Problem A
Riddle of the Sphinx
Time limit: 2 seconds

Oedipus and the Sphinx by Gustave Moreau, 1864, public domain

One of the most remarkable landmarks in Egypt is the Great Sphinx of Giza, a statue depicting a mythical creature with the head of a human, the body of a lion, and the wings of an eagle. Sphinxes were regarded as guardians in Egyptian and Greek mythologies. Probably the most famous sphinx is the one who guarded the Greek city of Thebes. According to myths, when Oedipus tried to enter the city, the sphinx gave him the following riddle: “Which creature has one voice, but has four feet in the morning, two feet in the afternoon, and three feet at night?” As you might have heard, Oedipus correctly answered, “Man — who crawls on all fours as a baby, then walks on two feet as an adult, and then uses a walking stick in old age.”

In this problem, you meet a different sphinx who gives you a somewhat reversed riddle: “How many legs do an axex, a basilisk, and a centaur have?” While you recognize these as creatures from Egyptian and Greek mythology, you have no clue how many legs each has (except that it is a nonnegative integer). The sphinx sternly instructs you to not touch anything so you are unable to search for the answer on your phone.

However, the sphinx allows you to ask her five questions. In each question you can ask the sphinx how many legs some number of these creatures have in total. For instance, you could ask, “How many legs do three basilisks and one axex have in total?” or “How many legs do five centaurs have?” Seems easy enough, you think, but then you remember that sphinxes are tricky creatures: one of the sphinx’s five answers might be an outright lie, and you do not know which one.

Write a program to talk to the sphinx, ask the five questions, and solve the riddle.

Interaction

There are exactly five rounds of questions. In each question round, you must first write a line containing three space-separated integers a, b, and c ($0 \leq a, b, c \leq 10$), representing the question “How many legs do a axex, b basilisks, and c centaurs have in total?” After the question is asked, an input line containing a single integer r ($0 \leq r \leq 10^5$) is available on standard input, giving the sphinx’s answer to your question.

After the five rounds of questions, output a line containing three space-separated nonnegative integers ℓ_a, ℓ_b, and ℓ_c, indicating the number of legs of an axex, a basilisk, and a centaur, respectively.

Sample Interaction 1

Read	Write
	1 1 1
12	
	1 1 1
13	
	5 0 1
24	
	1 0 0
4	
	1 1 0
8	
	4 4 4

Sample Interaction 2

Read	Write
	4 4 4
2023	
	1 0 0
0	
	0 1 0
42	
	0 0 1
2024	
	0 0 0
0	
	0 42 2024

Problem B
Schedule
Time limit: 2 seconds

The Institute for Creative Product Combinations (ICPC) tries to find unusual and innovative ways to unite seemingly unrelated products or technologies, opening up new markets and creating new jobs. (For instance, their most recent success was the "hairbachi," a hair-dryer with a hibachi grill top attachment for preparing on-the-go hot meals.) The company employs n teams of size 2 to research individual products, then members of the different teams get together to explore ways of combining products.

During the pandemic, the ICPC management organized everyone's schedule in such a way that there were never more than n people in the office at the same time, and things ran so smoothly that they continued the process once things began to return to normal. Here is the scheme they used. Label the teams with integers 1 through n and the two people on the i^{th} team as $(i, 1)$ and $(i, 2)$ for each i from 1 to n. Each week, exactly one person from each team is allowed in the office, while the other has to stay away. The employees $(i, 1)$ and $(i, 2)$ know each other well and collaborate productively regardless of being isolated from each other, so members of the same team do not need to meet in person in the office. However, isolation between members from different teams is still a concern.

Each pair of teams i and j for $i \neq j$ has to collaborate occasionally. For a given number w of weeks and for fixed team members (i, a) and (j, b), let $w_1 < w_2 < \ldots < w_k$ be the weeks in which these two team members meet in the office. The isolation of those two people is the maximum of

$$\{w_1, w_2 - w_1, w_3 - w_2, \ldots, w_k - w_{k-1}, w + 1 - w_k\},$$

or infinity if those two people never meet. The isolation of the whole company is the maximum isolation across all choices of i, j, a, and b.

You have been tasked to find a weekly schedule that minimizes the isolation of the whole company over a given number w of weeks.

Input

The input consists of a single line containing two integers n ($2 \leq n \leq 10^4$) and w ($1 \leq w \leq 52$), where n is the number of teams and w is the number of weeks that need to be scheduled.

