Problem A. Ascending Photo

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

The members of the No-Weather-too-Extreme Recreational Climbing society completed their 100th successful summit today! To commemorate the occasion, we took a picture of all the members standing together in one row, to use for marketing purposes.

However, the photograph looks messy; as usual, the members refused to order themselves in any kind of aesthetically pleasing way. We will need to reorder the picture.

Figure 1: This picture has been cut up and pasted back together to solve Sample Input 1.

Our research tells us that having the climbers in ascending (non-decreasing) height order from left to right will be most visually appealing. We must cut up the picture we have and somehow paste it back together in this order.

Find the minimum number of cuts you need to make to put the photograph into ascending order.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^6$), the number of people in the photo.
- One line with n integers $h_1, \ldots, h_n$ ($1 \leq h_i \leq 2 \cdot 10^9$ for each i), the heights of the people in the photograph, from left to right.

Output

Output the minimum number of cuts needed to rearrange the photograph into any one ascending (non-decreasing) height order from left to right.

Example

standard input	standard output
11 3 6 12 7 7 7 7 8 10 5 5	4
3 5000000 5500000 7000000	0
12 1 2 2 3 3 1 2 3 4 1 2 3	6

Problem B. Boss Battle

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

You are stuck at a boss level of your favourite video game. The boss battle happens in a circular room with n indestructible pillars arranged evenly around the room. The boss hides behind an unknown pillar. Then the two of you proceed in turns.

- First, in your turn, you can throw a bomb past one of the pillars. The bomb will defeat the boss if it is behind that pillar, or either of the adjacent pillars.
- Next, if the boss was not defeated, it may either stay where it is, or use its turn to move to a pillar that is adjacent to its current position. With the smoke of the explosion you cannot see this movement.

The last time you tried to beat the boss you failed because you ran out of bombs. This time you want to gather enough bombs to make sure that whatever the boss does you will be able to beat it. What is the minimum number of bombs you need in order to defeat the boss in the worst case? See Figure 2 for an example.

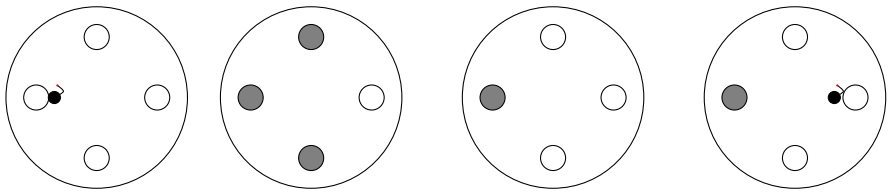


Figure 2: Example for $n = 4$. In this case 2 bombs are enough. Grey pillars represent pillars where the boss cannot be hiding. The bomb is represented in black.

Input

The input consists of:

- One line with a single integer n ($1 \leq n \leq 100$), the number of pillars in the room.

Output

Output the minimum number of bombs needed to defeat the boss in the worst case.

Example

standard input	standard output
4	2
7	5

Problem C. Connect the Dots

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

A famous logical problem is that of connecting 9 dots on a paper by drawing 4 line segments with a pencil, while never lifting the pencil from the paper. While this is easy enough (although it requires some thinking outside of the box), Simone has recently been building a game called "Connect the Dots" around a generalisation of the concept.

In Connect the Dots, you are presented with a 4×4 regular grid of dots. Each dot is given a unique number between 1 and 16. The task is then to connect the dots in order by their numbers, starting with dot 1 and ending with dot 16. The dots should be connected using *as few line segments as possible*, starting at dot 1, with the end of each segment being the start point of the next. The segments are allowed to intersect and overlap one another. Additionally, it is allowed to pass through other points while trying to connect the current point. This means, for example, that visiting the first four points in the order $1, 4, 2, 3, 2, 4, \ldots$ is acceptable. Formally, the sequence $1, 2, \ldots, 15, 16$ must be a subsequence of the sequence of dots visited.

