TASK	DOM	F7	KONCERT	LJUBOMORA	SNAGA	MARS

source code	`dom.pas`
	`dom.c`
	`dom.cpp`
		`f7.pas`
		`f7.c`
		`f7.cpp`
			`koncert.pas`
			`koncert.c`
			`koncert.cpp`
				`ljubomora.pas`
				`ljubomora.c`
				`ljubomora.cpp`
					`snaga.pas`
					`snaga.c`
					`snaga.cpp`
						`mars.pas`
						`mars.c`
						`mars.cpp`

input	standard input (*stdin*)					
output	standard output (*stdout*)					
time limit	1 second	1 second	3 seconds	1 second	1 second	1 second
memory limit	32 MB	32 MB	32 MB	32 MB	32 MB	32 MB
point value	**50**	**80**	**100**	**120**	**140**	**160**
	650					

Problems translated from Croatian by: **Ivan Pilat**

In a small country far, far away, all the best and brightest high schoolers apply to universities abroad (even further away). The exact cause is always different and hard to find. Of course, the country's leaders aren't happy about that since they care deeply about their gifted (read profitable) youth.

That's why the Education Ministry has started research into various subliminal messages to coax high schoolers into staying in their homeland. Their first try was handing out booster packs for the recent *Magic: The Gathering* expansion known as *Return to Ravnica*[1] as competition prizes. Then they started to play Tony Cetinski's song "Ostani zauvijek"[2] on math radio all the time. Finally, there is this task!

Given a word, remove from it all letters contained in the word CAMBRIDGE.

INPUT

The first and only line of input contains a single word consisting of at least 3 and at most 100 uppercase English letters.

OUTPUT

The first and only line of output should contain the word obtained by removing all the letters in CAMBRIDGE from the given word. Test data will ensure that the solution contains at least one letter.

SAMPLE TESTS

input LOVA **output** LOV	**input** KARIJERA **output** KJ

1 In Croatian, "ravnica" means "plain" (noun – level terrain; in MTG, white mana land type)
2 "Stay Forever"

The fictional World Championship of Formula 7 Drivers 2012 was characterized by exciting races and frequent shifts of driver positions on the leaderboard. Antun has missed most of it because he was training for olympiads in informatics. Now his only consolation are his medals and being the main character in this task. He has a simple question for you COCI contestants: „How many drivers participating in this Championship still had a chance to become Formula 7 World Champion **at the start of the final race**?" The World Champion is, of course, the driver with the **largest point total** at the end (after the final race).

There are **N** drivers participating in the Championship. They are all assigned points after each race, including the final one. The winner of the race is awarded **N points**, the runner-up gets **N - 1 points**, and so on until the last driver, who gets **1 point**. Two drivers cannot finish a race in the same spot.

Write a program to calculate, based on the total number of points that each driver has earned before the final race, how many drivers still have a chance to have the largest total after the final race and thus win the Championship. If more than one driver has the same maximum point total, they are all awarded the World Champion title.

INPUT

The first line of input contains the positive integer **N** ($3 \leq$ **N** $\leq 300\ 000$), the number of drivers participating in the Championship.

Each of the following **N** lines contains a single integer **Bi** ($0 \leq$ **Bi** $\leq 2\ 000\ 000$, **i** = 1, ..., **N**), the number of points that a driver has before the final race.

OUTPUT

The first and only line of output should contain the requested number of drivers that can still win.

SAMPLE TESTS

input	input
3 8 10 9	5 15 14 15 12 14
output	**output**
3	4

M guys and **N** girls are waiting in front of a concert venue. Some of them already have a ticket, while others are hoping they can still buy one. However, news are just in that one of the performers has had to cancel his appearance. Even worse, all tickets are already sold out! The girls **don't want** to stay at the concert anymore since their favourite performer won't appear; however, all the guys **do want** to stay anyways. **Tickets aren't tied to a particular person**, so the guys can ask **the girls who have tickets** to give the tickets to them.

Each guy and girl have either **zero or one tickets** in the beginning, but they can generally carry an **unlimited number of tickets**. Each person who has **at least one ticket** can give one of their tickets to **any person on the same side of the entrance (either in front of the entrance or inside the venue)**. Each person can enter the venue **only if they have a ticket**, which they keep upon entering. Each person in the venue can exit **with or without a ticket**, keeping any ticket upon exiting.

