A

Factory Repairs

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A factory produces thimbles in bulk. Typically, it can produce up to a thimbles a day. However, some of the machinery is defective, so it can currently only produce b thimbles each day. The factory intends to choose a k-day period to do maintenance and construction; it cannot produce any thimbles during this time, but will be restored to its full production of a thimbles per day after the k days are complete.

Initially, no orders are pending. The factory receives updates of the form d_i, a_i, indicating that a_i new orders have been placed for the d_i-th day. Each order requires a single thimble to be produced on precisely the specified day. The factory may opt to fill as many or as few of the orders in a single batch as it likes.

As orders come in, the factory owner would like to know the maximum number of orders he will be able to fill if he starts repairs on a given day p_i. Help the owner answer his questions.

Input

The first line contains five integers n, k, a, b, and q ($1 \le k \le n \le 200\,000$, $1 \le b < a \le 10\,000$, $1 \le q \le 200\,000$) — the number of days, the length of the repair time, the production rates of the factory, and the number of updates, respectively.

The next q lines contain the descriptions of the queries. Each query is of one of the following two forms:

- $1\ d_i\ a_i$ ($1 \le d_i \le n$, $1 \le a_i \le 10\,000$), representing an update of a_i orders on day d_i, or
- $2\ p_i$ ($1 \le p_i \le n - k + 1$), representing a question: at the moment, how many orders could be filled if the factory decided to commence repairs on day p_i?

It's guaranteed that the input will contain at least one query of the second type.

Output

For each query of the second type, print a line containing a single integer — the maximum number of orders that the factory can fill over all n days.

Examples

input
5 2 2 1 8 1 1 2 1 5 3 1 2 1 2 2 1 4 2 1 3 2 2 1 2 3

output
3 6 4

input
5 4 10 1 6 1 1 5 1 5 5 1 3 2 1 5 2 2 1 2 2

output
7 1

Note

Consider the first sample.

We produce up to 1 thimble a day currently and will produce up to 2 thimbles a day after repairs. Repairs take 2 days.

For the first question, we are able to fill 1 order on day 1, no orders on days 2 and 3 since we are repairing, no orders on day 4 since no thimbles have been ordered for that day, and 2 orders for day 5 since we are limited to our production capacity, for a total of 3 orders filled.

For the third question, we are able to fill 1 order on day 1, 1 order on day 2, and 2 orders on day 5, for a total of 4 orders.

B Secret Combination

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You got a box with a combination lock. The lock has a display showing n digits. There are two buttons on the box, each button changes digits on the display. You have quickly discovered that the first button adds 1 to all the digits (all digits 9 become digits 0), and the second button shifts all the digits on the display one position to the right (the last digit becomes the first one). For example, if the display is currently showing number `579`, then if we push the first button, the display will show `680`, and if after that we push the second button, the display will show `068`.

You know that the lock will open if the display is showing the smallest possible number that can be obtained by pushing the buttons in some order. The leading zeros are ignored while comparing numbers. Now your task is to find the desired number.

Input

The first line contains a single integer n ($1 \le n \le 1000$) — the number of digits on the display.

The second line contains n digits — the initial state of the display.

Output

Print a single line containing n digits — the desired state of the display containing the smallest possible number.

Examples

input
3 579
output
024

input
4 2014
output
0142

C Appleman and Easy Task

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Toastman came up with a very easy task. He gives it to Appleman, but Appleman doesn't know how to solve it. Can you help him?

Given a $n \times n$ checkerboard. Each cell of the board has either character 'x', or character 'o'. Is it true that each cell of the board has even number of adjacent cells with 'o'? Two cells of the board are adjacent if they share a side.

Input

The first line contains an integer n ($1 \le n \le 100$). Then n lines follow containing the description of the checkerboard. Each of them contains n characters (either 'x' or 'o') without spaces.

Output

Print "YES" or "NO" (without the quotes) depending on the answer to the problem.

