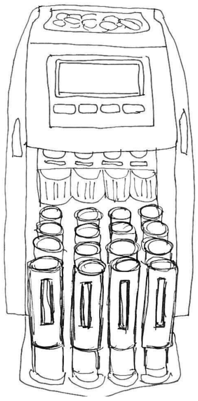
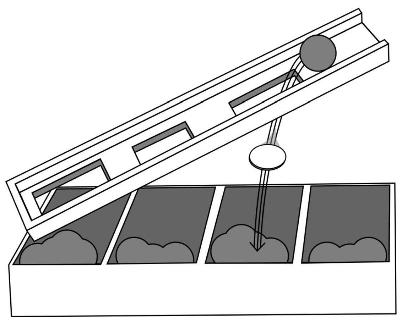
Problem A. Automatic Accountant

Source file name: automatic.c, automatic.cpp, automatic.java, automatic.py
Input: `Standard`
Output: `Standard`



The bank you work in has purchased an advanced technological solution to the problems it has with counting money deposited by clients. The machine works by running each individual coin along a sloped track. At every integer multiple of centimetres along, starting from 1cm, there is a slot in the track with a bucket underneath.



The slot will allow a coin to fall in, if the thickness of the coin (in millimetres) is **less than or equal to** the width of the slot (also in millimetres), and the mass of the coin (in grams) is **greater than or equal to** the trigger mass of the slot (also in grams).

Since the slots are spaced $1cm$ apart centre-to-centre, and since there can be a large number of coins (or other metal shapes) inserted, the amount of wear on the track will depend on total distance travelled by all coins.

Given a list of the coins that will be deposited, what total distance will they travel, in centimetres?

Input

The input consists of:

- one line containing the number of slots, s ($1 \leq s \leq 10^5$).
- s further lines, the ith line containing the width of a slot in millimetres and trigger mass in grams of the ith slot, a_i and b_i respectively ($1 \leq a, b \leq 10^5$).
- one line containing the integer c ($1 \leq c \leq 10^5$), the number of coins.
- c further lines, the jth line containing the thickness in millimetres and mass in grams of the jth coin, u_i and v_i respectively ($1 \leq u, v \leq 10^5$).

It is guaranteed that every coin will be able to fall into at least one slot.

Output

Output the total distance in centimetres travelled by coins.

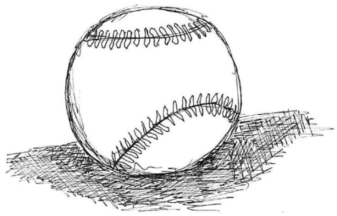
Example

Input	Output
1 10 10 1 5 15	1
3 2 2 1 3 1 1 2 2 2 1 1	4
3 2 2 1 3 1 1 2 2 2 1 2	2
5 2 3 2 2 4 2 4 5 10 1 5 1 5 2 1 4 2 5 3 5 5	19

Problem B. Ballpark Estimate

Source file name:	ballpark.c, ballpark.cpp, ballpark.java, ballpark.py
Input:	`Standard`
Output:	`Standard`

Giving the right level of detail is an important skill for efficient communication. Sometimes, only the high-level message matters.



For example, whenever a person asks for a number, often they just want an estimate. If the value is in the millions, they do not need to know the precise number of hundreds and tens. Likewise, if the value is in the billions, they do not necessarily care about little things like millions.

