Codeforces Round #496 (Div. 3)

A. Tanya and Stairways

1 second, 256 megabytes

Little girl Tanya climbs the stairs inside a multi-storey building. Every time Tanya climbs a stairway, she starts counting steps from 1 to the number of steps in this stairway. She speaks every number aloud. For example, if she climbs two stairways, the first of which contains 3 steps, and the second contains 4 steps, she will pronounce the numbers $1, 2, 3, 1, 2, 3, 4$.

You are given all the numbers pronounced by Tanya. How many stairways did she climb? Also, output the number of steps in each stairway.

The given sequence will be a valid sequence that Tanya could have pronounced when climbing one or more stairways.

Input

The first line contains n $(1 \le n \le 1000)$ — the total number of numbers pronounced by Tanya.

The second line contains integers $a_1, a_2, \dots, a_n$ $(1 \le a_i \le 1000)$ — all the numbers Tanya pronounced while climbing the stairs, in order from the first to the last pronounced number. Passing a stairway with x steps, she will pronounce the numbers $1, 2, \dots, x$ in that order.

The given sequence will be a valid sequence that Tanya could have pronounced when climbing one or more stairways.

Output

In the first line, output t — the number of stairways that Tanya climbed. In the second line, output t numbers — the number of steps in each stairway she climbed. Write the numbers in the correct order of passage of the stairways.

input
```
7
1 2 3 1 2 3 4
```
output
```
2
3 4
```

input
```
4
1 1 1 1
```
output
```
4
1 1 1 1
```

input
```
5
1 2 3 4 5
```
output
```
1
5
```

input
```
5
1 2 1 2 1
```
output
```
3
2 2 1
```

B. Delete from the Left

1 second, 256 megabytes

You are given two strings s and t. In a single move, you can choose any of two strings and delete the first (that is, the leftmost) character. After a move, the length of the string decreases by 1. You can't choose a string if it is empty.

For example:

- by applying a move to the string "`where`", the result is the string "`here`",
- by applying a move to the string "`a`", the result is an empty string "".

You are required to make two given strings equal using the fewest number of moves. It is possible that, in the end, both strings will be equal to the empty string, and so, are equal to each other. In this case, the answer is obviously the sum of the lengths of the initial strings.

Write a program that finds the minimum number of moves to make two given strings s and t equal.

Input

The first line of the input contains s. In the second line of the input contains t. Both strings consist only of lowercase Latin letters. The number of letters in each string is between 1 and $2 \cdot 10^5$, inclusive.

Output

Output the fewest number of moves required. It is possible that, in the end, both strings will be equal to the empty string, and so, are equal to each other. In this case, the answer is obviously the sum of the lengths of the given strings.

input
```
test
west
```
output
```
2
```

input
```
codeforces
yes
```
output
```
9
```

input
```
test
yes
```
output
```
7
```

input
```
b
ab
```
output
```
1
```

In the first example, you should apply the move once to the first string and apply the move once to the second string. As a result, both strings will be equal to "`est`".

In the second example, the move should be applied to the string "`codeforces`" 8 times. As a result, the string becomes "~~`codeforc`~~`es`" $\to$ "`es`". The move should be applied to the string "`yes`" once. The result is the same string "~~`y`~~`es`" $\to$ "`es`".

In the third example, you can make the strings equal only by completely deleting them. That is, in the end, both strings will be empty.

In the fourth example, the first character of the second string should be deleted.

C. Summarize to the Power of Two

3 seconds, 256 megabytes

A sequence $a_1, a_2, \dots, a_n$ is called good if, for each element a_i, there exists an element a_j ($i \ne j$) such that $a_i + a_j$ is a power of two (that is, 2^d for some non-negative integer d).

For example, the following sequences are good:

- $[5, 3, 11]$ (for example, for $a_1 = 5$ we can choose $a_2 = 3$. Note that their sum is a power of two. Similarly, such an element can be found for a_2 and a_3),
- $[1, 1, 1, 1023]$,
- $[7, 39, 89, 25, 89]$,
- $[]$.

Note that, by definition, an empty sequence (with a length of 0) is good.

For example, the following sequences are not good:

- $[16]$ (for $a_1 = 16$, it is impossible to find another element a_j such that their sum is a power of two),
- $[4, 16]$ (for $a_1 = 4$, it is impossible to find another element a_j such that their sum is a power of two),
- $[1, 3, 2, 8, 8, 8]$ (for $a_3 = 2$, it is impossible to find another element a_j such that their sum is a power of two).