Output

Output a line containing either an integer representing the minimum isolation achievable for n teams or the word `infinity` if no schedule guarantees that every pair of individuals on different teams can meet. If the isolation is finite, it is followed by w lines representing a schedule that achieves this isolation. The j^{th} line of the schedule is a string of length n containing only the symbols `1` and `2`, where the i^{th} symbol indicates which of the two members from team i comes into the office on week j.

Sample Input 1	Sample Output 1
2 6	4 11 12 21 22 11 12

Sample Input 2	Sample Output 2
2 1	infinity

Problem C
Three Kinds of Dice
Time limit: 1 second

Image from rawpixel, CC0

See how they roll! According to a famous story, Warren Buffett once challenged Bill Gates to a simple game of dice. He had three dice; the first player could examine them and choose one of the three. The second player would then choose one of the remaining dice, and both players would roll their dice against each other, aiming for the highest numbers. Warren offered to let Bill go first, but this made Bill suspicious so he opted to go second. It turned out to be a wise choice: these were *intransitive dice*. The first die had an advantage when rolling against the second, the second had an advantage when rolling against the third, but the first did not have an advantage when rolling against the third!

To formalize this: define a "die" as any shape with at least one face such that each face shows a positive integer. When a die is rolled, one of its faces is selected uniformly at random. When two dice roll against each other, the die whose selected face shows a higher number earns 1 point; if both numbers are equal, each die earns $\frac{1}{2}$ points. For dice D and D', define $score(D, D')$ as the expected number of points D earns from a single roll against D'. If $score(D, D') > \frac{1}{2}$, we say that D has an advantage over D'; if $score(D, D') = \frac{1}{2}$, the two dice are tied. For example, if D is the first die in the sample input and D' is the second, $score(D, D') = \frac{4}{9}$ and $score(D', D) = \frac{5}{9}$, so D' has an advantage over D.

Given two dice D_1 and D_2 such that D_1 has an advantage over D_2, you want a third die D_3 that forms an intransitive trio with the other two. Among all D_3 that have an advantage over or tie with D_1, compute the lowest possible $score(D_3, D_2)$. If this is less than $\frac{1}{2}$, you can make an intransitive trio! Similarly, among all D_3 such that D_2 has an advantage over or ties with D_3, compute the highest possible $score(D_3, D_1)$.

Input

The input contains two lines, each describing one die. One of the dice (the first or the second) has an advantage over the other. The die with the advantage is D_1 and the other is D_2.

The first integer on a line gives n ($1 \le n \le 10^5$), the number of faces on the die. Then follow n integers f_i ($1 \le f_i \le 10^9$ for each $1 \le i \le n$), giving the integer on each face.

Output

Output one line containing the lowest $score(D_3, D_2)$ and the highest $score(D_3, D_1)$ under the above conditions. The two scores do not need to use the same die D_3. Your answer should have an absolute error of at most 10^{-6}.

Sample Input 1	Sample Output 1
6 1 1 6 6 8 8 3 2 4 9	0.291666667 0.750000000

Sample Input 2	Sample Output 2
4 9 3 7 5 3 4 2 3	0.500000000 0.500000000

Problem D
Carl's Vacation
Time limit: 1 second

Carl the ant is back! After traversing meandering paths (Problem A, 2004 World Finals) and wandering over octahedrons (Problem C, 2009 World Finals) it is time for a little vacation — time to see the sights! And where better to see the sights than at the tips of tall structures like, say, pyramids!! And where better to see tall pyramids but Egypt!!! (This is so exciting!!!!!)

After taking in the view from the tip of one pyramid, Carl would like to go to the tip of another. Since ants do not do particularly well in the hot sun, he wants to find the minimum distance to travel between the tips of these two pyramids, assuming he can only walk on the surfaces of the pyramids and the plane which the pyramids sit upon. The pyramids are, geometrically, right square pyramids, meaning the apex of the pyramid lies directly above the center of a square base.