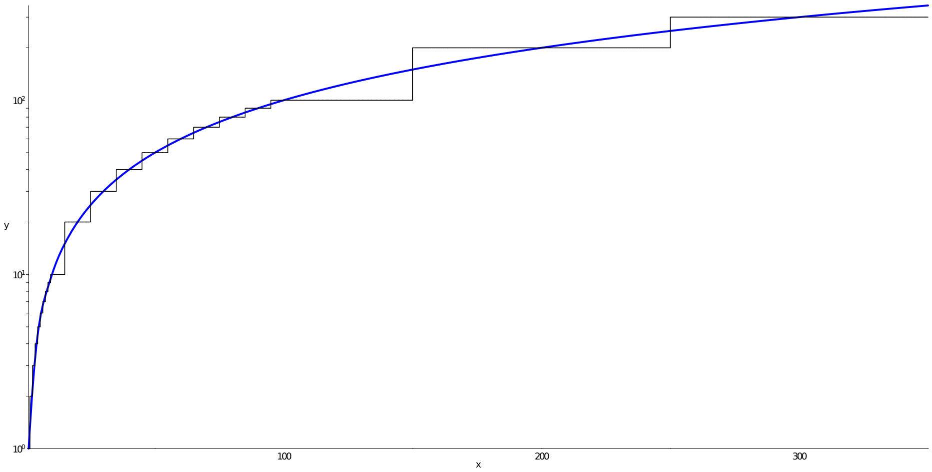


Figure B.1: Illustration of ballpark figures versus actual figures, as a log chart.

Given a (possibly very large) number, print its numerically closest representation that has only one digit other than trailing zeroes.

The closeness of the representation r of a number n is defined by $\text{abs}(r - n)$.

Input

The input consists of:

- one line with the positive integer n ($1 \le n \le 10^{18}$).

Output

Output the closest number to n with exactly one significant (non-zero) figure. If there are two equally-close answers, print the larger one.

Example

Input	Output
150	200
4	4
33471234512345	30000000000000

Problem C. Code Matching

Source file name:	match.c, match.cpp, match.java, match.py
Input:	`Standard`
Output:	`Standard`

James is a spy, working in a country where communication back to his headquarters is not easy. He receives instructions from his superiors over a radio signal, through the use of number stations. A number station is a radio station where a list of numbers are occasionally read out, that have some encrypted meaning. James has a codebook with a mapping of number sequences (e.g., “732736”) to instructions (e.g., “abort the mission”). Every sequence of numbers listed in his codebook maps to a unique instruction. However, James is a lazy spy; he only checks in on the radio occasionally. His superiors know this, and so they assume that he will start listening to their sequence at a random digit, each equally likely (as early as the first, as late as the last). Their instructions are very important though, so it’s essential that they figure out exactly how long it will take James to decode the message. As soon as James can be positive that there is only one message that matches what he is hearing (factoring in that he may have missed some digits before he tuned in), he will stop listening and take action.

Each digit takes one second to read, and there will be silence one second after the last digit (telling James the message is complete). For each message that headquarters might send, you have to determine how long it will take James to understand the message on average. Note that it is possible that sometimes even after listening until the end of the message, James cannot be sure which message he heard. In that case, print “Impossible”.

Given the codebook, and the code to be sent, find the expected amount of time that James will spend listening to his radio.

Input

You will be given an integer, N ($N \leq 10^5$), specifying the number of sequences in James’s codebook. The next N lines will each contain one sequence, with total length not exceeding 10^5 (i.e., the length of all sequences together will not exceed 10^5).

Output

Output N lines. The i^{th} line should contain the expected number of seconds James will spend listening before he can be certain of the message he is hearing if headquarters sends the i^{th} message in his codebook. Your answer will be judged to an absolute error of 10^{-5}. If it’s possible that James might not ever be able to determine the message, print out `Impossible` for that line.

Example

Input	Output

`4`
`17383`
`126`
`385`
`485`
	`2`
	`1.333333333`
	`Impossible`
	`Impossible`

Explanation

For the first message (17383), there are 5 possible spots where James could tune in. If he tunes in at the first digit, it could still be the second sequence (126) since both contain a 1. Once he hears the following 7, it could only be the first sequence. Therefore, tuning in at this digit takes 2 seconds. If he tunes in on the second digit, it takes 1 second. The third digit takes 3 seconds, the fourth digit takes 2 seconds, and the fifth digit takes 2 seconds (he hears the 3, then hears silence). This gives an average of

