Winter Contest 2026

January 31st

Problems

This page is intentionally left (almost) blank.

Problem A: Animal Appendages

Time limit: 1 second

THE Town Musicians of Bremen finally arrived in Bremen after their long journey. There, they played music in the streets and had a very successful first day. As night falls, they decide to stop and look for an inn, since taking over a robber's home would be far more difficult in a town.

The Town Musicians of Bremen, made by Gerhard Marcks. CC BY-SA 4.0 by Wuzur on Wikimedia Commons

Before they can choose an inn to stay at, they need to count the money they earned. They quickly realize that they earned more money than they can count with their paws, hooves, and feet. Kaja, a fellow street musician, notices their dire situation and offers them help. With their human hand Kaja shows the Town Musicians of Bremen how to count in binary from 0 up to 1023 with ten fingers.

The cat, the smartest of their group, notices that this principle can be extended even further. Instead of only extending or curling in fingers or toes, it may also be possible to curl only one of the joints or stretch them into a different direction. Having more than two distinguishable states for some fingers or toes allows them to count even higher.

The number of states fingers or toes can take varies greatly from animal to animal, and even from toe to toe. The cat and the dog both have five toes per paw, but their thumb is barely operable. The rooster has four fingers per foot, but none of them is very flexible. The donkey on the other hand has to get by with hooves only.

Given how many "active" distinguishable states their fingers or toes have, determine the maximum number they can reach counting up from zero. Here "active" states refers to states beyond the neutral state, that is the number of distinguishable gestures excluding a neutral, "do nothing" gesture.

Input

The input consists of:

- One line with ten integers $a_1, \ldots, a_{10}$ ($0 \leq a_i \leq 5$ for each i), the number of different "active" gesture states each finger or toe can represent.

If they have less than ten fingers or toes, there will still be ten numbers in the input, as the remaining numbers will be filled with zero.

Output

Output the maximum number they can count up to from zero with their fingers or toes.

Sample Input 1	Sample Output 1
`1 1 1 1 1 1 1 1 1 1`	`1023`

Sample Input 2

```
0 1 2 3 0 0 4 2 1 0
```

Sample Output 2

```
719
```

Sample Input 3

```
0 0 0 0 1 1 0 0 0 0
```

Sample Output 3

```
3
```

Problem B: Bewitched Broomstick

Time limit: 5 seconds

ONCE upon a time, the old Sorcerer left their apprentice to clean the house.

The old sorcerer has vanished
And for once has gone away!
Now be quick or do get punished.
Still we'll do a bit of play.
Come, old broomstick, come obey
Wipe the old and dusty floor.
Take the bucket, don't delay.
Fill it up and bring and pour.

Clean the house while I lie here,
Catching up the needed sleep.
Before long, I wake and peer
Oh my god, the flood runs deep.
Water surges to my knees
Broomstick sweeps like stormy seas.

Stop, I command, but what's the spell?
This never happened hitherto.
To my knees the waters swell
The flood is dire – what to do?
Of the spell, I know the start
The trailing letters I forget.
There's a very frequent part.
Help me, save me from this threat.

More precisely, the apprentice remembers the beginning of the spell s and a length ℓ. They now have to find the remaining $\ell - 1$ letters of the spell. They also know that the most frequently appearing substring[1] of length ℓ within the spell appears very often. Determine the last $\ell - 1$ letters that maximize the number of appearances of the most frequently appearing substring of length ℓ within the whole spell. If there are multiple valid options, you may output any one of them.

Input

The input consists of:

- One line with two integers n and ℓ ($2 \leq \ell < n \leq 2 \cdot 10^5$), the length of the beginning of the spell and the length of the frequently appearing part of the spell.
- One line with a string s, the beginning of the spell, consisting of n English lowercase letters (`a-z`).

[1]A substring of a string is a contiguous segment of characters from it.

Output

Output one spell consisting of $n+\ell-1$ English letters, that maximizes the number of appearances of the most frequently appearing substring of length ℓ. The first n letters should match the beginning of the spell s.

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

Sample Input 1

```
13 4
wintercontest
```

Sample Output 1

```
wintercontestest
```

The substring of length 4, "`test`", appears twice in "`wintercontestest`". It is possible to show that no substring of length 4 can appear more than twice.

Sample Input 2

```
6 4
ababab
```

Sample Output 2

```
ababababa
```

Sample Input 3

```
19 8
bewitchedbroomstick
```

Sample Output 3

```
bewitchedbroomstickblunder
```

Sample Input 4

```
6 5
helppp
```

Sample Output 4

```
helpppppp
```

Problem C: Cinderella's Chore

Time limit: 3 seconds

CINDERELLA's cruel stepmother and stepsisters often force her to do all kinds of hard work. They often mock her and sometimes even make up chores for her to do. This time, Cinderella wants to attend a local festival, but her stepmother refuses because Cinderella has no appropriate clothing.