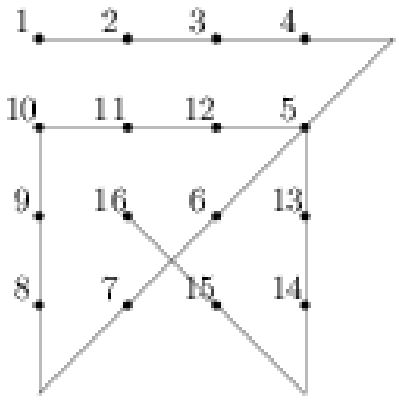


Figure 3: A solution to the first sample.

Simone asked you to try the puzzle out, while betting you a balloon that it would be too hard. Prove her wrong by writing a program that solves the puzzle for you!

Input

The input consists of:

- 4 lines, each with 4 integers, the numbers of the dots in the grid. The jth number on the ith line is the number of the jth dot in the ith row of the grid of dots.

The 16 numbers in the input are all between 1 and 16 (inclusive) and pairwise distinct.

Output

Output the minimum number of line segments needed to connect all the dots in order.

Example

standard input	standard output
1 2 3 4 10 11 12 5 9 16 6 13 8 7 15 14	6
1 2 3 4 8 9 10 11 7 15 16 12 6 14 13 5	7

Problem D. Dunglish

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

A confused Dutchman trying to speak English could say "*I am in the war*", even though there is no hostile activity going on. The confusion[1] here is that the English sentence "*I am confused*" is translated in Dutch as "*Ik ben in de war*", which is phonetically ("sounding") quite close to the first sentence. Such confusion leads to much enjoyment, but can complicate matters a bit.

Given a sentence in Dutch and a dictionary containing both correct translations as well as phonetic (incorrect) translations of individual words, find the translation of the sentence and indicate whether it is correct, or in case there is more than one find the total number of correct and incorrect translations. A sentence is correctly translated when each word of the sentence is correctly translated.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 20$), the number of words in the Dutch sentence.
- One line with n words, the Dutch sentence s.
- One line with an integer m ($1 \leq m \leq 10^5$), the number of words in the dictionary.
- m lines, each with three strings d, e and c, a Dutch word, the English translation, and "`correct`" if this is the correct translation or "`incorrect`" otherwise.

A word consists of between 1 and 20 lowercase letters. Each word in s appears at least once as a Dutch word in the dictionary, no word appears more than 8 times as a Dutch word in the dictionary, and each combination of a Dutch and English word appears at most once.

Output

In case there is only a single translation of s, output one line with the translation followed by one line with "`correct`" or "`incorrect`". In case there is more than one translation, output one line with the number of possible correct translations followed by "`correct`", and one line with the number of possible incorrect translations followed by "`incorrect`".

[1] Pun intended.

Example

standard input	standard output
7 als mollen mollen mollen mollen mollen mollen 4 als when correct mollen moles correct mollen destroy correct mollen mills incorrect	64 correct 665 incorrect
5 de zuigers zijn buiten werking 6 zijn are correct banaan banana correct de the correct zuigers suckers incorrect buiten out correct werking working incorrect	the suckers are out working incorrect

Problem E. English Restaurant

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

After finding a suitable residence in the Swiss Alps, Robin thought he finally had what it takes to be happy. But every day he woke up feeling that something was missing. Maybe it was the lack of English food in these parts? Spotting a market opportunity and solving his problem at the same time, he teamed up with the famous English chef Jim to open a restaurant nearby. While he has no doubts about Jim's talents as a chef, Robin wants to be sure that opening an English restaurant in the Swiss Alps is a good idea.

Fortunately, as a local, he knows the habits of restaurant customers. At the stroke of each hour, a group of people arrives of size that is uniformly random between 1 and g people (inclusive). The group finds the smallest completely unoccupied table that fits the group and occupies it. If they can not find such a table, the group leaves with great disappointment. Once seated, a group never leaves until the restaurant closes, as Jim has no difficulty keeping the guests entertained.

As an example, suppose the restaurant has 3 tables of capacities 5, 8 and 9. If groups of sizes $5, 10$ and 3 arrive (in that order), in the end there will be 8 people in the restaurant. The first group occupies the table of capacity 5, the second group leaves and the last group occupies the table of capacity 8.