Examples

input
3 xxo xox oxx
output
YES

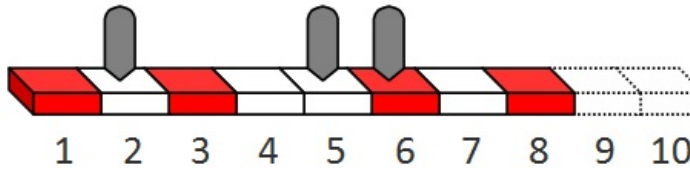
input
4 xxxo xoxo oxox xxxx
output
NO

D Read Time

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Mad scientist Mike does not use slow hard disks. His modification of a hard drive has not one, but n different heads that can read data in parallel.

When viewed from the side, Mike's hard drive is an endless array of tracks. The tracks of the array are numbered from left to right with integers, starting with 1. In the initial state the i-th reading head is above the track number h_i. For each of the reading heads, the hard drive's firmware can move the head exactly one track to the right or to the left, or leave it on the current track. During the operation each head's movement does not affect the movement of the other heads: the heads can change their relative order; there can be multiple reading heads above any of the tracks. A track is considered read if at least one head has visited this track. In particular, all of the tracks numbered h_1, h_2, ..., h_n have been read at the beginning of the operation.



Mike needs to read the data on m distinct tracks with numbers p_1, p_2, ..., p_m. Determine the minimum time the hard drive firmware needs to move the heads and read all the given tracks. Note that an arbitrary number of other tracks can also be read.

Input

The first line of the input contains two space-separated integers n, m ($1 \le n, m \le 10^5$) — the number of disk heads and the number of tracks to read, accordingly. The second line contains n distinct integers h_i in ascending order ($1 \le h_i \le 10^{10}$, $h_i < h_{i+1}$) — the initial positions of the heads. The third line contains m distinct integers p_i in ascending order ($1 \le p_i \le 10^{10}$, $p_i < p_{i+1}$) - the numbers of tracks to read.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is recommended to use the `cin`, `cout` streams or the `%I64d` specifier.

Output

Print a single number — the minimum time required, in seconds, to read all the needed tracks.

Examples

input
3 4 2 5 6 1 3 6 8
output
2

input
3 3 1 2 3 1 2 3
output
0

input
1 2 165 142 200
output
81

Note

The first test coincides with the figure. In this case the given tracks can be read in 2 seconds in the following way:

1. during the first second move the 1-st head to the left and let it stay there;
2. move the second head to the left twice;
3. move the third head to the right twice (note that the 6-th track has already been read at the beginning).

One cannot read the tracks in 1 second as the 3-rd head is at distance 2 from the 8-th track.

E Guess the array

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is an interactive problem. You should use `flush` *operation after each printed line. For example, in C++ you should use* `fflush(stdout)`*, in Java you should use* `System.out.flush()`*, and in Pascal —* `flush(output)`*.*

In this problem you should guess an array a which is unknown for you. The only information you have initially is the length n of the array a.

The only allowed action is to ask the sum of two elements by their indices. Formally, you can print two indices i and j (the indices should be **distinct**). Then your program should read the response: the single integer equals to $a_i + a_j$.

It is easy to prove that it is always possible to guess the array using at most n requests.

Write a program that will guess the array a by making at most n requests.

Interaction

In each test your program should guess a single array.

The input starts with a line containing integer n ($3 \le n \le 5000$) — the length of the array. Your program should read it at first.

After that your program should print to the standard output the requests about the sum of two elements or inform that the array is guessed.

- In case your program is making a request to ask the sum of two elements, it should print line in the format "`? i j`" (i and j are distinct integers between 1 and n), where i and j are indices in the array a.
- In case your program informs that the array is guessed, it should print line in the format "`!` a_1 a_2 ... a_n" (it is guaranteed that all a_i are positive integers not exceeding 10^5), where a_i is the i-th element of the array a.

The response on a request is a single integer equal to $a_i + a_j$, printed on a separate line.

Your program can do at most n requests. Note that the final line «`!` a_1 a_2 ... a_n» is not counted as a request.

Do not forget about `flush` operation after each printed line.

After you program prints the guessed array, it should terminate normally.