When the girl insists, the woman guarantees her that she is going to give her permission if she manages to quickly complete a task. Specifically, Cinderella should place some flowers into porcelain pots arranged on a straight line in regular intervals. To make the task particularly infuriating and time-consuming, the woman does not tell her in which pots to put the flowers but only how far apart each pair of flowers should be.

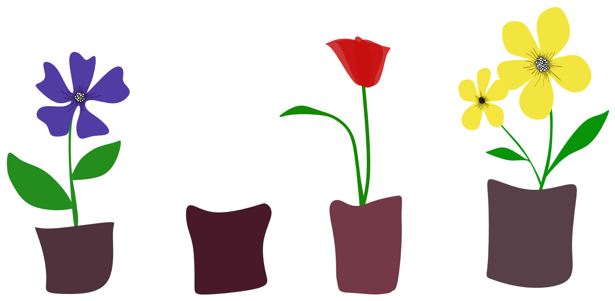


Figure C.1: Illustration of the first sample.

There are 10^{10} pots numbered 1 to 10^{10}, and flowers placed in pots i and j have distance $|i - j|$.

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 1000$), the number of flowers.
- n lines, the ith of which contains n integers $d_{i,1}, \ldots, d_{i,n}$ ($0 \leq d_{i,j} \leq 10^9$ for each j), where $d_{i,j}$ is the desired distance between flower i and flower j.

Integers on the diagonal are zero, all other entries are positive. It is further guaranteed that $d_{i,j} = d_{j,i}$ for all $1 \leq i, j \leq n$.

Output

If the flowers cannot be placed into the pots according to the stepmother's specification, output "`impossible`". Otherwise, output n integers, the ith of which is the pot Cinderella should put the ith flower.

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

Sample Input 1

```
3
0 1 2
1 0 3
2 3 0
```

Sample Output 1

```
3 4 1
```

Sample Input 2

```
2
0 1000000000
1000000000 0
```

Sample Output 2

```
1 1000000001
```

Sample Input 3

```
3
0 2 3
2 0 2
3 2 0
```

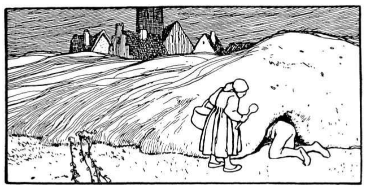
Sample Output 3

```
impossible
```

Problem D: Delicious Disaster

Time limit: 1 second

THE little girl Dana lives alone with her mother. They are not able to afford enough food, so the girl needs to go begging. An old woman gifts her a magic pot that starts cooking delicious, sweet porridge whenever the phrase "Cook, little pot, cook!" is uttered. All that is needed to end cooking is the phrase "Stop, little pot!".


Residents eat their way through the mound of porridge. Public Domain on Wikimedia Commons

Dana shows the pot to her mother, makes it start cooking using the magic phrase and leaves to go into town. However, she forgets to tell her mother the phrase needed to stop cooking, so now the pot keeps producing more and more porridge, and has enveloped the girl's home, as well as some of the surrounding homes.

The surroundings of Dana's home are modelled as an infinite grid of cells, with the girl's home located in cell $(0, 0)$. At any time, the porridge occupies all the cells that are at most r steps away from the girl's home when moving in the four cardinal directions.

Dana's mother has left the pot unattended for a while and now wants to figure out how far the porridge has spread. In other words, she wants to figure out r, the size of the porridge. She will shout out to people in the neighbourhood, asking them whether their houses are already surrounded by the porridge. Her shouting can only be heard in houses that are at most 1000 steps away, but she can find out the status of houses at a greater distance by arranging a relay of messages, where a group of people shout the message from one person to the next, with each two consecutive locations in the chain at most 1000 steps apart.

Formally, this means that her first question can be any location (x_1, y_1) such that $|x_1| + |y_1| \leq 1000$, her second question can be any location (x_2, y_2) such that $|x_2 - x_1| + |y_2 - y_1| \leq 1000$, her third question can be any location (x_3, y_3) such that $|x_3 - x_2| + |y_3 - y_2| \leq 1000$, and so on.

How can she coordinate the relays of messages in order to find out the size of the porridge while using at most 5000 questions? The current size r of the porridge is at most 10^6. However, the porridge keeps growing while questions are asked, and before every question, the porridge spreads by 1 step in every direction. An example of this process can be seen in Figure D.1.

Interaction

This is an interactive problem. Your submission will be run against an *interactor*, which reads from the standard output of your submission and writes to the standard input of your submission.