Robin plans to keep his restaurant open for t hours in total. In the restaurant business the most important metric is the expected number of people in the restaurant when it is closes. Can you help Robin calculate the expected occupancy after t hours?

Input

The input consists of:

- One line with three integers n, g, t ($1 \leq n \leq 100, 1 \leq g \leq 200, 1 \leq t \leq 100$), the number of tables in the restaurant, the maximum group size, and the number of hours the restaurant is open.
- One line with n integers $c_1, \ldots, c_n$ ($1 \leq c_i \leq 200$ for each i) giving the capacities of the tables.

Output

Output the expected number of people in the restaurant when it closes. Your answer should have an absolute or relative error of at most 10^{-6}.

Example

standard input	standard output
3 3 2 1 2 3	3.666666667
4 11 4 10 10 10 10	20.000000000
4 3 3 4 1 3 2	5.888888888888

Problem F. Factor-Free Tree

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

A *factor-free tree* is a rooted binary tree where every node in the tree contains a positive integer value that is coprime with all of the values of its ancestors. Two positive integers are coprime if their greatest common divisor equals 1.

The *inorder sequence* of a rooted binary tree can be generated recursively by traversing first the left subtree, then the root, then the right subtree. See Figure 4 below for the inorder sequence of one factor-free tree.

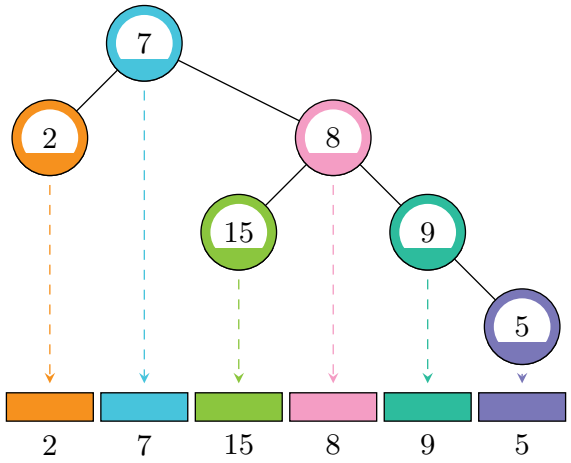


Figure 4: Illustration of Sample 1. The tree is factor-free; for example, the value of the node marked "5" is coprime with all of the values of its ancestors, marked "9", "8", and "7".

Given a sequence $a_1, a_2, \ldots, a_n$, decide if it is the inorder sequence of some factor-free tree and if so construct such a tree.

Input

The input consists of:

- One line with one integer n ($1 \leq n \leq 10^6$), the length of the sequence.
- One line with n integers $a_1, \ldots, a_n$ ($1 \leq a_i \leq 10^7$ for each i), the elements of the sequence.

Output

If there exists a factor-free tree whose inorder sequence is the given sequence, output n values. For each value in the sequence, give the 1-based index of its parent, or 0 if it is the root. If there are multiple valid answers, print any one of them.

If no such tree exists, output "`impossible`" instead.