Example

input
5 9 7 9 11 6

output
? 1 5 ? 2 3 ? 4 1

```
? 5 2

? 3 4

! 4 6 1 5 5
```

Note

The format of a test to make a hack is:

- The first line contains an integer number n ($3 \le n \le 5000$) — the length of the array.
- The second line contains n numbers $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^5$) — the elements of the array to guess.

F Dexterina's Lab

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Dexterina and Womandark have been arch-rivals since they've known each other. Since both are super-intelligent teenage girls, they've always been trying to solve their disputes in a peaceful and nonviolent way. After god knows how many different challenges they've given to one another, their score is equal and they're both desperately trying to best the other in various games of wits. This time, Dexterina challenged Womandark to a game of Nim.

Nim is a two-player game in which players take turns removing objects from distinct heaps. On each turn, a player must remove at least one object, and may remove any number of objects from a single heap. The player who can't make a turn loses. By their agreement, the sizes of piles are selected randomly from the range $[0, x]$. Each pile's size is taken independently from the same probability distribution that is known before the start of the game.

Womandark is coming up with a brand new and evil idea on how to thwart Dexterina's plans, so she hasn't got much spare time. She, however, offered you some tips on looking fabulous in exchange for helping her win in Nim. Your task is to tell her what is the probability that the first player to play wins, given the rules as above.

Input

The first line of the input contains two integers n ($1 \le n \le 10^9$) and x ($1 \le x \le 100$) — the number of heaps and the maximum number of objects in a heap, respectively. The second line contains $x + 1$ real numbers, given with up to 6 decimal places each: $P(0), P(1), \ldots, P(X)$. Here, $P(i)$ is the probability of a heap having exactly i objects in start of a game. It's guaranteed that the sum of all $P(i)$ is equal to 1.

Output

Output a single real number, the probability that the first player wins. The answer will be judged as correct if it differs from the correct answer by at most 10^{-6}.

Example

input
2 2 0.500000 0.250000 0.250000
output
0.62500000

G Appleman and Toastman

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Appleman and Toastman play a game. Initially Appleman gives one group of n numbers to the Toastman, then they start to complete the following tasks:

- Each time Toastman gets a group of numbers, he sums up all the numbers and adds this sum to the score. Then he gives the group to the Appleman.
- Each time Appleman gets a group consisting of a single number, he throws this group out. Each time Appleman gets a group consisting of more than one number, he splits the group into two non-empty groups (he can do it in any way) and gives each of them to Toastman.

After guys complete all the tasks they look at the score value. What is the maximum possible value of score they can get?

Input

The first line contains a single integer n ($1 \le n \le 3 \cdot 10^5$). The second line contains n integers a_1, a_2, ..., a_n ($1 \le a_i \le 10^6$) — the initial group that is given to Toastman.

Output

Print a single integer — the largest possible score.

Examples

input
3 3 1 5
output
26

input
1 10
output
10

Note

Consider the following situation in the first example. Initially Toastman gets group [3, 1, 5] and adds 9 to the score, then he give the group to Appleman. Appleman splits group [3, 1, 5] into two groups: [3, 5] and [1]. Both of them should be given to Toastman. When Toastman receives group [1], he adds 1 to score and gives the group to Appleman (he will throw it out). When Toastman receives group [3, 5], he adds 8 to the score and gives the group to Appleman. Appleman splits [3, 5] in the only possible way: [5] and [3]. Then he gives both groups to Toastman. When Toastman receives [5], he adds 5 to the score and gives the group to Appleman (he will throws it out). When Toastman receives [3], he adds 3 to the score and gives the group to Appleman (he will throws it out). Finally Toastman have added 9 + 1 + 8 + 5 + 3 = 26 to the score. This is the optimal sequence of actions.

H DZY Loves Colors

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

DZY loves colors, and he enjoys painting.

On a colorful day, DZY gets a colorful ribbon, which consists of n units (they are numbered from 1 to n from left to right). The color of the i-th unit of the ribbon is i at first. It is colorful enough, but we still consider that the colorfulness of each unit is 0 at first.

