TASK	KINO	ZIMA	KEKS	OGRADA	BROJ	KRIPTOGRAM

source code	`kino.pas`
	`kino.c`
	`kino.cpp`
		`zima.pas`
		`zima.c`
		`zima.cpp`
			`keks.pas`
			`keks.c`
			`keks.cpp`
				`ograda.pas`
				`ograda.c`
				`ograda.cpp`
					`broj.pas`
					`broj.c`
					`broj.cpp`
						`kriptogram.pas`
						`kriptogram.c`
						`kriptogram.cpp`

input	standard input (*stdin*)					
output	standard output (*stdout*)					
time limit	1 second	1 second	1 second	1 second	1 second	1 second
memory limit	32 MB	32 MB	32 MB	32 MB	64 MB	64 MB
point value	**50**	**100**	**100**	**120**	**140**	**140**
	650					

New theater just opened in Mirko's home town, and Mirko and Slavko naturally went to check it out. Opening projection was filled to the last place, and Mirko got mad because cup-holders at both sides of his seat were taken and he had nowhere to put his Coke.

A row in theater has **N** seats. There is a single cup-holder between adjacent seats, and also two additional cup-holders at both ends of the row. Exception to this are pairs of **love seats** - there is no cup-holder between them.

Your task is to help Mirko. Given sequence of letters describing seats in some row, and assuming that all seats are taken, find the **maximum number** of people that can put their cups in a cup-holder right next to their seat.

Letter 'S' in the sequence denotes ordinary seat, and 'L' denotes love seat. Love seats will always come in pairs of adjacent seats.

Diagram below corresponds to sequence 'SLLLLSSLL', with asterixes denoting cup-holders.

* S * L L * L L * S * S * L L *

For this example, at least two persons won't be able to put their cups into cup-holders.

INPUT

The first line of input contains the integer **N** ($1 \leq \mathbf{N} \leq 50$), number of seats in a row.

The following line contains a sequence of **N** letters 'L' or 'S', describing the row as stated above.

OUTPUT

The first and only line of output should contain the maximum number of people that can put their cups in cup-holder right next to them.

SAMPLE TESTS

input	**input**	**input**
3 SSS	4 SLLS	9 SLLLLSSLL
output	**output**	**output**
3	4	7

In the old times, when dragons ruled the land, air temperature never fell below zero degrees. But after dragons left, nice and warm times went with them.

We say that day is a winter day when average temperature for that day is below zero. We say that **T** consecutive winter days form a winter period of length **T**.

There are some people that annoy everyone by rambling on about some winter that is coming. So a law had to be made, stating that you are allowed to say that winter is coming **at most 2T days before a winter period of length T**. Exception to this is only the winter period with the longest length, that can be announced **at most 3T days before it begins**. During some winter period, you can't say that this period is coming, since it's already here, but you can announce future winter periods according to the rules above. If there are more than one winter periods with the longest length, then only one is chosen and 3T days rule is applied **only to that period**.

Knowing expected temperatures for some time period, find out the **maximum number of days during which it is allowed to say that winter is coming.**

INPUT

The first line of input contains the integer **N** ($1 \leq \mathbf{N} \leq 100\,000$), length of the overall time period we are considering. The following line contains **N** integers, temperatures of consecutive days in considered period. Absolute values of these integers won't exceed 100.

OUTPUT

The first and only line of output should contain the maximum number of days during which it is allowed to announce the winter is coming.

SCORING

In test cases **worth 40%** of total points, there will be only one winter period with the longest length.

SAMPLE TESTS

input	input
8 1 -1 4 3 8 -2 3 -3	15 1 2 -1 2 3 4 5 6 1 4 8 3 -1 -2 1
output	**output**
6	8

First sample description: There are three winter periods of length one. In order to obtain the requested maximum, best thing to do is to choose the second one to apply the 3T days rule, and starting announcing it three days before it arrives.

Day	1.	2.	3.	4.	5.	6.	7.	8.
Temperature	1	-1	4	3	8	-2	3	-3
Winter	no	yes	no	no	no	yes	no	yes
Winter is coming	yes	no	yes	yes	yes	yes	yes	no

Mirko and Slavko are bored at math class again so they came up with new game. Mirko writes down an **N** digit number, and Slavko's task is to obtain the **largest** possible number after having removed exactly **K** digits.

Help him do that!