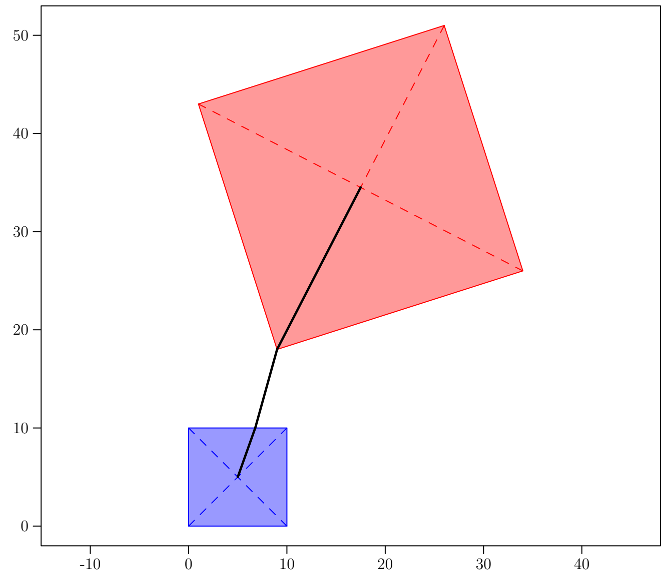


Figure D.1: Illustration of two pyramids corresponding to Sample Input 1. The black line shows the shortest path between their apexes.

Input

The first line of input contains five integers x_1, y_1, x_2, y_2, h where x_1, y_1, x_2, y_2 ($-10^5 \leq x_1, x_2, y_1, y_2 \leq 10^5$ and $(x_1, y_1) \neq (x_2, y_2)$) define an edge of the first pyramid, with the body of the pyramid lying to the left of the directed vector from (x_1, y_1) to (x_2, y_2), and h ($1 \leq h \leq 10^5$) is the height of the pyramid. The second line of input describes the second pyramid in the same format. The intersection of the bases of the two pyramids has 0 area.

Output

Output the minimum distance Carl travels between the tips of the two pyramids. Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
0 0 10 0 4 9 18 34 26 42	60.866649532

Problem E
A Recurring Problem
Time limit: 20 seconds

You have a very big problem! You love recurrence relations, perhaps a bit too much. In particular, you are a fan of positive linear recurrence relations (PLRR), which can be defined as follows. First, you choose the order k of the relation. Then you choose coefficients $c_1, c_2, \ldots, c_k$, and the first k elements of a sequence $a_1, a_2, \ldots, a_k$. The relation is called "positive" if all of these numbers are positive integers. The rest of the sequence can then be generated indefinitely using the formula

$$a_{i+k} = c_1 \cdot a_i + c_2 \cdot a_{i+1} + \cdots + c_k \cdot a_{i+k-1} \quad \text{for} \quad i \geq 1.$$

The Fibonacci sequence is the most famous recurrence of this form, but there are many others.

In fact, yesterday, in a fit of mad mathematical inspiration, you wrote down *all* possible ways of choosing a positive linear recurrence relation, and each associated infinite sequence, on some index cards, one per card. (You have a lot of index cards; you buy in bulk.) It has all been a bit of a blur. But when you woke up today, you realized that you do not have a good way to order or count the PLRRs. You tried just sorting the sequences lexicographically, but there are too many that start with "1" — you will never make it to the later ones.

Fortunately, inspiration struck again! You realized that you can instead order the PLRRs lexicographically by the generated part of the sequence only (that is, the part of the sequence starting after the initial k values). Ties are broken by lexicographic order of the coefficients. For example $k = 1$, $c_1 = 2$, $a_1 = 2$ comes before $k = 2$, $(c_1, c_2) = (2, 1)$, $(a_1, a_2) = (1, 2)$, even though the continuation of the sequence is the same for both. This allows you to properly index your cards, starting from 1, with every card being assigned a number.

Given the number on a card, describe the sequence on it!

Input

The input consists of a single line with an integer n ($1 \leq n \leq 10^9$), the index of the desired PLRR.

Output

Output four lines detailing the desired recurrence relation. The first line contains its order k. The second line contains the k coefficients $c_1, \ldots, c_k$. The third line contains the k starting values $a_1, \ldots, a_k$. The fourth line contains the first 10 of the generated values.

Sample Input 1	Sample Output 1
3	2 1 1 1 1 2 3 5 8 13 21 34 55 89 144

Sample Input 2	Sample Output 2
1235	4 1 1 3 1 3 2 1 1 9 15 44 99 255 611 1519 3706 9129 22377

Problem F
Tilting Tiles
Time limit: 3 seconds

You found a weird puzzle in a box with old toys in your attic. The puzzle forms a rectangular grid board made of $h \times w$ square cells. Some cells in that grid have a colored tile placed on them, as shown in Figure F.1.