DZY loves painting, we know. He takes up a paintbrush with color x and uses it to draw a line on the ribbon. In such a case some contiguous units are painted. Imagine that the color of unit i currently is y. When it is painted by this paintbrush, the color of the unit becomes x, and the colorfulness of the unit increases by $|x - y|$.

DZY wants to perform m operations, each operation can be one of the following:

1. Paint all the units with numbers between l and r (both inclusive) with color x.
2. Ask the sum of colorfulness of the units between l and r (both inclusive).

Can you help DZY?

Input

The first line contains two space-separated integers n, m $(1 \le n, m \le 10^5)$.

Each of the next m lines begins with a integer $type$ $(1 \le type \le 2)$, which represents the type of this operation.

If $type = 1$, there will be 3 more integers l, r, x $(1 \le l \le r \le n;\ 1 \le x \le 10^8)$ in this line, describing an operation 1.

If $type = 2$, there will be 2 more integers l, r $(1 \le l \le r \le n)$ in this line, describing an operation 2.

Output

For each operation 2, print a line containing the answer — sum of colorfulness.

Examples

input
3 3 1 1 2 4 1 2 3 5 2 1 3
output
8

input
3 4 1 1 3 4 2 1 1 2 2 2 2 3 3
output
3 2 1

input
10 6 1 1 5 3 1 2 7 9 1 10 10 11 1 3 8 12

```
1 1 10 3
2 1 10
```

output

```
129
```

Note

In the first sample, the color of each unit is initially $[1, 2, 3]$, and the colorfulness is $[0, 0, 0]$.

After the first operation, colors become $[4, 4, 3]$, colorfulness become $[3, 2, 0]$.

After the second operation, colors become $[4, 5, 5]$, colorfulness become $[3, 3, 2]$.

So the answer to the only operation of type 2 is 8.

Igor and his way to work

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Woken up by the alarm clock Igor the financial analyst hurried up to the work. He ate his breakfast and sat in his car. Sadly, when he opened his GPS navigator, he found that some of the roads in Bankopolis, the city where he lives, are closed due to road works. Moreover, Igor has some problems with the steering wheel, so he can make **no more than two turns** on his way to his office in bank.

Bankopolis looks like a grid of n rows and m columns. Igor should find a way from his home to the bank that has no more than two turns and doesn't contain cells with road works, or determine that it is impossible and he should work from home. A turn is a change in movement direction. Igor's car can only move to the left, to the right, upwards and downwards. Initially Igor can choose any direction. Igor is still sleepy, so you should help him.

Input

The first line contains two integers n and m ($1 \le n, m \le 1000$) — the number of rows and the number of columns in the grid.

Each of the next n lines contains m characters denoting the corresponding row of the grid. The following characters can occur:

- "`.`" — an empty cell;
- "`*`" — a cell with road works;
- "`S`" — the cell where Igor's home is located;
- "`T`" — the cell where Igor's office is located.

It is guaranteed that "`S`" and "`T`" appear exactly once each.

Output

In the only line print "`YES`" if there is a path between Igor's home and Igor's office with no more than two turns, and "`NO`" otherwise.