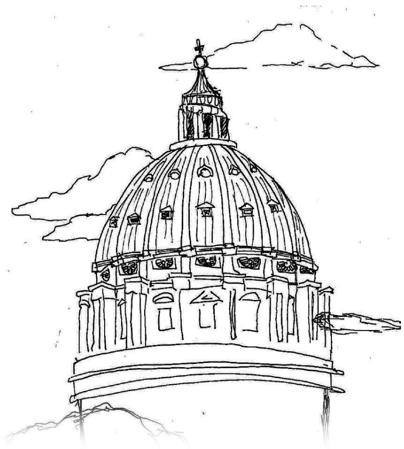
$(2 + 1 + 3 + 2 + 2)/5 = 2$.

For the third sequence, if he tunes in on the 8 then he will hear "`85_`", and therefore cannot tell it apart from the fourth sequence. Thus the answer here is `Impossible`.

Problem D. Dome Construction

Source file name:	dome.c, dome.cpp, dome.java, dome.py
Input:	Standard
Output:	Standard

The world's largest indoor water park is built inside a hemispherical dome that was once used as an aircraft hangar. The park attracts more than 10 000 visitors per day and is big enough that it even has its own tropical micro-climate with clouds forming inside.



Management would like to expand business operations by opening another branch in the dome of your local cathedral. The micro-climate is a key selling point, so to really capitalise on the cathedral they asked you to expand the dome's radius so that it contains at least a given number of clouds. A cloud is contained if its centre is on or inside the boundary of the dome.

You are a cloud engineer by trade, and hence a competent meteorologist. You already identified several potential clouds close by and plotted them in three dimensions relative to the centre of the current structure. In order to capture enough of them, how large do you need to make the radius of the dome?

Input

- The first line contains the number of clouds you found, n, and the number that must be contained, k, respectively ($1 \le k \le n \le 10^5$).
- The next n lines each contain three real numbers x_i, y_i, z_i, the coordinates of the ith cloud relative to the centre of the dome ($0 \le |x_i|, |y_i|, |z_i| \le 10^6$). Every cloud has a non-negative y-coordinate.

Output

Output the minimum radius of the dome required to enclose at least k points. Your answer must be accurate to an absolute error of 10^{-6}.

Example

Input	Output
5 3 -4 2 1 2.1 3 5 1.2 1 -1 -2.2 3 2 1 0 2.1	4.22374242

Problem E. Estate Agent

Source file name:	estate.c, estate.cpp, estate.java, estate.py
Input:	`Standard`
Output:	`Standard`

Rupert makes a living as the only real estate agent in a small town in England. He asks for 5% commission for every house that he sells.

Rupert organises one big auction per year. Every family (numbered from 1 to n) must participate in this action, although making or an accepting an offer is optional. Everyone puts in bids for the houses they would like to move to, provided they can sell their current house at the same time.

This is a very transparent process–Rupert can see exactly how much commission he will make if he accepts the right buyers' offers on behalf of the sellers. He may discard some offers from buyers in order to drive up the overall commission. In fact, he might even decide to discard all of the offers from one family and let them stay in their current home, if it makes more money for him.

Find the maximum commission Rupert can make if he discards offers optimally.

Input

The input consists of:

- one line containing two integers n and m ($1 \leq n \leq 150$, $0 \leq m \leq n \times (n-1)$), the number of families on the market and the number of offers made.
- m lines, describing the offers.

  The ith such line contains three integers f_i, h_i and a_i ($1 \leq f_i, h_i \leq n$, $f_i \neq h_i$, $0 \leq a_i \leq 10^6$), the family making the offer, the family that owns the house that the offer is for and the amount offered.

  No family makes more than one offer to the same house.

Output

Output how much Rupert will earn via commissions if he discards offers optimally. Your answer must be accurate to an absolute error of 10^{-6}.