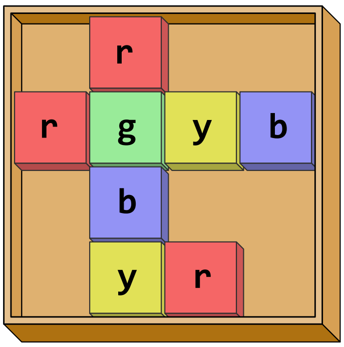


Figure F.1: Color tiles correspond to the starting arrangement in Sample Input 1.

You are not yet sure what the exact goal of this puzzle is, but you started examining possible ways of rearranging the tiles. Their arrangement can be manipulated by tilting the grid in one of the four cardinal directions: to your left, to your right, towards you, or away from you. Tilting causes all the tiles to slide in the respective direction until they are blocked either by the boundary or by another tile. Given a starting and ending arrangement, determine whether there exists some sequence of tilts that transforms the former into the latter. Figure F.2 illustrates tilting of the puzzle shown in Sample Input 1.

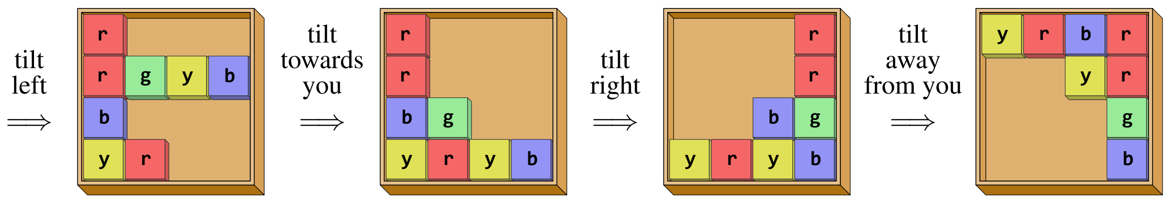


Figure F.2: Solution to Sample Input 1.

Input

The first line of input contains two integers h and w ($1 \leq h, w \leq 500$) representing the height and width of the grid. Then follow h lines giving the starting arrangement from the top row to the bottom row. Each of these lines contains a string of length w describing cells on the row from left to right. If a cell is empty, the corresponding character is a dot (`.`). If there is a tile, the color of that tile is given, denoted by a lowercase letter (`a`-`z`). Different letters represent different colors, and tiles of the same color cannot be distinguished.

After the starting arrangement, there is one empty line and then follows a description of the ending arrangement, consisting of h lines in the same format as for the starting arrangement.

Output

Output `yes` if a sequence of tilts exists that transforms the starting arrangement to the ending arrangement, and `no` otherwise.

Sample Input 1

```
4 4
.r..
rgyb
.b..
.yr.

yrbr
..yr
...g
...b
```

Sample Output 1

```
yes
```

Sample Input 2

```
1 7
....x..

..x....
```

Sample Output 2

```
no
```

Sample Input 3

```
4 3
yr.
..b
ry.
b..

...
..b
.ry
byb
```

Sample Output 3

```
no
```

Problem G
Turning Red
Time limit: 3 seconds

Mei's parents have spent the last year remodeling their house, but their lighting system is quite complex! Each room in the house has an LED light, which can be set to red, green, or blue, as seen in Figure G.1.

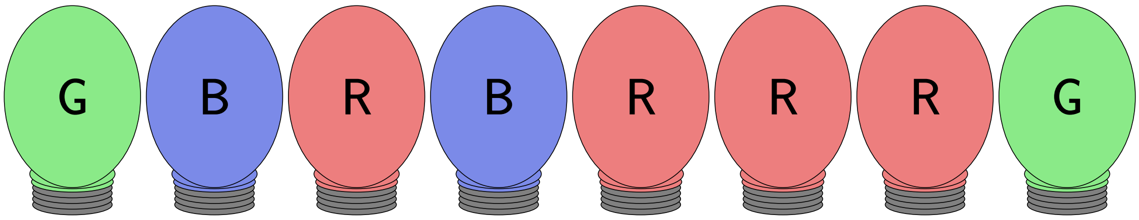


Figure G.1: The initial state of the lights in Sample Input 1. Buttons and wires not shown.