The interaction proceeds in rounds. In the kth round, you ask a query of the form "`?` x_k y_k", where (x_k, y_k) are the coordinates of the cell you want to ask about, subject to the rules described above. The interactor will respond with "`in`" if this cell is taken over by the porridge, and "`out`" otherwise. Formally, the porridge has an initial size of r_0 ($0 \leq r_0 \leq 10^6$) that is unknown to you, and the answer will be "`in`" if and only if $|x_k| + |y_k| \leq r_0 + k$.

Once you know the size of the porridge, you should output "`!` r", where r is the *current* size of the porridge. It is guaranteed that the size of the porridge does not change between your last '`?`' query and your '`!`' query. The interaction then ends and your program must exit.

You may send at most 5000 queries of type '`?`'.

After every request you should *flush* the standard output to ensure that the request is sent to the interactor. For example, you can use `fflush(stdout)` in C++, `System.out.flush()` in Java, `sys.stdout.flush()` in Python, and `hFlush stdout` in Haskell.

A testing tool is provided to help you develop your solution.

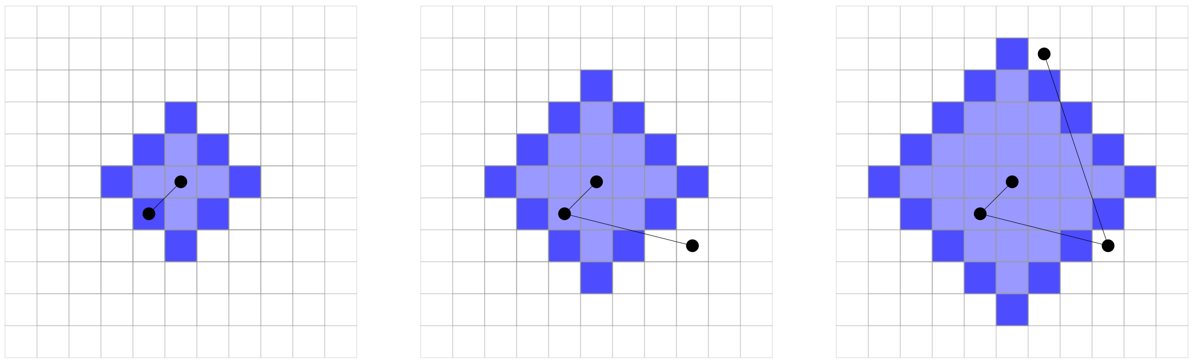


Figure D.1: Illustration of Sample Interaction 1. Initially, the size of the porridge is $r = 1$. Before the first query, this increases to $r = 2$, so the response to that query is "`in`", as can be seen in the left image. The other two images show the remaining two queries, where the light blue cells show the extent of the porridge before it grows, the dark blue cells are those that become swallowed as it grows, and the chain of connected dots corresponds to the relay of messages. The final size of the porridge is $r = 4$, which can be shown to be the only size consistent with the query responses.

Read	Sample Interaction 1	Write
		`? -1 -1`
`in`		
		`? 3 -2`
`out`		
		`? 1 4`
`out`		
		`! 4`

Read	Sample Interaction 2	Write
		`? 500 500`
`in`		
		`? 1500 500`
`in`		
		`? 2000 304`
`out`		
		`? 2000 304`
`in`		
		`! 2304`

Problem E: Evening Entertainment

Time limit: 1 second

A tailor had three sons. While away from home, each son has obtained a magical item. The first one obtained a magical table, that decks itself with the finest food and drinks when its owner says “Table, deck yourself!”. The second son got a magical donkey that produces gold out of its mouth when the owner says “Bricklebrit!”. The youngest son got a magical cudgel. When its owner says “Cudgel, out of the sack!”, the cudgel will clobber someone until its owner says “Cudgel, in the sack!”.

The youngest son on his way to the bar with the table, the donkey and the cudgel. Public Domain on Wikimedia

Ever since returning home, the three sons and their father go to the local bar. In the bar, the oldest son brings his table and proclaims “Table, deck yourself” so that they can eat and drink all night. The second son brings his donkey and says “Bricklebrit!” to produce gold as a gift for the barkeeper while they stay.

The regulars in the bar often play a trick-taking card game. The objective of the game is for each player to correctly bid on the number of tricks they themselves will take during the subsequent round of play.

The game is played in multiple rounds. Before each round, every player draws hand cards equal to the current round number. In other words: for the first round, every player draws one card. In the second round, every player draws two cards, and so forth. After the cards are drawn, the round starts. For each trick, every player plays a card and then a complicated set of rules is applied to decide which player takes the trick. The players play one trick after the other until they have no more cards. Afterwards either cards are drawn for the next round or the game ends.