Example

Input	Output
4 5 1 2 3 2 3 9 3 1 5 3 2 11 4 1 6	1.0
4 5 1 2 15 2 3 9 3 1 5 3 2 11 4 1 6	1.45

Problem F. Feeding Seals

Source file name:	feeding.c, feeding.cpp, feeding.java, feeding.py
Input:	Standard
Output:	Standard

You are in charge of feeding the seals in the Welsh Mountain Zoo. This involves purchasing buckets of fish and allocating them to volunteers to trek into the enclosure and distribute fairly to the blubbery residents.

The buckets of fish are already set out. Each volunteer can be assigned to carry either one or two of these buckets, as long as the combined weight of the buckets is small enough.

How many volunteers will you need to distribute all of the fish in one trip?

Input

- The first line contains the number of buckets to be delivered, n ($1 \le n \le 10^5$), and the integer carrying capacity of a volunteer, c ($1 \le c \le 10^9$).
- The second line contains the integer weights of each of the n buckets, $w_1 \ldots w_n$ ($1 \le w \le c$).

Output

Output the minimum number of volunteers required to deliver all of the buckets of fish.

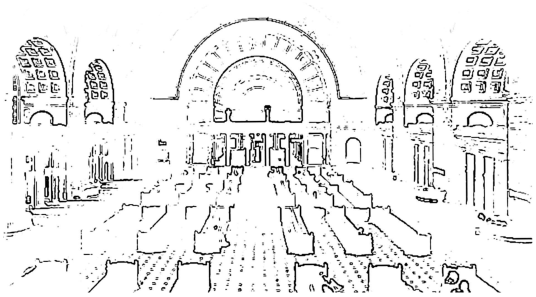
Example

Input	Output
4 100 44 35 66 67	3
1 10 7	1
3 12 10 5 6	2

Problem G. Grand Central Station

Source file name:	grand.c, grand.cpp, grand.java, grand.py
Input:	`Standard`
Output:	`Standard`

The city you live in just finished construction of its new transport network, PlusRail. There are n stations and exactly one way to get between any given pair of them; this is because there are only $n-1$ direct station:station connections. In other words, the network forms a tree.



You have been hired to put together the signage for each of the stations which shows where on the network a passenger is with a big arrow pointing to the bright red station in the centre.

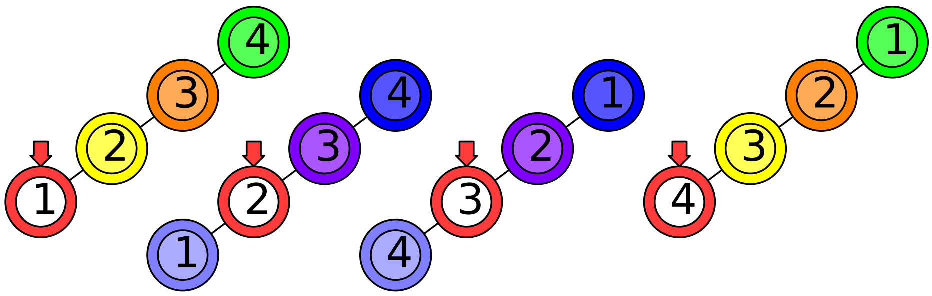


Figure G.1: Illustration of Example Input 1 and how two designs are reused four times, with the labels painted at different places.

Because the drawings of the network are fairly crude, it is actually possible that you could use the same sign in more than one station, and just write a different permutation of labels for the station names.

If you want to make signage for the whole network, what is the minimum number of unique designs you will need?

Input

- The first line of input contains the number of stations, n ($1 \le n \le 3 \times 10^5$).
- The following $n-1$ lines each contain two distinct vertex indices a and b ($1 \le a, b \le n$) indicating that there is a direct route between these stations.

Output

Output the minimum number of map designs that can be made, such that for any station at least one of these map designs can be re-labelled such that this station is in the centre.