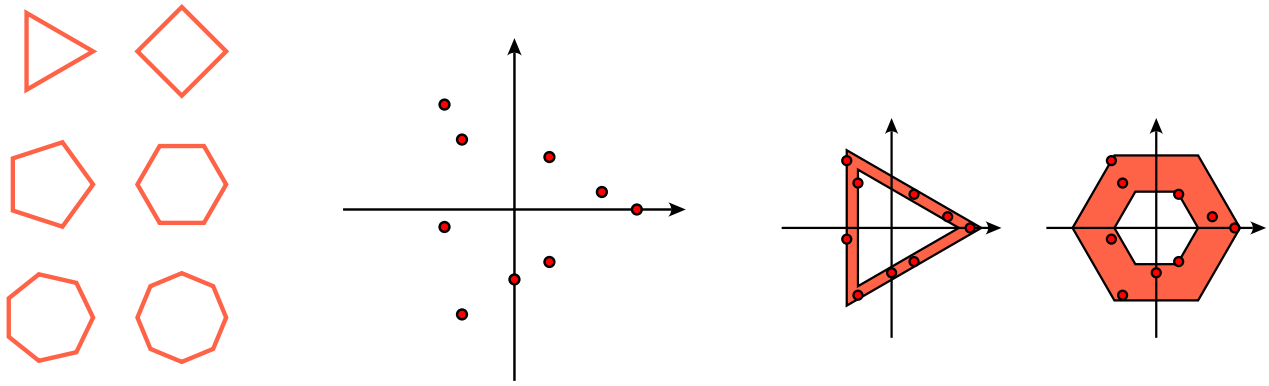
Example

standard input	standard output
6 2 7 15 8 9 5	2 0 4 2 4 5
6 2 7 15 8 9 6	impossible

Problem G. Glyph Recognition

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

You are an archaeologist working at an excavation site where your team has found hundreds of clay tablets containing glyphs written in some ancient language. Not much is known about the language yet, but you know that there are only six different glyphs, each of them in the shape of a regular polygon with one vertex pointing to the right (see Figure 5a below). Only the boundary of each polygon is carved out of the clay.



(a) The six glyphs.

(b) The first sample input.

(c) Fitting triangles and hexagons to the first sample. The triangles' score is higher.

Figure 5

You want to start analysing the language right away, so you need to get the text on the tablets into some machine readable format. Ideally, you would like to use an OCR (optical character recognition) tool for that, but you do not have one installed on your laptop and there is no internet connection at the site.

Because of this you have devised your own scheme to digitise the ancient writings: for every glyph on a tablet you first find a number of sample points that are in the carved out region, i.e. on the boundary of the polygon. Based on those sample points you then calculate a score for each of the six glyphs and mark the one with the highest score as the recognised glyph.

For a given number of corners k ($3 \leq k \leq 8$), the score is computed as follows. Two regular k-gons are fitted to the sample points, one from the inside and one from the outside, such that the following hold:

- Each polygon is centered at the origin, i.e. all vertices have equal distance to $(0, 0)$.
- Each polygon has a vertex on the positive x-axis.
- The inner polygon is the largest such polygon containing none of the sample points.
- The outer polygon is the smallest such polygon containing all of the sample points.

An example can be seen in Figure 5c. The score for this value of k is $\frac{A_{\text{inner}}}{A_{\text{outer}}}$, where A_{inner} and A_{outer} are the areas of the inner and outer polygon, respectively.

Given a set of sample points, find the glyph with the highest score.

Input

The input consists of:

- One line with one integer n ($1 \leq n \leq 1\,000$), the number of sample points.
- n lines, each with two integers x, y ($-10^6 \leq x, y \leq 10^6$), specifying a point at coordinates (x, y).

No sample point is at the origin and all points are distinct.

Output

Output the optimal number of corners k ($3 \leq k \leq 8$), followed by the score obtained for that value of k. Your answer will be accepted if the absolute error does not exceed 10^{-6}. If several values of k result in a score that is within 10^{-6} of the optimal score, any one of them will be accepted.

Example

standard input	standard output
9 -4 -1 -4 6 -3 -6 -3 4 0 -4 2 -3 2 3 5 1 7 0	3 0.5625000000
4 1 0 0 1 -1 0 0 -1	8 1.0000000000

Problem H. High Score

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

Mrten and Simon enjoy playing the popular board game Seven Wonders, and have just finished a match. It is now time to tally the scores.

One of the ways to score in Seven Wonders is through the use of *Science*. During the game, the players may collect a number of Science tokens of three different types: *Cog*, *Tablet*, and *Compass*. If a player has a Cogs, b Tablets and c Compasses, that player gets $a^2 + b^2 + c^2 + 7 \cdot \min(a, b, c)$ points.