You are given a sequence $a_1, a_2, \dots, a_n$. What is the minimum number of elements you need to remove to make it good? You can delete an arbitrary set of elements.

Input

The first line contains the integer n ($1 \le n \le 120000$) — the length of the given sequence.

The second line contains the sequence of integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$).

Output

Print the minimum number of elements needed to be removed from the given sequence in order to make it good. It is possible that you need to delete all n elements, make it empty, and thus get a good sequence.

input
```
6
4 7 1 5 4 9
```
output
```
1
```

input
```
5
1 2 3 4 5
```
output
```
2
```

input
```
1
16
```
output
```
1
```

input
```
4
1 1 1 1023
```
output
```
0
```

In the first example, it is enough to delete one element $a_4 = 5$. The remaining elements form the sequence $[4, 7, 1, 4, 9]$, which is good.

D. Polycarp and Div 3

3 seconds, 256 megabytes

Polycarp likes numbers that are divisible by 3.

He has a huge number s. Polycarp wants to cut from it the maximum number of numbers that are divisible by 3. To do this, he makes an arbitrary number of vertical cuts between pairs of adjacent digits. As a result, after m such cuts, there will be $m + 1$ parts in total. Polycarp analyzes each of the obtained numbers and finds the number of those that are divisible by 3.

For example, if the original number is $s = 3121$, then Polycarp can cut it into three parts with two cuts: $3|1|21$. As a result, he will get two numbers that are divisible by 3.

Polycarp can make an arbitrary number of vertical cuts, where each cut is made between a pair of adjacent digits. The resulting numbers cannot contain extra leading zeroes (that is, the number can begin with `0` if and only if this number is exactly one character '`0`'). For example, `007`, `01` and `00099` are not valid numbers, but `90`, `0` and `10001` are valid.

What is the maximum number of numbers divisible by 3 that Polycarp can obtain?

Input

The first line of the input contains a positive integer s. The number of digits of the number s is between 1 and $2 \cdot 10^5$, inclusive. The first (leftmost) digit is not equal to `0`.

Output

Print the maximum number of numbers divisible by 3 that Polycarp can get by making vertical cuts in the given number s.

input
```
3121
```
output
```
2
```

input
```
6
```
output
```
1
```

input
```
1000000000000000000000000000000000
```
output
```
33
```

input
```
201920181
```
output
```
4
```

In the first example, an example set of optimal cuts on the number is `3|1|21`.

In the second example, you do not need to make any cuts. The specified number `6` forms one number that is divisible by 3.

In the third example, cuts must be made between each pair of digits. As a result, Polycarp gets one digit `1` and 33 digits `0`. Each of the 33 digits `0` forms a number that is divisible by 3.

In the fourth example, an example set of optimal cuts is `2|0|1|9|201|81`. The numbers 0, 9, 201 and 81 are divisible by 3.

E1. Median on Segments (Permutations Edition)

3 seconds, 256 megabytes

You are given a permutation $p_1, p_2, \dots, p_n$. A permutation of length n is a sequence such that each integer between 1 and n occurs exactly once in the sequence.

Find the number of pairs of indices (l, r) $(1 \le l \le r \le n)$ such that the value of the median of $p_l, p_{l+1}, \dots, p_r$ is exactly the given number m.

The median of a sequence is the value of the element which is in the middle of the sequence after sorting it in non-decreasing order. If the length of the sequence is even, the left of two middle elements is used.

For example, if $a = [4, 2, 7, 5]$ then its median is 4 since after sorting the sequence, it will look like $[2, 4, 5, 7]$ and the left of two middle elements is equal to 4. The median of $[7, 1, 2, 9, 6]$ equals 6 since after sorting, the value 6 will be in the middle of the sequence.

Write a program to find the number of pairs of indices (l, r) ($1 \le l \le r \le n$) such that the value of the median of $p_l, p_{l+1}, \dots, p_r$ is exactly the given number m.

Input

The first line contains integers n and m $(1 \le n \le 2 \cdot 10^5, 1 \le m \le n)$ — the length of the given sequence and the required value of the median.

The second line contains a permutation $p_1, p_2, \dots, p_n$ $(1 \le p_i \le n)$. Each integer between 1 and n occurs in p exactly once.

Output

Print the required number.

input
```
5 4
2 4 5 3 1
```
output
```
4
```

input
```
5 5
1 2 3 4 5
```
output
```
1
```

input
```
15 8
1 15 2 14 3 13 4 8 12 5 11 6 10 7 9
```
output
```
48
```

In the first example, the suitable pairs of indices are: $(1, 3)$, $(2, 2)$, $(2, 3)$ and $(2, 4)$.