Throughout the house are various buttons which are each connected to one or more lights. When a button is pressed, any red lights connected to that button become green, any green lights connected to that button become blue, and any blue lights connected to that button become red. Each button can be pressed multiple times. Because the house was built prior to the invention of crossbar wiring, each light is controlled by at most two buttons.

Mei's favorite color is red, so she wants to turn all of the lights red. Her parents, fearing the buttons will wear out, have asked her to minimize the total number of button presses.

Input

The first line of input contains two positive integers l and b, where l $(1 \leq l \leq 2 \cdot 10^5)$ is the number of lights and b $(0 \leq b \leq 2 \cdot l)$ is the number of buttons. The second line of input is a string of l characters, all either `R`, `G`, or `B`, where the i^{th} character is the initial color of the i^{th} light. The next b lines describe the buttons. Each of these lines begins with an integer k $(1 \leq k \leq l)$, the number of lights controlled by this button. Then k distinct integers follow, the lights controlled by this button. The lights are indexed from 1 to l, inclusive. Each light appears at most twice across all buttons.

Output

Output the minimum number of button presses Mei needs to turn all the lights red. If it is impossible for Mei to turn all of the lights red, output `impossible`.

Sample Input 1	Sample Output 1
8 6 GBRBRRRG 2 1 4 1 2 4 4 5 6 7 3 5 6 7 1 8 1 8	8

Sample Input 2	Sample Output 2
4 3 RGBR 2 1 2 2 2 3 2 3 4	impossible

Sample Input 3	Sample Output 3
4 4 GBRG 2 1 2 2 2 3 2 3 4 1 4	6

Sample Input 4	Sample Output 4
3 3 RGB 1 1 1 2 1 3	3

Problem H
Jet Lag
Time limit: 2 seconds

The ICPC World Finals are here and they are packed full of activities you want to attend — speeches, presentations, fun events, not to mention the contest itself. There is only one problem: when are you going to sleep?

When you fall asleep, you always set a timer because otherwise you would be able to sleep forever. Using the timer, you can choose to sleep for any positive integer amount of minutes. After sleeping for k minutes, you will be rested for another k minutes (and so you will not be able to fall asleep again); and then you will be able to function for a third k minutes (so you can stay awake, but you can also go to sleep if you want to).

You know the times of all the activities at the Finals; you should plan your sleep schedule to not miss any part of any event. Just before the Finals start (at minute 0), you will arrive in your hotel room after a long journey and you will need to sleep immediately.

Input

The first line of input contains a positive integer n ($1 \le n \le 200\,000$), the number of activities planned for the Finals.

The i^{th} of the remaining n lines contains two positive integers b_i and e_i ($b_i < e_i$, $e_i \le b_{i+1}$, $0 \le b_1$, $e_n \le 10^{10}$), the beginning and end time of the activity, counted in minutes from the beginning of the Finals.

Output

If it is possible to find a sleep schedule that allows you to participate in all planned activities in their entirety, then output such a schedule in the format described below. Otherwise, output `impossible`.

A sleep schedule is specified by a line containing the number p ($1 \le p \le 10^6$) of sleep periods, followed by p lines. The i^{th} of these lines contains two integers s_i and t_i — the beginning and end time of the i^{th} sleep period, counted in minutes from the beginning of the Finals. Note that you should not output any sleep period after the last activity.

The sleep periods must satisfy $0 = s_1 < t_1 < s_2 < t_2 < \ldots < t_p \le b_n$ as well as the condition described in the statement that does not allow you to fall asleep for some time after a sleep period. You may fall asleep immediately after an activity (so it may be that $s_i = e_j$) and you may wake up just before an activity (so it may be that $t_i = b_j$).

If there are multiple valid sleep schedules, any one will be accepted. It can be shown that if there is a valid sleep schedule, then there is also one with at most 10^6 sleep periods.