However, the scoring is complicated by the concept of *Wildcard Science* tokens. For each Wildcard Science token a player has, she may count that as one of the three ordinary types of Science tokens. For instance, the first player in Sample Input 1 has 2 Cogs, 1 Tablet, 2 Compasses, and 1 Wildcard Science, so could thus choose to have the distributions $(3, 1, 2), (2, 2, 2)$ or $(2, 1, 3)$ of Cogs, Tablets and Compasses, respectively. The possible scores for this player are then $3^2 + 1^2 + 2^2 + 7 \cdot 1 = 21$, $2^2 + 2^2 + 2^2 + 7 \cdot 2 = 26$ and $2^2 + 1^2 + 3^2 + 7 \cdot 1 = 21$ depending on how the Wildcard Science is assigned. Thus, the maximum score for this player is 26.

Given the number of tokens each player in the game has, compute the maximum possible score that each of them can achieve if they assign their Wildcard Science tokens optimally.

Input

The input consists of:

- One line with an integer n ($3 \leq n \leq 7$), the number of players in the game.
- n lines, each with four integers a, b, c, d ($0 \leq a, b, c, d \leq 10^9$), giving the number of Cog, Tablet, Compass, and Wildcard Science tokens of a player.

Output

For each player, in the same order they are given in the input, output the maximum score the player may get.

Example

standard input	standard output
3 2 1 2 1 3 2 1 0 1 3 0 1	26 21 18

Problem I. Installing Apps

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

Sandra recently bought her first smart phone. One of her friends suggested a long list of applications (more commonly known as "apps") that she should install on the phone. Sandra immediately started installing the apps from the list, but after installing a few, the phone did not have enough disk space to install any more apps. Sometimes, the app installation failed because there was not even enough space to download the installation package. Other apps could be downloaded just fine, but had insufficient space to store the installed app.

Each app that Sandra installs has a download size d and a storage size s. To download the app, Sandra's phone must have at least d megabytes of free disk space. After the app has been installed, it then uses s megabytes of disk space on the phone. The download size may be smaller than the storage size (e.g., if the app data is heavily compressed) or larger than the storage size (e.g., if the download contains material that might not get used such as translations to different languages). The installer is very efficient and can transform the downloaded package to an installed app without using any extra disk space. Thus, to install an app, the phone must have at least $\max(d, s)$ megabytes of free disk space.

Sandra quickly realised that she may have run out of space just because she installed apps in the wrong order. Thus, she decided to give the installation another try. She uninstalled all apps, and will now choose an installation order that lets her install the largest number of apps from the list. Sandra may not install any app more than once.

Help her determine what apps on the list she should install, and in what order.

Input

The input consists of:

- One line with two integers n, c ($1 \leq n \leq 500, 1 \leq c \leq 10\,000$), the number of available apps and the available disk space of the phone in megabytes.
- n lines, each with two integers d, s ($1 \leq d, s \leq 10\,000$), the download size and storage size of an app, in megabytes.

Output

Output one line with the maximum number of apps that can be installed. Then output one line listing the numbers of those apps, in the order that Sandra should install them. In the case that no apps can be installed, this line can be omitted.

The apps are numbered from 1 to n, in the order they are given in the input. If there are multiple optimal solutions, output any one of them.

Example

standard input	standard output
2 100 99 1 1 99	2 1 2
2 100 500 1 1 500	0

Problem J. Juggling Troupe

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

At the national centre for computing and advanced circus skills, technical demonstrations by students are strongly encouraged.

A troupe of n novice performers are at this very moment arrayed in a row attempting to put on a juggling show. Unfortunately, none of them are very confident in their craft, and they are struggling. Thus, as soon as an opportunity presents itself, they will try to reduce their part in the performance to make the task easier.

Whenever a juggler has more than one ball in their possession, they will throw one ball to each of their neighbours. In the case that a juggler does not have a neighbour in some direction, they will simply throw the ball offstage instead. Everybody throws their juggling balls simultaneously. The show ends when no juggler has more than one ball.

See Figure 6 below for an illustration of this process.