After all cards of a round have been played, points are awarded to each player for a correct bid and subtracted for an incorrect bid. For a correct bid, 20 base points are awarded plus 10 points for each taken trick. For an incorrect bid, they lose 10 points for each trick they took above or below their bid. The points obtained in every round are summed up and the player with the most points after all rounds have been played is the winner.

It is a well-known fact among the players that this is a highly skilled game. Even with the worst cards, you can take all the tricks in a round, if you play smart. On the other hand, you can end up with no tricks taken even with the best cards.

One day a player suggests a challenge: using the fact that players can force an arbitrary distribution of tricks in any given round, can the group play n rounds of the game so that every player ends with 0 points?

Input

The input consists of:

- One line with two integers n and k ($1 \leq n \leq 60$, $1 \leq k \leq 6$), the number of rounds and the number of players.

It is guaranteed that there are enough cards in the deck to play n rounds.

Output

The output should consist of n lines, one for each round. Each line should contain 2 integers for each player, the number bid b_i ($0 \leq b_i \leq 10^6$) by the player followed by the number of tricks taken. Note that the total number of tricks taken in round i must be i. If this is not possible, output “`impossible`”.

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

Sample Input 1

```
3 3
```

Sample Output 1

```
1 0 0 0 0 1
1 1 2 1 1 0
0 2 2 1 0 0
```

Sample Input 2

```
1 4
```

Sample Output 2

```
impossible
```

Notes

For Sample 1, the point stats after each round are as follows:

Player:	1	2	3
Round 1	-10	20	-10
Round 2	20	10	-20
Round 3	0	0	0

Problem F: Forgotten Fragments

Time limit: 5 seconds

ALICE wakes up from her dream, convinced that there is more to Wonderland than it seems. The memory is already faint, so she decides to go back to sleep before the dream fades completely.

Alice identified n important dream fragments numbered 1 to n. When she goes back to sleep she will be able to move between fragments via $n-1$ connections that hold all fragments together. However, the evil Queen of Hearts will send her deck of guards to the fragment where Alice starts her dream and make it impassible for the rest of the dream. Alice has just enough time to revisit the events at the starting fragment and escape to a neighbouring fragment before the guards arrive. Then, Alice can move freely between fragments and revisit them as long as she does not go to the starting fragment again.

After Alice wakes up again, she can reason about fragments she revisited but forgets the rest. Each fragment contains a clue and there is a consistent train of thought to unravel the true meaning of Wonderland. She will follow the clues until they lead to a fragment she did not revisit. For example, if she revisited fragments numbered 1, 2, 3, 5, and 6, she can follow the truth until the clue of the 3rd fragment.

How much of the true meaning of Wonderland can Alice unravel? Since Alice does not know where her dream will start, find the answer for all possible starts.

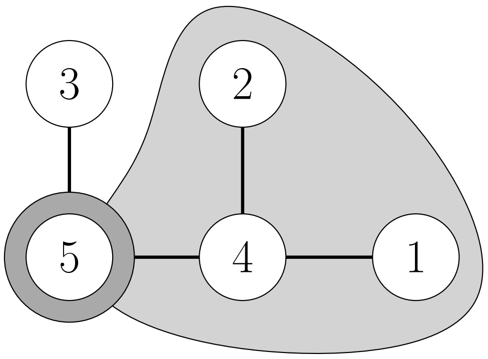


Figure F.1: Visualization of Sample Input 1. The dream starts in the fragment 5. Alice escapes to the subtree marked in gray to revisit clues 1, 2, 4, and 5. She cannot reach clue number 3 and only deciphers the meaning of the first two clues.

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 2 \cdot 10^5$), the number of dream fragments.
- $n-1$ lines with two integers u, v ($1 \leq u, v \leq n, u \neq v$), representing a connection between the dream fragments with the uth clue and vth clue.

It is guaranteed that each dream fragment is connected directly or indirectly to every other dream fragment.

Output

Output a line with n integers $a_1, \ldots, a_n$ ($1 \leq a_i \leq n$).

The value a_i should be the number of clues Alice can decipher if she starts her dream at the fragment with number i.