Example

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td>4
1 2
2 3
3 4</td><td>2</td></tr>
<tr><td>11
1 2
2 3
3 4
4 5
4 6
4 7
5 10
10 9
10 8
7 11</td><td>10</td></tr>
<tr><td>7
7 1
7 2
3 2
7 4
5 4
6 5</td><td>7</td></tr>
</table>

Problem H. House of Cards

Source file name:	houseofcards.c, houseofcards.cpp, houseofcards.java, houseofcards.py
Input:	`Standard`
Output:	`Standard`

Brian and Susan are old friends, and they always dare each other to do reckless things. Recently Brian had the audacity to take the bottom right exit out of their annual maze race, instead of the usual top left one. In order to trump this, Susan needs to think big. She will build a house of cards so big that, should it topple over, the entire country would be buried by cards. It's going to be huge!

Picture by Liftarn on Wikipedia, public domain

The house will have a triangular shape. The illustration to the right shows a house of height 6, and Figure H.1 shows a schematic figure of a house of height 5.

For aesthetic reasons, the cards used to build the tower should feature each of the four suits (clubs, diamonds, hearts, spades) equally often. Depending on the height of the tower, this may or may not be possible. Given a lower bound h_0 on the height of the tower, what is the smallest possible height $h \geq h_0$ such that it is possible to build the tower?

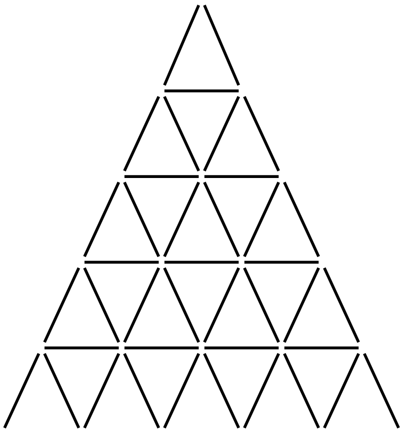


Figure H.1: A house of height 5 uses 40 cards.

Input

A single integer $1 \leq h_0 \leq 10^{1000}$, the minimum height of the tower.

Output

An integer, the smallest $h \geq h_0$ such that it is possible to build a tower of height h.

Example

Input	Output
2	5
42	45

Problem I. Islands

Source file name:	islands.c, islands.cpp, islands.java, islands.py
Input:	`Standard`
Output:	`Standard`

Weenesia is an archipelago of perfectly circular islands in the 2D plane. Many islands have palm trees growing in them, perfectly straight up, so we can represent a tree by a 2D point and a height.

Picture by Anne Sheppard, cc-by

We want to build a courier system in the islands, i.e., we want to be able to move any object from any land point to any other land point. It is possible to move an object within an island without restrictions. It is also possible to climb to a palm tree and throw the object to a distance proportional to the height of the palm tree (or less), and possibly in a different island.

Unfortunately, this may not be enough to reach our goal, so we might want to build a tunnel between two islands. The tunnel connects two points on two islands and may cross under both the sea and other islands. Each of the two entrances of a tunnel must be at least 1 meter away from the sea to avoid flooding.

Your task is to find the minimum length of a tunnel such that a courier system is possible.

Input

The first line contains three integers n, m, and k ($1 \le n \le 5\,000$, $0 \le m \le 10\,000$, $1 \le k \le 1\,000$), the number of islands and palm trees respectively, and the ratio between object throwing range and palm height.

Each of the next n lines contains three integers x, y, and r ($|x|, |y| \le 10^6$, $100 \le r \le 10^6$), the center and radius of an island, in centimetres. Each of the next m lines contains three integers x, y, h ($|x|, |y| \le 10^6$, $1 \le h \le 10^4$), the center and height of a palm tree, in centimetres.

No two islands intersect. Each palm tree lies strictly inside an island. No two palm trees grow in the same spot.