E2. Median on Segments (General Case Edition)

3 seconds, 256 megabytes

You are given an integer sequence $a_1, a_2, \dots, a_n$.

Find the number of pairs of indices (l, r) $(1 \le l \le r \le n)$ such that the value of median of $a_l, a_{l+1}, \dots, a_r$ is exactly the given number m.

The median of a sequence is the value of an element which is in the middle of the sequence after sorting it in non-decreasing order. If the length of the sequence is even, the left of two middle elements is used.

For example, if $a = [4, 2, 7, 5]$ then its median is 4 since after sorting the sequence, it will look like $[2, 4, 5, 7]$ and the left of two middle elements is equal to 4. The median of $[7, 1, 2, 9, 6]$ equals 6 since after sorting, the value 6 will be in the middle of the sequence.

Write a program to find the number of pairs of indices (l, r) ($1 \le l \le r \le n$) such that the value of median of $a_l, a_{l+1}, \dots, a_r$ is exactly the given number m.

Input

The first line contains integers n and m $(1 \le n, m \le 2 \cdot 10^5)$ — the length of the given sequence and the required value of the median.

The second line contains an integer sequence $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$).

Output

Print the required number.

input
```
5 4
1 4 5 60 4
```
output
```
8
```

input
```
3 1
1 1 1
```
output
```
6
```

input
```
15 2
1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```
output
```
97
```

In the first example, the suitable pairs of indices are: $(1, 3)$, $(1, 4)$, $(1, 5)$, $(2, 2)$, $(2, 3)$, $(2, 5)$, $(4, 5)$ and $(5, 5)$.

F. Berland and the Shortest Paths

5 seconds, 256 megabytes

There are n cities in Berland. Some pairs of cities are connected by roads. All roads are bidirectional. Each road connects two different cities. There is at most one road between a pair of cities. The cities are numbered from 1 to n.

It is known that, from the capital (the city with the number 1), you can reach any other city by moving along the roads.

The President of Berland plans to improve the country's road network. The budget is enough to repair exactly $n - 1$ roads. The President plans to choose a set of $n - 1$ roads such that:

- it is possible to travel from the capital to any other city along the $n - 1$ chosen roads,
- if d_i is the number of roads needed to travel from the capital to city i, moving only along the $n - 1$ chosen roads, then $d_1 + d_2 + \dots + d_n$ is minimized (i.e. as minimal as possible).

In other words, the set of $n - 1$ roads should preserve the connectivity of the country, and the sum of distances from city 1 to all cities should be minimized (where you can only use the $n - 1$ chosen roads).

The president instructed the ministry to prepare k possible options to choose $n - 1$ roads so that both conditions above are met.

Write a program that will find k possible ways to choose roads for repair. If there are fewer than k ways, then the program should output all possible valid ways to choose roads.

Input

The first line of the input contains integers n, m and k ($2 \le n \le 2 \cdot 10^5, n - 1 \le m \le 2 \cdot 10^5, 1 \le k \le 2 \cdot 10^5$), where n is the number of cities in the country, m is the number of roads and k is the number of options to choose a set of roads for repair. It is guaranteed that $m \cdot k \le 10^6$.

The following m lines describe the roads, one road per line. Each line contains two integers a_i, b_i $(1 \le a_i, b_i \le n, a_i \ne b_i)$ — the numbers of the cities that the i-th road connects. There is at most one road between a pair of cities. The given set of roads is such that you can reach any city from the capital.

Output

Print t $(1 \le t \le k)$ — the number of ways to choose a set of roads for repair. Recall that you need to find k different options; if there are fewer than k of them, then you need to find all possible different valid options.

In the following t lines, print the options, one per line. Print an option as a string of m characters where the j-th character is equal to '1' if the j-th road is included in the option, and is equal to '0' if the road is not included. The roads should be numbered according to their order in the input. The options can be printed in any order. All the t lines should be different.

Since it is guaranteed that $m \cdot k \le 10^6$, the total length of all the t lines will not exceed 10^6.

If there are several answers, output any of them.

input

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

output

```
2
1110
1011
```

input

```
4 6 3
1 2
2 3
1 4
4 3
2 4
1 3
```

output

```
1
101001
```

input

```
5 6 2
1 2
1 3
2 4
2 5
3 4
3 5
```

output

```
2
111100
110110
```