Sample Input 1

```
5
3 5
5 4
2 4
4 1
```

Sample Output 1

```
5 5 5 1 2
```

Sample Input 2

```
5
1 4
1 5
2 5
3 5
```

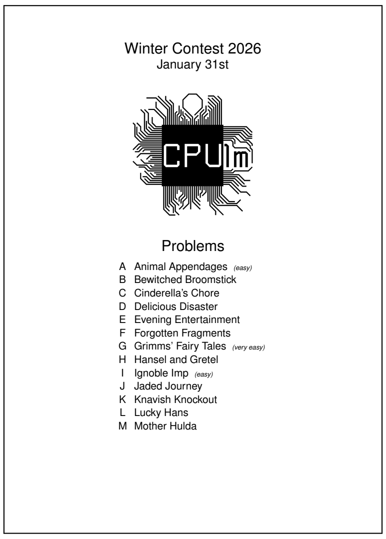
Sample Output 2

```
3 5 5 5 1
```

Problem G: Grimms' Fairy Tales

Time limit: 1 second

BROTHERS Grimm are excited to show you their newest project. They already finished their famous fairy tale bundles with over 200 fairy tales in total. This year they wanted a new challenge: programming problems incorporating various fairy tales. Over months and months they searched far and wide to gather the best fairy tale programming problems. The brothers want to publish their new bundle of 13 programming problems before the month of February arrives. With a few days to spare, the whole bundle was finished. Except, you, the editor noticed an important detail: they forgot to add page numbers in the table of contents. Please help them by finding out the right page number for each problem title.



Winter Contest 2026
January 31st

CPUlm

Problems

A	Animal Appendages *(easy)*
B	Bewitched Broomstick
C	Cinderella's Chore
D	Delicious Disaster
E	Evening Entertainment
F	Forgotten Fragments
G	Grimms' Fairy Tales *(very easy)*
H	Hansel and Gretel
I	Ignoble Imp *(easy)*
J	Jaded Journey
K	Knavish Knockout
L	Lucky Hans
M	Mother Hulda

The table of contents without page numbers.

Input

The input consists of:

- One line with a string s ($1 \leq |s| \leq 100$), the name of one of the problems in this contest.

The string s consists of English lowercase and uppercase letters (`a-z` and `A-Z`), apostrophes (`'`) and spaces. It is guaranteed that the title is the exact name of one of the problems in this contest, and is written exactly as in this document, including spacing and capitalization.

Output

Output the page number where the problem statement starts.

Sample Input 1

```
Animal Appendages
```

Sample Output 1

```
1
```

Sample Input 2

```
Grimms' Fairy Tales
```

Sample Output 2

```
13
```

This page is intentionally left (almost) blank.

Problem H: Hansel and Gretel

Time limit: 2 seconds

While the forest is a beautiful piece of nature, it is easy to get lost. Public Domain on Art Institute of Chicago

HANSEL and Gretel's parents are very poor. One night, Gretel overheard their parents discuss that they lack the money to feed their children. Reluctantly, they decide to abandon Hansel and Gretel in the nearby forest, which can be described as an undirected graph. Each vertex in the graph represents a clearing in the forest, whereas each edge represents a pathway that connects exactly two clearings. The graph is not necessarily connected.

To foil their parents' plan, Hansel and Gretel decide to chop down some trees to form a new clearing that serves as a point of orientation, so that they can find the way back to their parents house. In order to find this newly created clearing more easily, Hansel and Gretel may also create new pathways connecting the newly created clearing to some previously existing clearings. However, they cannot create pathways between two previously existing clearings.

After walking for a few hours inside the forest, Hansel and Gretel cannot tell the clearings apart anymore – they only know which pairs of clearings are connected by a pathway. In graph theoretic terms, they forgot the initial vertex labels and now they can only work with an isomorphic copy of the former graph.

Hansel and Gretel need your help to identify the clearing created by them, so that they can escape the forest and find back to their parents' house.

Input

This is a multi-pass problem. The submitted program will be invoked two times. The time limit is enforced separately for each pass. In the first pass, the input is a graph representing the initial layout of the forest. The program must decide which clearings should be connected by pathways to the new clearing created by Hansel and Gretel. In the second pass, the input is a graph representing the forest layout after creating a new clearing and connecting it to the other clearings according to the output of the first pass. In particular, the input of the second pass depends on the output of the first pass.

The input consists of:

- One line with an integer which is either 1 or 2, indicating the current pass.
- One line with two integers n and m, the number of clearings and the number of pathways.
- m lines with two integers u and v ($1 \leq u, v \leq n$, $u \neq v$), representing the pathways.

In the first pass, the bounds for n and m are $2 \leq n \leq 10^5$ and $0 \leq m \leq 2 \cdot 10^5$. In the second pass, a new clearing is added along with the d pathways from the program's output. If n' and m' were the values of n and m in the first pass, then in the second pass it holds that $n = n' + 1$ and $m = m' + d$.

It is guaranteed that each pair of clearings is connected by at most one pathway and that no clearing is connected to itself.

Output

In the first pass, output the number of clearings d ($0 \leq d \leq n$) to which the newly created clearing should be connected by a pathway, followed by d distinct integers $v_1, \ldots, v_d$ ($1 \leq v_i \leq n$ for each i), the labels of the clearings connected to the new clearing.

You may create at most one pathway between the new clearing and each other clearing. Furthermore, the newly created clearing can not be connected to itself.

In the second pass, output the label of the clearing that has been added in the first pass.