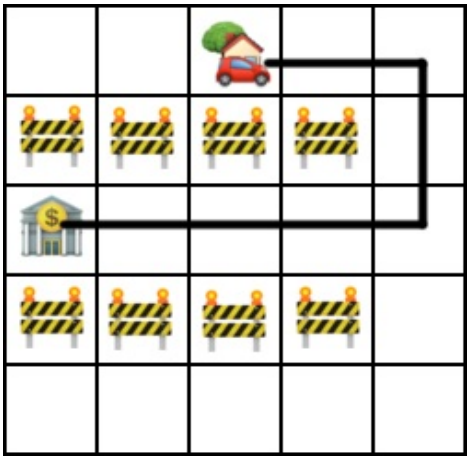
Examples

input
5 5 ..S.. ****. T.... ****. .....
output
YES

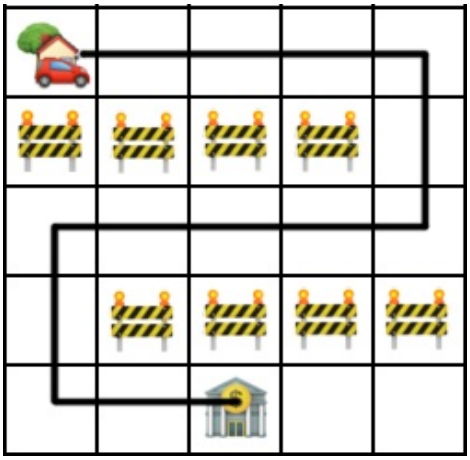
input
5 5 S.... ****. ..... .**** ..T..
output
NO

Note

The first sample is shown on the following picture:



In the second sample it is impossible to reach Igor's office using less that 4 turns, thus there exists no path using no more than 2 turns. The path using exactly 4 turns is shown on this picture:



Porcelain

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

During her tantrums the princess usually smashes some collectable porcelain. Every furious shriek is accompanied with one item smashed.

The collection of porcelain is arranged neatly on n shelves. Within each shelf the items are placed in one row, so that one can access only the outermost items — the leftmost or the rightmost item, not the ones in the middle of the shelf. Once an item is taken, the next item on that side of the shelf can be accessed (see example). Once an item is taken, it can't be returned to the shelves.

You are given the values of all items. Your task is to find the maximal damage the princess' tantrum of m shrieks can inflict on the collection of porcelain.

Input

The first line of input data contains two integers n ($1 \le n \le 100$) and m ($1 \le m \le 10000$). The next n lines contain the values of the items on the shelves: the first number gives the number of items on this shelf (an integer between 1 and 100, inclusive), followed by the values of the items (integers between 1 and 100, inclusive), in the order in which they appear on the shelf (the first number corresponds to the leftmost item, the last one — to the rightmost one). The total number of items is guaranteed to be at least m.

Output

Output the maximal total value of a tantrum of m shrieks.

Examples

input
2 3 3 3 7 2 3 4 1 5
output
15

input
1 3 4 4 3 1 2
output
9

Note

In the first case there are two shelves, each with three items. To maximize the total value of the items chosen, one can take two items from the left side of the first shelf and one item from the right side of the second shelf.

In the second case there is only one shelf, so all three items are taken from it — two from the left side and one from the right side.

K

Buggy Robot

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Ivan has a robot which is situated on an infinite grid. Initially the robot is standing in the starting cell $(0, 0)$. The robot can process commands. There are four types of commands it can perform:

- `U` — move from the cell (x, y) to $(x, y + 1)$;
- `D` — move from (x, y) to $(x, y - 1)$;
- `L` — move from (x, y) to $(x - 1, y)$;
- `R` — move from (x, y) to $(x + 1, y)$.

Ivan entered a sequence of n commands, and the robot processed it. After this sequence the robot ended up in the starting cell $(0, 0)$, but Ivan doubts that the sequence is such that after performing it correctly the robot ends up in the same cell. He thinks that some commands were ignored by robot. To acknowledge whether the robot is severely bugged, he needs to calculate the maximum possible number of commands that were performed correctly. Help Ivan to do the calculations!

Input

The first line contains one number n — the length of sequence of commands entered by Ivan $(1 \le n \le 100)$.

The second line contains the sequence itself — a string consisting of n characters. Each character can be `U`, `D`, `L` or `R`.

Output

Print the maximum possible number of commands from the sequence the robot could perform to end up in the starting cell.

Examples

input
4 LDUR
output
4

input
5 RRRUU
output
0

input
6 LLRRRR
output
4

April Fool's Problem

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The marmots need to prepare k problems for HC^2 over n days. Each problem, once prepared, also has to be printed.

The preparation of a problem on day i (at most one per day) costs a_i CHF, and the printing of a problem on day i (also at most one per day) costs b_i CHF. Of course, a problem cannot be printed before it has been prepared (but doing both on the same day is fine).

What is the minimum cost of preparation and printing?