Sample Input 1	Sample Output 1
3 30 45 60 90 120 180	2 0 30 90 120

Sample Input 2	Sample Output 2
1 0 60	impossible

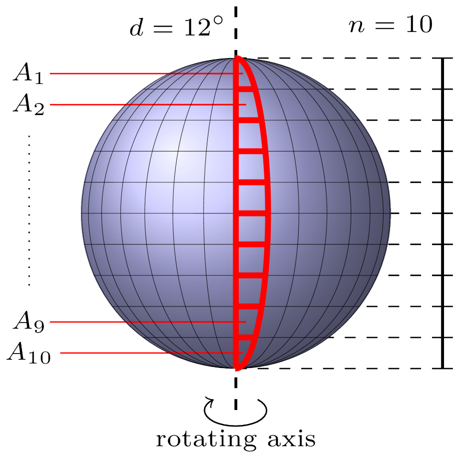
Sample Input 3	Sample Output 3
7 31 32 35 41 48 55 69 91 1000 2022 2022 2023 2994 4096	5 0 5 10 28 56 68 92 900 2025 2900

Problem I
Waterworld
Time limit: 3 seconds

Thousands of planets outside the Solar System have been discovered in recent years. An important factor for potential life support is the availability of liquid water. Detecting water on faraway planets is not easy. For rotating planets, a brand-new technology using relativistic quantum-polarized spectroscopy can help. It works as follows (this is a simplified description as only three people on *this* planet understand how it really works).

Assume the telescope shows the planet such that its rotating axis is vertical and its equator is horizontal. Only the vertical line at the center of the image (the line that covers the rotating axis) is analyzed, because it provides the highest resolution of the planet's surface.

The analysis proceeds in steps of d degrees. In one step, data is aggregated while the planet rotates by d degrees, so each step gives information about a slice of d degrees of the planet's surface. The image is split into n segments of equal height, which are analyzed separately. So the slice of d degrees is partitioned into n areas $A_1, \ldots, A_n$. For each area A_i, image analysis produces a number that gives the percentage of A_i covered by water. The areas A_i for one step are highlighted in the diagram on the right.



You may assume the planet's surface is a sphere. This means each area $A_2, \ldots, A_{n-1}$ is a spherical quadrilateral: it has four vertices, two sides parallel to the equator (that is, in planes parallel to the equator's plane) and two sides on great circles through the planet's poles, where the great circles are d degrees apart. At either pole, two of the four vertices collapse into the pole, so A_1 and A_n are spherical triangles with only one side parallel to the equator. Due to the curvature of the surface, sides that are parallel to the equator are longer if they are closer to the equator, while sides on great circles are longer if they are closer to the poles.

The above process is repeated for the next d degrees of rotation, and so on, a total number of m times, until the whole surface of the planet has been covered (that is, $md = 360$ degrees). Your task is to compute the percentage of the planet's surface covered by water from the given data.

Input

The first line of input contains the two integers n and m ($2 \leq n, m \leq 1000$). Each of the following n lines contains m integers $a_{i,j}$ ($0 \leq a_{i,j} \leq 100$ for $1 \leq i \leq n$ and $1 \leq j \leq m$). Each column of this matrix describes the measurements for a single step, that is, a rotation by d degrees. The number $a_{i,j}$ is the percentage of area A_i that is covered by water in the j^{th} step.

Output

Output the percentage of the planet's surface covered by water. Your answer should have an absolute error of at most 10^{-6}.

Sample Input 1	Sample Output 1
3 7 63 61 55 54 77 87 89 73 60 38 5 16 56 91 75 43 11 3 16 20 95	51.809523810

Sample Input 2	Sample Output 2
4 3 10 10 10 10 10 10 10 10 10 10 10 10	10.000000000

Problem J
Bridging the Gap
Time limit: 4 seconds

A bridge with low capacity

A group of walkers arrives at a river in the night. They want to cross a bridge, which can hold a limited number of walkers at a time. The walkers have just one torch, which needs to be used when crossing the bridge. Each walker takes a certain time to cross; a group crossing together must walk at the slowest walker's pace. What is the shortest time it takes for all walkers to cross the bridge?

For example, Sample Input 1 assumes the bridge can hold 2 walkers at a time and there are 4 walkers with crossing times 1 minute, 2 minutes, 5 minutes and 10 minutes, respectively. The shortest time of 17 minutes can be achieved by the following sequence of crossings. First, the two fastest walkers cross in 2 minutes. Second, the fastest walker crosses back in 1 minute. Third, the two slowest walkers cross in 10 minutes. Fourth, the second-fastest walker crosses back in 2 minutes. Fifth, the two fastest walkers cross in 2 minutes.

Input

The first line of input contains two integers n and c, where n ($2 \le n \le 10^4$) is the number of walkers, and c ($2 \le c \le 10^4$) is the number of walkers the bridge can hold at a time.