Output

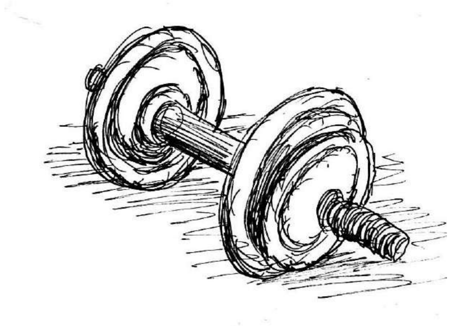
Output the minimum length of a tunnel in centimetres, 0 if no tunnel is needed, or `impossible` if no such tunnel exists. Answers with an absolute precision up to 10^{-6} will be accepted.

Example

Input	Output
3 2 3 0 0 400 1000 0 400 2000 0 400 300 0 150 1300 0 150	1400
3 2 2 0 0 400 1000 0 400 2000 0 400 300 0 100 1300 0 100	impossible

Problem J. Jammed Gym

Source file name: jammed.c, jammed.cpp, jammed.java, jammed.py
Input: `Standard`
Output: `Standard`



You are at the fitness centre to run through your exercise programme. You must use the kinds of exercise machine in an order precisely dictated by the programme, although there may be more than one instance of a machine.

You start at the centre of a unit circle around which the exercise stations are arranged. You can walk directly between any two points in the circle, and you may also visit the same point multiple times. See Figure 6 below for an example.

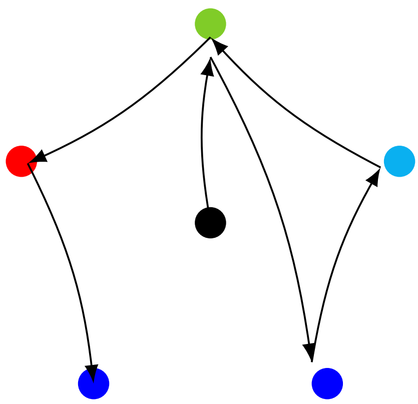


Figure J.1: Illustration of Example Input 1. Types of machine: $[1, 2, 4, 1, 3, 2]$

Exercise is an important and noble endeavour, but in today's busy world we must strive for efficiency in everything we do. Find the most efficient way of visiting exercise stations that matches the order given.

Input

- The first line of input contains the number of exercises in the programme, n $(1 \le n \le 100)$.
- The second line of input contains n space-separated integers each denoting the type of an item on the programme t $(1 \le t_i \le 100)$. There will always be at least one station for each programme in this list.
- The third line of input contains the number of stations, m $(1 \le m \le 100)$.
- The fourth line of input contains m space-separated integers each denoting the type of a station q $(1 \le q_i \le 100)$.

Output

Output the minimum distance you will need to walk. Your answer must be accurate to an absolute error of 10^{-6}.

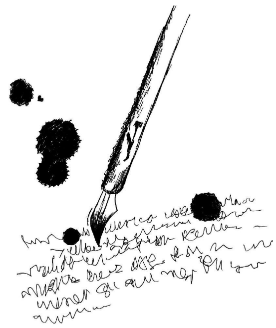
Example

Input	Output
6 1 2 4 1 3 2 5 1 4 2 2 3	7.604395
5 4 2 1 3 1 6 1 2 1 3 1 4	5.732051

Problem K. Knocked Ink

Source file name:	knocked.c, knocked.cpp, knocked.java, knocked.py
Input:	`Standard`
Output:	`Standard`

You knocked over the inkwell for the team fountain pen... Now spots of ink are beginning to form on the page and spread out. This is really going to hamper your speed at writing up programming contest solutions.



The ink spreads out by forming infinitesimally-small blots on the page. A blot that appears at time t seconds after the incident grows in radius smoothly, at a rate of 1cm per second, and may eventually overlap with other blots on the page.

At first the page is still usable, but when the combined size of the ink blots grows large enough, you will have to abandon your work and find another piece of paper upon which to type up solutions.

How long will it take for this to happen?