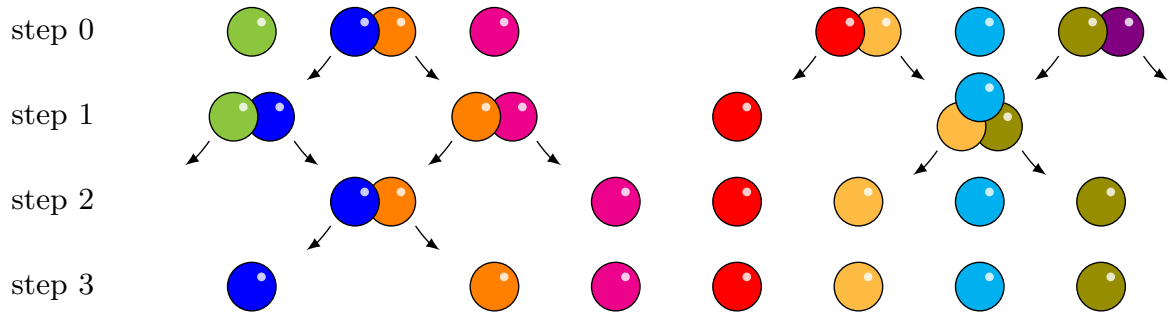


Figure 6: Illustration of Sample Input 1. A performance with $n = 8$ jugglers.

As a member of the audience, you are not impressed by this performance. However, you do wonder how many balls each of the jugglers will have left at the end of the show.

Input

The input consists of:

- One line with a string s of length n ($1 \leq n \leq 10^6$) over the characters `0`, `1` and `2`. The ith character in s represents the number of juggling balls initially held by the ith person.

Output

Output a string s of length n over the characters `0` and `1`, the ith giving the number of juggling balls the ith person has at the end of the show.

Example

standard input	standard output
12100212	10111111
000111222000222111222001	111111101111111111111111

Problem K. Knockout Tournament

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

Laura is organising a knockout tournament, in which her friend Dale takes part. Laura would like to maximise the probability of Dale winning the tournament by arranging the games in a favourable way. She does not know how to do it, so she asked you for help. Naturally, you refuse to cooperate with such a deplorable act—but then you realise that it is a very nice puzzle!

When the number of players is a power of two, the tournament setup can be described recursively as follows: the players are divided into two equal groups that each play their own knockout tournament, after which the winners of both tournaments play each other. Once a player loses, they are out of the tournament.

When the number of players is not a power of two, some of the last players in the starting line-up advance from the first round automatically so that in the second round the number of players left is a power of two, as shown in Figure 7.

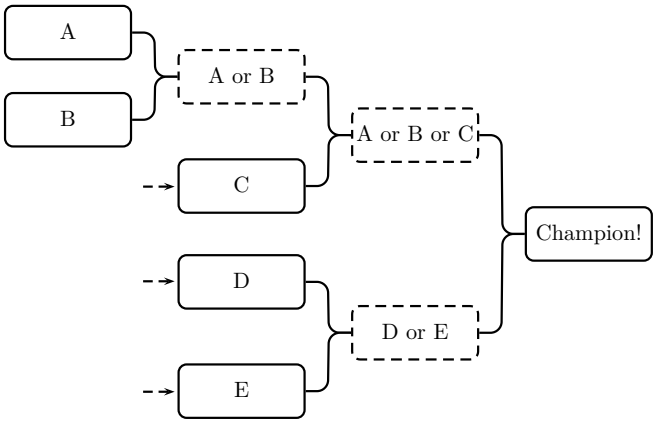


Figure 7: A tournament tree with 5 players. Players C, D, and E advance from the first round automatically.

Every player has a rating indicating their strength. A player with rating a wins a game against a player with rating b with probability $\frac{a}{a+b}$ (independently of any previous matches played).

Laura as the organiser can order the starting line-up of players in any way she likes. What is the maximum probability of Dale winning the tournament?

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 4096$), the total number of players.
- n lines, each with an integer r ($1 \leq r \leq 10^5$), the rating of a player. The first rating given is Dale's rating.

Output

Output the maximum probability with which Dale can win the tournament given a favourable setup. Your answer should have an absolute or relative error of at most 10^{-6}.

Examples

standard input	standard output
4 3 1 2 4	0.364285714
5 1 1 3 3 3	0.125