Then follows a line containing n integers $t_1, \ldots, t_n$ ($1 \le t_i \le 10^9$ for all i). The i^{th} walker takes time t_i to cross.

Output

Output the minimum total time it takes for the entire group to cross the bridge.

Sample Input 1	Sample Output 1
4 2 1 2 10 5	17

Sample Input 2	Sample Output 2
4 6 1 2 10 5	10

This page is intentionally left blank.

Problem K
Alea Iacta Est
Time limit: 10 seconds

You play a game with multiple fair six-sided dice. Each die's face displays a single symbol. The objective of the game is to roll the dice and create a valid word from the symbols on top of each die. If you cannot form a word, you may reroll the dice for another attempt.

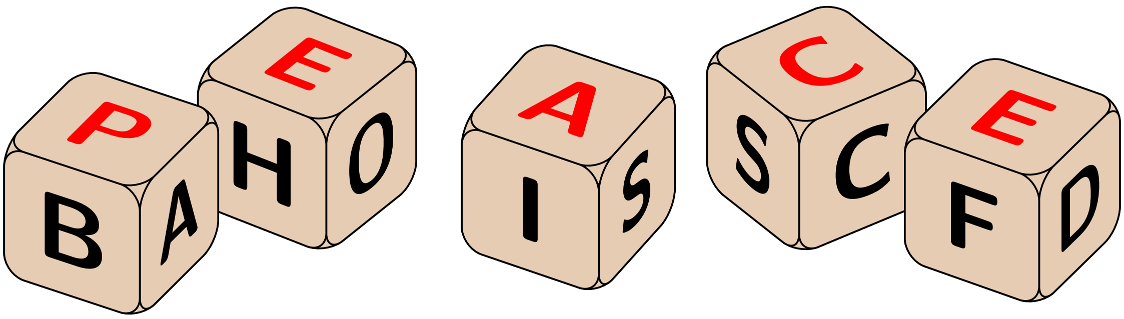


Figure K.1: Five dice making a valid word corresponding to Sample Input 1.

Suppose there are five dice: one of them contains letters `A`, `B`, `C`, `D`, `E`, and `P` (abbreviated as `ABCDEP`), and the other dice contain letters `AEHOXU`, `AISOLR`, `ABCDEF`, and `ABCSCC`. The first roll yields the following letters on the tops of respective dice: `P`, `X`, `R`, `E`, and `S`. As it is impossible to arrange these letters into a valid word, you decide to keep the `P`, `S`, and `E`, and reroll the other dice, in an attempt to make words like `PARSE`, `PAUSE`, `PHASE`, `POISE`, `PROSE`, `PULSE`, or `PURSE`. The two dice yield `E` and `A`, resulting in the following five letters: `P`, `E`, `A`, `E`, and `S`. You still cannot think of a valid word, so you decide to keep four letters and reroll only the last die, which has three sides with letter `C`. By doing so, there is a 50% chance that it will be possible to make a final valid word: `PEACE`, as shown in Figure K.1.

When you roll a die, it lands on any one of its faces with equal probability. What is the expected number of rolls needed to make a valid word, assuming you use an optimal strategy?

Input

The first line of input contains two numbers d and w, where d ($1 \le d \le 6$) is the number of dice and w ($1 \le w \le 2 \cdot 10^5$) is the number of valid words in the dictionary. The following d lines each have 6 symbols, one for each face of the die. The final w lines contain w distinct valid words in the dictionary. Every word has exactly d symbols.

All symbols in the input are either uppercase letters (`A`–`Z`) or digits (`0`–`9`).

Output

If it is possible to make a valid word, output the expected number of rolls needed to make a valid word when using an optimal strategy. Otherwise, output `impossible`. Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1

```
5 8
ABCDEP
AEHOXU
AISOLR
ABCDEF
ABCSCC
PARSE
PAUSE
PHASE
POISE
PROSE
PULSE
PURSE
PEACE
```

Sample Output 1

```
9.677887141
```

Sample Input 2

```
2 1
AAAAAA
BBBBBB
AB
```

Sample Output 2

```
1.0
```

Sample Input 3

```
3 1
123456
123456
123456
666
```

Sample Output 3

```
10.555444555
```

Sample Input 4

```
2 1
ABCDEF
GHI234
AB
```

Sample Output 4

```
impossible
```