Determine a sequence of entering, exiting, and ticket giving actions, such that **all girls end up outside the venue** and a **maximum number of guys end up inside the venue**.

INPUT

The first line of input contains two positive integers, **M** ($1 \leq$ **M** $\leq 100\ 000$), the number of guys, and **A** ($1 \leq$ **A** $\leq$ **M**), the number of guys owning a ticket. Each guy is identified by a unique positive integer between **1** and **M**.

The second line of input contains the identifiers of guys with tickets, sorted in ascending order.

The third line of input contains two positive integers, **N** ($1 \leq$ **N** $\leq 100\ 000$), the number of girls, and **B** ($1 \leq$ **B** $\leq$ **N**), the number of girls owning a ticket. Each girl is identified by a unique positive integer between **1** and **N**.

The fourth line of input contains the identifiers of girls with tickets, sorted in ascending order.

OUTPUT

Output **any sequence** of actions satisfying the problem constraints, with length at most 1 000 000. All illegal actions **will be ignored**. Output each action in its own line. Let **X** and **Y** be numeric identifiers of guys and girls.

Output a guy entering the venue as **ENTER GUY X**, and a girl entering as **ENTER GIRL X**.

Output a guy exiting the venue as **EXIT GUY X**, and a girl exiting as **EXIT GIRL X**.

Output a person giving a ticket to a person as **GIVE GUY X GUY Y, GIVE GUY X GIRL Y, GIVE GIRL X GUY Y** or **GIVE GIRL X GIRL Y**.

SAMPLE TESTS

input	input
2 1 1 1 1 1	3 1 3 4 4 1 2 3 4
output	**output**
ENTER GUY 1 GIVE GIRL 1 GUY 2 ENTER GUY 2	GIVE GIRL 3 GUY 1 GIVE GIRL 2 GUY 1 GIVE GUY 1 GUY 2 ENTER GUY 2 ENTER GUY 1 ENTER GUY 3

A marble factory has donated a large box of marbles to a kindergarten. Each marble has one out of **M** different colours. The governess needs to divide **all** the marbles between the **N** children in her group. It is acceptable if some children don't get any marbles. However, no child wants marbles of different colours – in other words, all marbles that a child gets need to be the **same colour**.

The governess also knows that children will be jealous if a child gets too many marbles. As an approximation, we will define the **envy level** in the group as the **largest** number of marbles given to one child. Help the governess divide the marbles in order to **minimize** the envy level.

For example, if the box contains 4 red marbles (RRRR) and 7 blue marbles (BBBBBBB) which we have to divide between 5 children, we can achieve an envy level of 3 by dividing the marbles in the following way: RR, RR, BB, BB, BBB. This is the lowest achievable envy level.

INPUT

The first line of input contains two positive integers, **N** ($1 \le \mathbf{N} \le 10^9$), the number of children, and **M** ($1 \le \mathbf{M} \le 300\,000$, $\mathbf{M} \le \mathbf{N}$), the number of different colours.

Each of the following **M** lines contains a positive integer from the interval $[1, 10^9]$, with the integer in line **K** denoting the number of marbles with colour **K**.

OUTPUT

The first and only line of output should contain the minimum possible envy level.

SAMPLE TESTS

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

Let us begin with a positive integer **N** and find the **smallest** positive integer which **doesn't divide N**. If we repeat the procedure with the resulting number, then again with the new result and so on, we will eventually obtain the number 2 (two). Let us define **strength**(**N**) as the length of the resulting sequence.

For example, for **N** = 6 we obtain the sequence 6, 4, 3, 2 which consists of 4 numbers, thus strength(6) = 4.

Given two positive integers **A** < **B**, calculate the sum of strengths of all integers between **A** and **B** (inclusive), that is,

$$\text{strength}(\mathbf{A}) + \text{strength}(\mathbf{A} + 1) + \ldots + \text{strength}(\mathbf{B}).$$

INPUT

The first and only line of input contains two positive integers, **A** and **B** ($3 \le \mathbf{A} < \mathbf{B} < 10^{17}$).