A testing tool is provided to help you develop your solution.

Sample Input 1 (Pass 1)

```
1
4 3
1 2
2 3
4 3
```

Sample Output 1 (Pass 1)

```
2
3 2
```

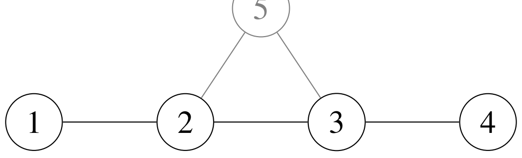
Sample Input 1 (Pass 2)

```
2
5 5
4 1
2 4
3 5
5 1
4 5
```

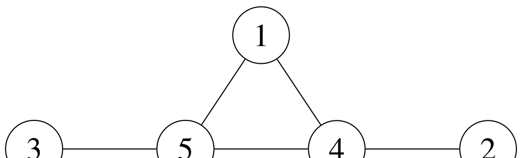
Sample Output 1 (Pass 2)

```
1
```

Consider the first pass of the sample. Initially, the forest consists of four clearings connected by three pathways. Hansel and Gretel add a new clearing and connect it to the clearings labelled 2 and 3. In the following visualization, this new clearing is labelled 5:



The input to the second pass describes the forest layout after adding the new clearing and the new pathways. However, the clearings are labelled differently:



From the graph structure, it can be inferred that the clearing with label 1 is the clearing which Hansel and Gretel added in pass 1.

Problem I: Ignoble Imp
Time limit: 1 second

ONCE upon a time, the Queen was forced to make a very disadvantageous deal with a little man that spun straw into gold. For his service, the Queen had to promise to give her firstborn child to the little man.

Years later, when the Queen's first child is born, the little man returns and demands his payment. She of course wants to keep her child, and after much begging and bargaining, he finally agrees to leave her child if she can guess his name within three days.

In the final night, after all her failed guesses, she wanders into the woods and hears him sing:

Tonight, tonight, my plans I make.
Tomorrow, tomorrow, the baby I take.
The Queen will never win the game,
for hard to discover is my name.

Through endless nights I heard her plea,
she solves my riddle – she'll be free.
Two words I revealed to her in play,
for games delight me more than pay.

From my name those words are spun,
neither is farther than the other one.
Not twice a place is changed in both,
never or once, I swear an oath.

I'd like to follow the old tradition,
if they agree in one position,
then my name is there the same,
for not to make it an unfair game.

Finally, a clue! From what the little man sang, she concludes that she already heard two words of equal length n from which the man's name can be derived. Specifically, the maximum number of different positions between either word and the man's name should be minimal.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^5$), the length of the two words.
- Two lines, each containing a word of length n, consisting only of English lowercase letters (`a-z`).

Output

Output a string of length n, consisting only of English lowercase letters (`a-z`), that could be the little man's name.

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

Sample Input 1

```
7
chances
charted
```

Sample Output 1

```
charles
```

Sample Input 2

```
3
aaa
bbb
```

Sample Output 2

```
abz
```

Sample Input 3

```
1
a
b
```

Sample Output 3

```
b
```

Sample Input 4

```
15
rumpelstiltskin
rumpelstiltskin
```

Sample Output 4

```
rumpelstiltskin
```

Problem J: Jaded Journey

Time limit: 1 second

Painting of a baghlah, a large deep-sea traditional Arabic sailing vessel. CC BY-SA 4.0 by Xavier Romero-Frias on Wikimedia Commons

HINDBAD'S life was transformed during the seven days he spent with Sindbad the Sailor. As he listened to the stories of Sindbad's seven voyages of distant seas and strange lands, an unfamiliar world unfolded before his eyes. Wanting to repay Sindbad for his generosity and companionship, Hindbad resolved to take him on one final journey, Sindbad's eighth voyage.

This final journey entails an n farsakh[2] long sea travel. As Hindbad was not nearly as financially endowed as his companion, the ship he hired for the voyage had some clear deficiencies. To make matters worse, Sindbad's reputation as the lone survivor of so many voyages had preceded him. The crew, convinced they were ferrying a walking ill omen, demanded unreasonable extra compensation in exchange for letting him sail with them.

The ship has two means of moving. The crew can row the ship with rudders, but for each farsakh they row they demand x additional food ration packs for their effort. When there is wind, the sail can be used to move instead. Using the sail to travel will not cost Hindbad anything, but since he hired a cheap ship, the sail breaks down frequently. He can ask the crew to fix the sail for r extra ration packs as often as necessary. Unfortunately, whenever the sail is repaired, it will break after d further farsakhs, regardless of how much it is used during this time. Initially, the sail is broken.

Given an accurate forecast on where there will be wind and where there will not, determine the minimum number of extra food ration packs Hindbad needs to load onto the ship.