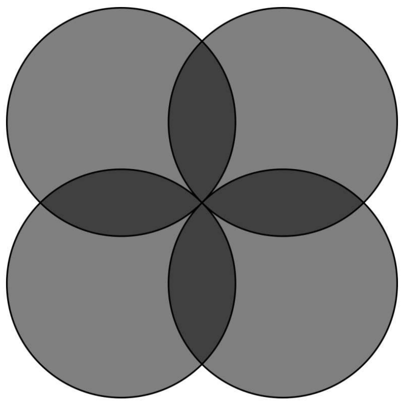


Figure K.1: Illustration of Example Input 1.

Input

The input consists of:

- one line with the number of inkblots, n ($1 \leq n \leq 100$), and the real-valued total area of ink in square centimetres at which the paper must be abandoned, a ($1 \leq a \leq 10^9$).
- n further lines, each with the x and y coordinates in centimetres of an ink blot ($-10^6 \leq x, y \leq 10^6$) and the time in seconds at which the blot first appears, t ($0 \leq t \leq 10^6$).

Output

Output the time in seconds at which the ink blots cover exactly a square centimetres of the infinitely-large page. Your answer must be accurate to an absolute error of 10^{-6}.

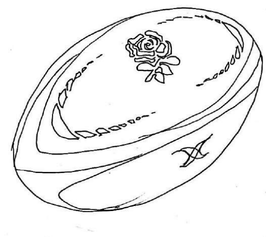
Example

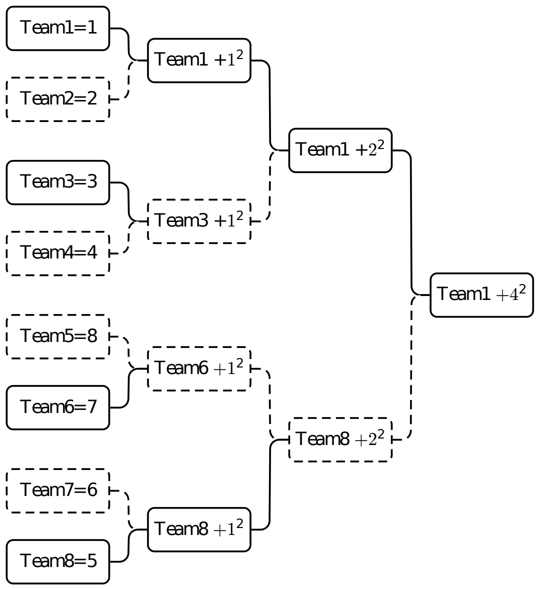
Input	Output
4 20.566371 0.0 0.0 0.0 0.0 2.0 0.0 2.0 0.0 0.0 2.0 2.0 0.0	1.4142135624
2 785.398163397 -50 0 20 50 0 30	35
5 10000 0 0 0 0 0 1 0 0 2 10 0 1 0 -5 2	53.3322048

Problem L. Low Effort League

Source file name:	league.c, league.cpp, league.java, league.py
Input:	`Standard`
Output:	`Standard`

The teams in your local rugby league aren't particularly good, but they make up for it in enthusiasm. We are going to organise a single-elimination knockout tournament between them, where the 2^n teams play n rounds. In each round, the $2i+1$th remaining team pairs up with the $2i+2$th team and one or the other team is eliminated.





Each team has a scalar skill level. In the normal course of things, a team with higher skill level will always beat a team with lower skill level. However, training plays a part too: if one team studies another, learns its techniques, and trains against them, it can win.

The number of hours a team with skill a must train to beat a team with skill b (where $a \leq b$) is $|b-a|^2$. This training only affects that one game (it does not transfer to other teams).

You would quite like your favourite team to win the tournament. If you take complete control over how every team trains, you can always make this happen. What is the minimum number of hours needed, in total across all teams, in order for your team (team 1) to win?

Input

The input consists of:

- one line containing the integer r ($1 \leq r \leq 14$), the number of rounds in the tournament.
- one line with 2^{r} integers $s_1 \dots s_{2^{\mathrm{s}}}$ ($0 \leq s_i \leq 10^6$ for each i), where s_i is the skill level of the ith team.