INPUT

The first line of input contains integers **N** and **K** ($1 \leq \mathbf{K} < \mathbf{N} \leq 500\,000$).

The following line contains **N** digit number. This number starts with non-zero digit.

OUTPUT

The first and only line of output should contain the largest possible number Slavko can obtain by removing **K** digits from the given number.

SCORING

In test cases worth 50% of total points, **N** will not exceed 1000.

SAMPLE TESTS

input	**input**	**input**
4 2 1924	7 3 1231234	10 4 4177252841
output	**output**	**output**
94	3234	775841

In Mirko's village all fences are made of exactly **N** boards with **different heights**. Mirko doesn't have his own fence yet, so he decided to build one.

Each board is represented by a positive integer less than 10^9 - it's height. We define **the niceness** of the fence as a **sum of height differences between adjacent boards**.

Mirko already bought the boards, but he doesn't know how to order them into a fence. He would like his fence to be **similar** to Slavko's fence, but also to be as nice as possible.

We say that two fences are **similar** if ordering of adjacent boards is the same in both fences, i.e. if **i**-th board of one fence is smaller (or larger) than (**i+1**)-st, than the same must hold for that boards of the other fence.

Given Slavko's fence configuration, and heights of boards that Mirko bought, put Mirko's fence together so that it is similar to Slavko's but also as nice as possible. If there is more than one solution, output any of them.

INPUT

The first line of input contains integer **N** ($2 \leq \mathbf{N} \leq 300\,000$), number of boards in each fence.

The following line conatins **N** different positive integers representing Slavko's fence.

The following line contains **N** different positive integers representing heights of boards Mirko bought for his fence.

OUTPUT

The first line of output should contain the maximum possible niceness of Mirko's fence.

The following line should contain **N** integers, Mirko's boards in optimal order as described.

SCORING

50% of the points for test case will be awarded if **niceness** is correct, but second line is incorrect or not outputed at all.

SAMPLE TESTS

input

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

output

```
7
2 4 1 3
```

input

```
10
9 5 1 2 6 7 4 18 20 12
10 40 20 30 50 70 80 100 1000 500
```

output

```
3010
100 80 10 40 50 1000 20 70 500 30
```

First sample description: Mirko bought boards of heights 1, 2, 3 and 4. Fences similar to Slavko's that he can build are:

{1,3,2,4} - niceness 2+1+2=5
{1,4,2,3} - niceness 3+2+1=6
{2,3,1,4} - niceness 1+2+3=6
{2,4,1,3} - niceness 2+3+2=7
{3,4,1,2} - niceness 1+3+1=5

Find the **N**-th smallest positive integer whose least prime factor is **P**, or state that the result is greater than 10^9.

INPUT

The first and only line of input contains space seperated integers **N** and **P** ($1 \leq \mathbf{N}, \mathbf{P} \leq 10^9$). **P** will always be prime.

OUTPUT

Output a single line with the expected result, or zero if result exceeds 10^9.

SCORING

In test cases worth 30% of total points, expected result will either be less than 100 000, or will exceed 10^9.

In test cases worth additional 30% of total points, **P** will be greater than 1000.

SAMPLE TESTS

input	input	input
1 2	2 3	1000 1000003
output	output	output
2	9	0

Mirko intercepted an encrypted message. Mirko knows only that specific sentence was **part of the original message**. Find **the first occurence** of this sentence **inside the encrypted message**.

Message is encrypted by substituting **every word** from the original message with some (possibly the same one) word. If some word appears more than once in the original message, it will be substituted using the same word on each appearance. No two different words will have the same substitute word.

Words are space seperated sequences of lowercase letters. Sentence is a sequence of consecutive words.

INPUT

The first line of input contains the encrypted message. This message will not contain more than 10^6 characters. There will be exactly one whitespace between adjacent words, and end of the line will be marked with $. Trailing $ is not considered to be a part of message.

The following line contains the sentence that appears in the original message, and which we must find in the encrpyted message. This sentence won't be longer than 10^6 characters, and will follow the same format as described above.

OUTPUT

The first and only line of output should contain the index in the encrypted message of the first word in the first occurence of given original message sentence.

Solution will always exist.

SAMPLE TESTS

input	input	input
a a a b c d a b c $ x y $	xyz abc abc xyz $ abc abc $	a b c x c z z a b c $ prvi dr prvi tr tr x $
output	**output**	**output**
3	2	3