Input

The input consists of:

- One line with four integers n, x, r, and d ($1 \leq d \leq n \leq 2 \cdot 10^5, 1 \leq x, r \leq 10^9$), the distance you need to travel, the required extra food packs for rowing one farsakh, the required food packs to repair the sail, and the distance until the sail breaks after repair.
- One line with n integers $w_1, \ldots, w_n$ ($0 \leq w_i \leq 1$ for each i), with w_i being 1 if there is wind to use the sail in the ith farsakh of the journey, and 0 otherwise.

Output

Output the minimum extra food ration packs Hindbad needs to load onto the ship.

Sample Input 1	Sample Output 1
3 2 1 3 0 0 0	6

Since there is no wind in this sample, the crew has to row all three farsakhs. Therefore the total extra food ration packs needed is $3 \cdot x = 6$.

[2]An old distance unit used in the Middle East, roughly equivalent to 6 km

Sample Input 2

```
4 3 5 2
1 0 1 1
```

Sample Output 2

```
11
```

Here it is optimal to let the crew row the first two farsakhs and then repair the sail for the last two farsakhs.

Sample Input 3

```
7 2 3 3
0 1 1 1 1 0 1
```

Sample Output 3

```
10
```

Problem K: Knavish Knockout

Time limit: 2 seconds

Richilde selling her apples.
Public Domain on Wikimedia Commons

RICHILDE has just received shocking news from her magic mirror. Not only is Blanca, her unbeloved stepchild, alive and well, but on top of that, she is still the fairest of them all! Richilde, who has already made several failed attempts to get rid of Blanca using poisoned soaps and letters, is furious. This time, she decides to take care of the problem on her own.

Richilde knows that Blanca lives in a cottage in a far away forest together with $n-1$ dwarfs. She plans to travel to this forest to execute the following plan: for k consecutive days, Richilde will visit the cottage disguised as an old peddler and sell n poisoned apples for them to eat at their dinner.

With help from her court physician Sambul, Richilde has collected $n \cdot k$ apples from his special apple tree, which bears fruit ingrained with a slow acting poison. Each apple has a certain size and contains a certain amount of poison. Since dwarfs are known for their hearty appetite and since she knows Blanca to be quite modest, Richilde is confident that each day at dinner, Blanca will eat the smallest apple that was sold on that day. The remaining $n-1$ apples will be eaten by the dwarfs. Of course, Richilde wants to maximize the total amount of poison that is consumed by Blanca. To achieve this, she will carefully select which n apples to sell on each of the k days.

Unbeknownst to Richilde, Sambul has long ago ensured that all poisonous soaps, letters and apples that Richilde might use in her wicked plans are actually laced with a non-lethal sleeping poison. He is therefore confident that Blanca will yet again survive Richilde's latest attempt. However, he is still worried about Blanca's health and wonders how much of the sleeping poison she will consume in total. What is the maximum total amount of poison that Blanca will consume, given that Richilde can choose which apples to sell on each day?

Input

The input consists of:

- One line with two integers n, k ($1 \leq n, k \leq 10^5$, $n \cdot k \leq 2 \cdot 10^5$), the number of apples sold each day and the number of days.
- $n \cdot k$ lines, the ith of which contains two integers s_i, p_i ($1 \leq s_i, p_i \leq 10^9$), the size and poison amount of the ith apple.

It is guaranteed that all sizes s_i are distinct.

Output

Output the maximum total amount of poison that will be consumed by Blanca (the total amount is the sum over the poison values of all apples she will eat).

Sample Input 1

```
2 3
1 10
2 10
3 10
6 5
7 5
8 5
```

Sample Output 1

```
30
```

Sample Input 2

```
3 3
1 8
2 5
9 4
8 8
6 7
7 4
3 8
4 4
5 6
```

Sample Output 2

```
23
```

Problem L: Lucky Hans

Time limit: 2 seconds

HANS had served his master for seven years, and as reward, Hans received a piece of gold as big as his head. On his way home, Hans met a lot of people offering him to trade. Being a miserable merchant, Hans eventually ended up with some object of significantly smaller value than the piece of gold.

Hans accepting one bad trade after the other.
Public Domain on Wikimedia Commons

More precisely, the situation could be described as follows: there were n objects of different value. The piece of gold initially owned by Hans was the most valuable of these objects. Also, there were m trade offers available to Hans. A trade offer was a pair (x, y) of objects meaning that Hans could exchange object x for object y. Since the other person involved in the trade was less naive than Hans, the value of y was always less than the value of x.

When Hans owned object x and there was at least one trade offer available for x, then Hans luckily accepted such an offer to get rid of x. However, Hans did not pick an arbitrary offer. Out of all available trade offers (x, y), he picked the one where y was the most valuable. Hans performed this until there were no trade offers available anymore.

