherlands. The race takes place on the water and follows a straight line from start to finish. Any islands on the route must be jumped over – it is not allowed to surf around them.

The length of the race is s metres and the archipelago consists of a number of non-intersecting intervals between start and finish line. During the race, Nora can move in two different ways:

1. Nora can surf between any two points at a speed of 1 metre per second, provided there are no islands between them.
2. Nora can jump between any two points if they are at most d metres apart and neither of them is on an island. A jump always takes t seconds, regardless of distance covered.

While it is not possible to land on or surf across the islands, it is still allowed to visit the end points of any island.

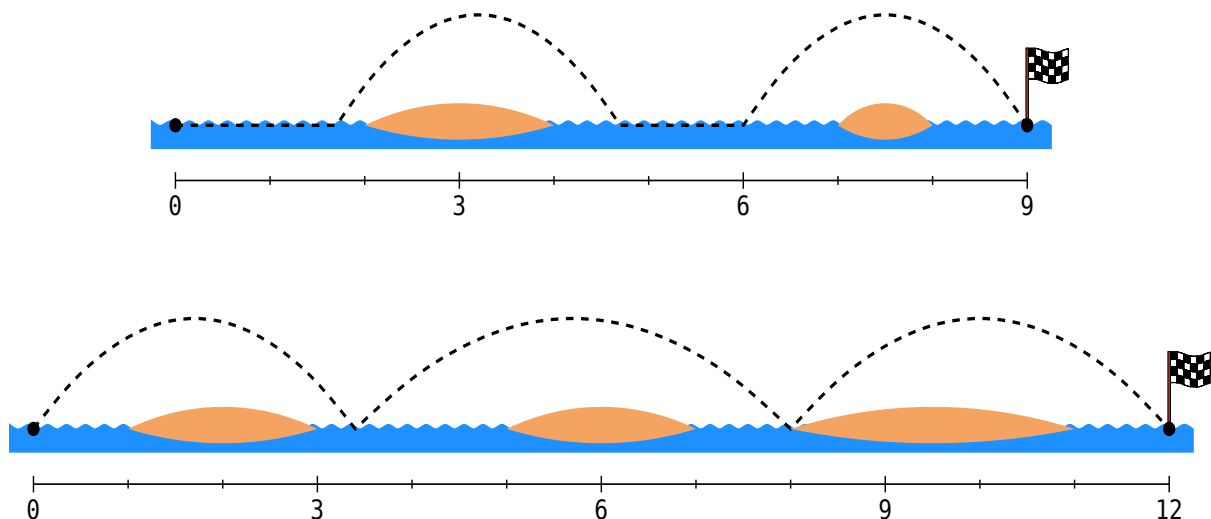


Illustration of the two sample cases.

Your task is to find the shortest possible time Nora can complete the race in. You may assume that no island is more than d metres long. In other words it is always possible to finish the race.

Input

The input consists of:

- One line with three integers s, d and t ($1 \leq s, d, t \leq 10^9$), where s is the length of the race in metres, d is the maximal jump distance in metres, and t is the time needed for each jump in seconds.
- One line with an integer n ($0 \leq n \leq 500$), the number of islands.
- n lines, the i th of which contains two integers ℓ_i and r_i ($0 < \ell_i < r_i < s$ and $r_i - \ell_i \leq d$), giving the boundaries of the i th island in metres, relative to the starting point.

The islands do not touch and are given from left to right, that is $r_i < \ell_{i+1}$ for each valid i .

Output

Output one number, the shortest possible time in seconds needed to complete the race. It can be shown that this number is always an integer.

Examples

standard input	standard output
9 3 4 2 2 4 7 8	11
12 5 3 3 1 3 5 7 8 11	9