OUTPUT

The first and only line of output should contain the requested sum of strengths.

SAMPLE TESTS

input 3 6 **output** 11	**input** 100 200 **output** 262

Scientists have discovered some strange bacteria on Mars and are now busy studying them. They have noticed that the number of bacteria is a power of 2, since each bacterium on Mars splits into two new bacteria (dying in the process), and it all started from a single bacterium.

Thus, in the first generation there was a single bacterium. It split into two bacteria of the second generation, which split into four bacteria of the third generation, and so on – until the $2^{\mathbf{K}}$ bacteria of generation **K**+1 that the scientists have discovered. They have numbered the bacteria using numbers from 1 to $2^{\mathbf{K}}$ in the following way:

- descendants of bacteria of the previous (**K**th) generation are, in this order: {1, 2}, {3, 4}, {5, 6}, ..., {$2^{\mathbf{K}}$ - 1, $2^{\mathbf{K}}$}
- descendants of bacteria of the older ((**K**-1)th) generation are, in this order: {1, 2, 3, 4}, {5, 6, 7, 8}, ..., {$2^{\mathbf{K}}$ - 3, $2^{\mathbf{K}}$ - 2, $2^{\mathbf{K}}$ - 1, $2^{\mathbf{K}}$}
- descendants of bacteria of the even older ((**K**-2)th) generation are, in this order: {1, 2, 3, 4, 5, 6, 7, 8}, ..., {$2^{\mathbf{K}}$ - 7, $2^{\mathbf{K}}$ - 6, $2^{\mathbf{K}}$ - 5, $2^{\mathbf{K}}$ - 4, $2^{\mathbf{K}}$ - 3, $2^{\mathbf{K}}$ - 2, $2^{\mathbf{K}}$ - 1, $2^{\mathbf{K}}$}
- ...
- descendants of the two bacteria of the second generation are, in this order: {1, 2, ..., $2^{\mathbf{K}-1}$} and {$2^{\mathbf{K}-1}$ + 1, $2^{\mathbf{K}-1}$ + 2, ..., $2^{\mathbf{K}}$}

where curly braces denote a set of descendants of a single bacteria. That is, the $2^{\mathbf{K}}$ bacteria of the current generation were numbered such that descendants of any older bacterium have consecutive numbers.

Notice that there exist many different permutations of these bacteria which still satisfy the rule that **descendants of any older bacterium have consecutive sequence numbers**. Scientists want to arrange the bacteria into such a sequence which also has the **minimum possible length**. The length of a bacteria sequence is the **sum of distances** between all **neighbouring** bacteria pairs.

Specifically, there is a certain quantifiable **repulsion** between every two bacteria, which is the minimum distance between them if they are next to each other in the sequence. (Repulsion plays no role between bacteria that are not immediate neighbours in the sequence.) Given the repulsion values for all bacteria pairs, find the minimum possible length of a bacteria sequence (permutation) satisfying the descendant rule given above.

INPUT

The first line of input contains the positive integer **K** ($1 \le \mathbf{K} \le 9$) from the problem statement.

Each of the following $2^{\mathbf{K}}$ lines contains $2^{\mathbf{K}}$ integers from the interval $[0, 10^6]$. These $2^{\mathbf{K}} \times 2^{\mathbf{K}}$ numbers represent repulsion between bacteria pairs: the number in row **m** and column **n** is the repulsion between bacteria **m** and **n**. This number will, of course, be equal to the number in row **n** and column **m**. For **m** = **n** the number will be 0.

OUTPUT

The first and only line of output should contain the minimum possible length of a bacteria sequence satisfying the constraints.

SAMPLE TESTS

input	input
2 0 7 2 1 7 0 4 3 2 4 0 5 1 3 5 0	3 0 2 6 3 4 7 1 3 2 0 7 10 9 1 3 6 6 7 0 3 5 6 5 5 3 10 3 0 9 8 9 7 4 9 5 9 0 9 8 4 7 1 6 8 9 0 8 7 1 3 5 9 8 8 0 10 3 6 5 7 4 7 10 0
output	**output**
13	32