Input

The first line of input contains two space-separated integers n and k ($1 \le k \le n \le 2200$). The second line contains n space-separated integers $a_1, ..., a_n$ ($1 \le a_i \le 10^9$) — the preparation costs. The third line contains n space-separated integers $b_1, ..., b_n$ ($1 \le b_i \le 10^9$) — the printing costs.

Output

Output the minimum cost of preparation and printing k problems — that is, the minimum possible sum $a_{i_1} + a_{i_2} + ... + a_{i_k} + b_{j_1} + b_{j_2} + ... + b_{j_k}$, where $1 \le i_1 < i_2 < ... < i_k \le n$, $1 \le j_1 < j_2 < ... < j_k \le n$ and $i_1 \le j_1$, $i_2 \le j_2$, ..., $i_k \le j_k$.

Example

input
8 4 3 8 7 9 9 4 6 8 2 5 9 4 3 8 9 1
output
32

Note

In the sample testcase, one optimum solution is to prepare the first problem on day 1 and print it on day 1, prepare the second problem on day 2 and print it on day 4, prepare the third problem on day 3 and print it on day 5, and prepare the fourth problem on day 6 and print it on day 8.

M

inc ARG

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Sergey is testing a next-generation processor. Instead of bytes the processor works with memory cells consisting of n bits. These bits are numbered from 1 to n. An integer is stored in the cell in the following way: the least significant bit is stored in the first bit of the cell, the next significant bit is stored in the second bit, and so on; the most significant bit is stored in the n-th bit.

Now Sergey wants to test the following instruction: "add 1 to the value of the cell". As a result of the instruction, the integer that is written in the cell must be increased by one; if some of the most significant bits of the resulting number do not fit into the cell, they must be discarded.

Sergey wrote certain values of the bits in the cell and is going to add one to its value. How many bits of the cell will change after the operation?

Input

The first line contains a single integer n ($1 \le n \le 100$) — the number of bits in the cell.

The second line contains a string consisting of n characters — the initial state of the cell. The first character denotes the state of the first bit of the cell. The second character denotes the second least significant bit and so on. The last character denotes the state of the most significant bit.

Output

Print a single integer — the number of bits in the cell which change their state after we add 1 to the cell.

Examples

input
4 1100
output
3

input
4 1111
output
4

Note

In the first sample the cell ends up with value 0010, in the second sample — with 0000.

DZY Loves Strings

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

DZY loves strings, and he enjoys collecting them.

In China, many people like to use strings containing their names' initials, for example: `xyz`, `jcvb`, `dzy`, `dyh`.

Once DZY found a lucky string s. A lot of pairs of good friends came to DZY when they heard about the news. The first member of the i-th pair has name a_i, the second one has name b_i. Each pair wondered if there is a substring of the lucky string containing both of their names. If so, they want to find the one with minimum length, which can give them good luck and make their friendship last forever.

Please help DZY for each pair find the minimum length of the substring of s that contains both a_i and b_i, or point out that such substring doesn't exist.

A substring of s is a string $s_l s_{l+1} \ldots s_r$ for some integers l, r $(1 \le l \le r \le |s|)$. The length of such the substring is $(r - l + 1)$.

A string p contains some another string q if there is a substring of p equal to q.

Input

The first line contains a string s $(1 \le |s| \le 50000)$.

The second line contains a non-negative integer q $(0 \le q \le 100000)$ — the number of pairs. Each of the next q lines describes a pair, the line contains two space-separated strings a_i and b_i $(1 \le |a_i|, |b_i| \le 4)$.

It is guaranteed that all the strings only consist of lowercase English letters.

Output

For each pair, print a line containing a single integer — the minimum length of the required substring. If there is no such substring, output `-1`.

Examples

input
xudyhduxyz 3 xyz xyz dyh xyz dzy xyz
output
3 8 -1

input
abcabd 3 a c ab abc ab d
output
2 3 3

input
baabcabaaa 2 abca baa aa aba

output

```
6
4
```

Note

The shortest substrings in the first sample are: `xyz`, `dyhduxyz`.

The shortest substrings in the second sample are: `ca`, `abc` and `abd`.

The shortest substrings in the third sample are: `baabca` and `abaa`.