By the time the Grimm brothers were writing down Hans' story, it was long forgotten how valuable the different objects were back then. The only thing still known for sure was that the piece of gold was the most valuable object, and that for every trade offer (x, y), the object y was less valuable than object x. While the trade offers available to Hans were still known, it was unknown which trade offers he had accepted. Also, it was unknown which object Hans owned in the end when no trade offer had been available anymore. The Grimm brothers only knew that this object was the kth least valuable.

Help the Grimm brothers by figuring out how valuable the objects could have been back in Hans' days such that Hans could have ended up with the kth least valuable object or determine that the information gathered by the Grimm brothers is contradictory.

Input

The input consists of:

- One line with three integers n, m, and k ($2 \leq n \leq 3000$, $1 \leq m \leq 9000$, $1 \leq k \leq n$), the number of objects, the number of trade offers and the relative value of the object Hans ends up with.
- m lines with two integers x and y ($1 \leq x, y \leq n$, $x \neq y$), each describing a trade offer: object x can be traded for object y.

It is guaranteed that the trade offers are distinct and acyclic. The piece of gold is the object with number 1 and it is guaranteed that every object can be obtained when starting with the piece of gold.

Output

If Hans could not have ended up with the kth least valuable object, output "`impossible`". Otherwise, output a permutation $v_1, \ldots, v_n$ meaning that object i was the v_ith least valuable.

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

Sample Input 1

```
4 3 2
1 2
1 3
3 4
```

Sample Output 1

```
4 1 3 2
```

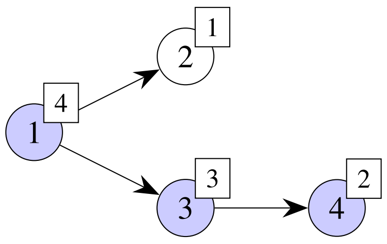
Sample Input 2

```
4 4 2
1 2
2 3
2 4
4 3
```

Sample Output 2

```
impossible
```

Consider the first sample input with the object values given in the sample output. In the following illustration, the objects owned by Hans during the trading process are marked in blue:



Initially, Hans owns the object labelled 1, i.e., the large piece of gold. He is offered to trade it for object 2 or object 3. He trades it for object 3 since this object is more valuable. After this, he is offered to trade object 3 for object 4. Since there are no other available trade offers, he accepts this one. When he owns object 4, he does not receive any trade offer for it. Thus, Hans ends up with object 4 which is the second least valuable.

Problem M: Mother Hulda
Time limit: 2 seconds

Illustration by Hermann Vogel. Public Domain, Leipziger Turm-Verlag via Wikimedia Commons

ONCE upon a time, a girl lost a spindle in a well. Her stepmother made her fetch the spindle, and the girl had to jump into the well. After that she knew nothing, and when she came to herself, she was in a beautiful meadow where she met an old woman. The woman introduced herself as Mother Hulda, and asked the girl to live with her. She promised that things would go well with the girl if she only did the woman's housework.

From then on, the girl did everything to the woman's satisfaction, and shook the bed with such a will that the feathers flew about like snowflakes.

After many days, the girl began to feel homesick. Before letting her go, Mother Hulda gave her one final task: the girl was to place some figurines on a tiled floor, but Mother Hulda believed that figuring out on which tile to place which figurine was part of the task. So she gave the girl only the distances that each pair of figurines should have.

The tiles form a $10^9 \times 10^9$ grid, where the rows and columns are numbered 1 through 10^9. The distance between a figurine placed on the tile in row i and column j and one on the tile in row i' and column j' is $\sqrt{(i-i')^2+(j-j')^2}$.

Input

The input consists of:

- One line with an integer n ($2 \le n \le 500$), the number of figurines.
- n lines, the ith of which contains n integers $d_{i,1}, \dots, d_{i,n}$ ($0 \le d_{i,j} \le 10^9$ for each j), where $d_{i,j}$ is the *square* of the desired Euclidean distance between figurines i and j.

Integers on the diagonal are zero, all other entries are positive. It is further guaranteed that $d_{i,j} = d_{j,i}$ for all $1 \le i, j \le n$.

Output

If the figurines cannot be placed onto the tiles according to Mother Hulda's specification, output "`impossible`".

Otherwise, output the n coordinates of the figurines.

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

Sample Input 1

```
3
0 1 1
1 0 2
1 2 0
```

Sample Output 1

```
1 1
1 2
2 1
```

Sample Input 2

```
2
0 1000000000
1000000000 0
```

Sample Output 2

```
1 1
1201 31601
```

Sample Input 3

```
3
0 7 13
7 0 17
13 17 0
```

Sample Output 3

```
impossible
```