Output

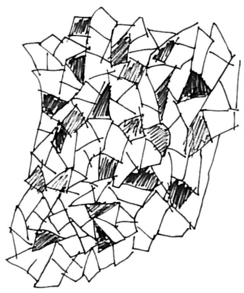
Output the smallest number of hours needed for team 1 to win the tournament.

Example

Input	Output
1 50 40	0
3 1 2 3 4 8 7 6 5	28

Problem M. Mosaic Mansion

Source file name:	mosaic.c, mosaic.cpp, mosaic.java, mosaic.py
Input:	`Standard`
Output:	`Standard`



A mosaic is a picture made from square tiles arranged in a grid, at least for today's purposes.

We would like to make a mosaic with exactly the same number of tiles of each colour. We will do this by taking an existing design and removing some of the rows from it.

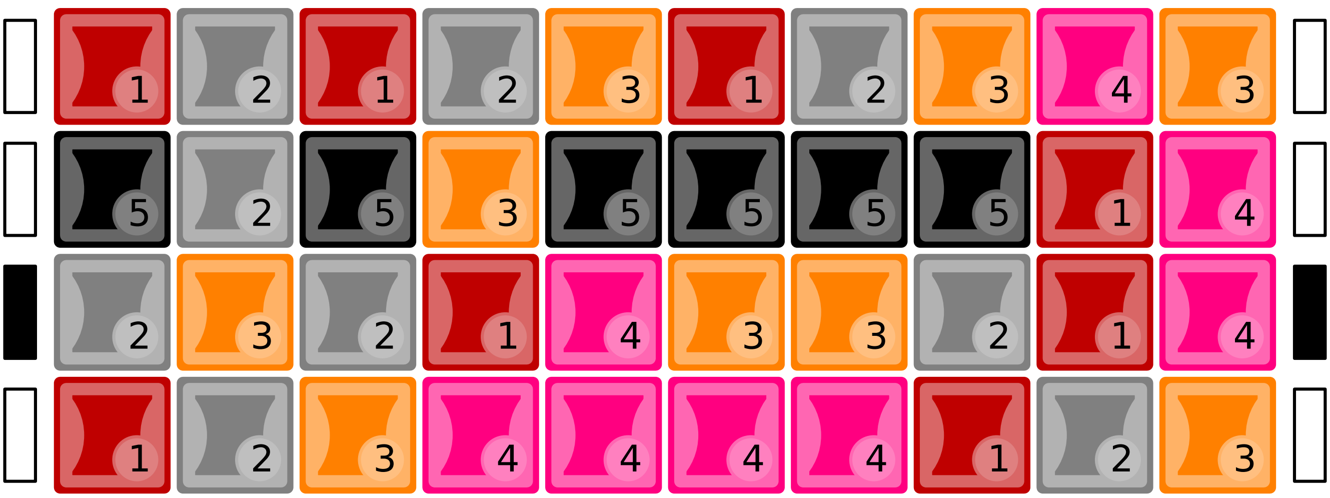


Figure M.1: Illustration of a solution to Example Input 1. The three rows annotated with white can be kept, giving 6 of each colour of tile.

What is the greatest number of rows we can keep?

Input

- The first line of input contains the number of rows, n ($1 \le n \le 40$), the number of columns, m ($1 \le m \le 10^5$), and the number of colours, c ($1 \le c \le 10^5$) in the mosaic respectively.
- Each of the next n lines contains m colours of cells $p_1 \ldots p_m$ ($1 \le p \le c$).

Output

Output the greatest number of rows that can be kept while keeping equal representation for each colour in the input, or 0 if no rows can be kept.

Example

Input	Output
4 10 5 1 2 1 2 3 1 2 3 4 3 5 2 5 3 5 5 5 5 1 4 2 3 2 1 4 3 3 2 1 4 1 2 3 4 4 4 4 1 2 3	3