Problem A. Merges and Acquisitions

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

The CEO of the company named S is planning to merge with another company named T. The CEO wants to keep the original company name S through the merge process on the plea that both company names are mixed into a new one.

The CEO insists that the mixed company name is produced as follows.

Let s be an arbitrary subsequence of S, and t be an arbitrary subsequence of T. The new company name must be a string of the same length to S obtained by alternatively lining up the characters in s and t. More formally, $s_0 + t_0 + s_1 + t_1 + \ldots$ or $t_0 + s_0 + t_1 + s_1 + \ldots$ can be used as the company name after merging. Here,s_k denotes the k-th (0-based) character of string s. Please note that the lengths of s and t will be different if the length of S is odd. In this case, the company name after merging is obtained by $s_0 + t_0 + \ldots + t_{|S|/2} + s_{|S|/2+1}$ or $t_0 + s_0 + \ldots + s_{|S|/2} + t_{|S|/2+1}$ ($|S|$ denotes the length of S and "/" denotes integer division).

A subsequence of a string is a string which is obtained by erasing zero or more characters from the original string. For example, the strings "`abe`", "`abcde`" and "" (the empty string) are all subsequences of the string "`abcde`".

You are a programmer employed by the acquiring company. You are assigned a task to write a program that determines whether it is possible to make S, which is the original company name, by mixing the two company names.

Input

The first line contains a string S which denotes the name of the company that you belong to. The second line contains a string T which denotes the name of the target company of the planned merging. The two names S and T are non-empty and of the same length no longer than 10^3 characters, and all the characters used in the names are lowercase English letters.

Output

Print "`Yes`" in a line if it is possible to make original company name by combining S and T. Otherwise, print "`No`" in a line.

Examples

standard input	standard output
`acmicpc` `tsukuba`	`No`
`hoge` `moen`	`Yes`
`abcdefg` `xacxegx`	`Yes`

Problem B. Change a Password

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Password authentication is used in a lot of facilities. The office of JAG also uses password authentication. A password is required to enter their office. A password is a string of N digits '0'-'9'. This password is changed on a regular basis. Taro, a staff of the security division of JAG, decided to use the following rules to generate a new password from an old one.

1. The new password consists of the same number N of digits to the original one and each digit appears at most once in the new password. It can have a leading zero. (Note that an old password may contain same digits twice or more.)
2. The new password maximizes the difference from the old password within constraints described above. (Definition of the difference between two passwords is described below.)
3. If there are two or more candidates, the one which has the minimum value when it is read as an integer will be selected.

The difference between two passwords is defined by $min(|a - b|, 10^N - |a - b|)$, where a and b are the integers represented by the two passwords. For example, the difference between "11" and "42" is 31, and the difference between "987" and "012" is 25.

Taro would like to use a computer to calculate a new password correctly, but he is not good at programming. Therefore, he asked you to write a program. Your task is to write a program that generates a new password from an old password.

Input

The input consists of a single test case. The first line of the input contains a string S which denotes the old password. You can assume that the length of S is no less than 1 and no greater than 10. Note that old password S may contain same digits twice or more, and may have leading zeros.

Output

Print the new password in a line.

Examples

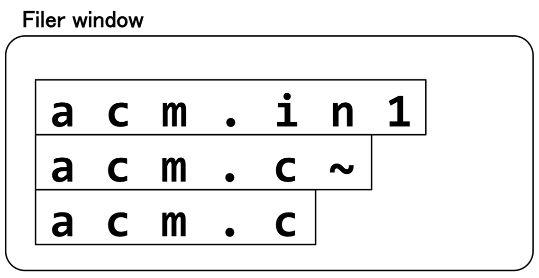
standard input	standard output
201	701
512	012
99999	49876
765876346	265874931

Problem C. Delete Files

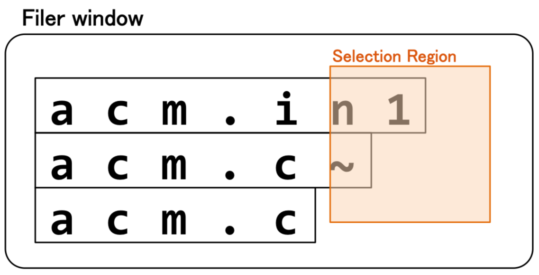
Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

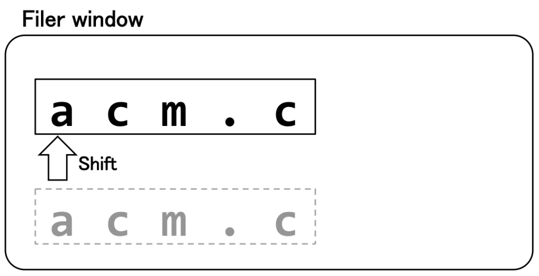
You are using an operating system named "Jaguntu". Jaguntu provides "Filer", a file manager with a graphical user interface.

When you open a folder with Filer, the name list of files in the folder is displayed on a Filer window. Each filename is displayed within a rectangular region, and this region is called a filename region. Each filename region is aligned to the left side of the Filer window. The height of each filename region is 1, and the width of each filename region is the filename length. For example, when three files "`acm.in1`", "`acm.c ̃`", and "`acm.c`" are stored in this order in a folder, it looks like this on the Filer window.



You can delete files by taking the following steps. First, you select a rectangular region with dragging a mouse. This region is called selection region. Next, you press the delete key on your keyboard. A file is deleted if and only if its filename region intersects with the selection region. After the deletion, Filer shifts each filename region to the upside on the Filer window not to leave any top margin on any remaining filename region. For example, if you select a region from the first picture below, then the two files "`acm.in1`" and "`acm.c ̃`" are deleted, and the remaining file "`acm.c`" is displayed on the top of the Filer window as at the second picture.





You are opening a folder storing N files with Filer. Since you have almost run out of disk space, you want to delete unnecessary files in the folder. Your task is to write a program that calculates the minimum number of times to perform deletion operation described above.

Input

The input consists of a single test case.

The first line contains one integer N ($1 \leq N \leq 1,000$), which is the number of files in a folder. Each of the next N lines contains a character D_i and an integer L_i: D_i indicates whether the i-th file should be deleted or not, and L_i ($1 \leq L_i \leq 1,000$) is the filename length of the i-th file. If D_i is '`y`', the i-th file should be deleted. Otherwise, D_i is always '`n`', and you should not delete the i-th file.

Output

Output the minimum number of deletion operations to delete only all the unnecessary files.

Examples

standard input	standard output

`3`
`y 7`
`y 6`
`n 5`
	`1`

`3`
`y 7`
`n 6`
`y 5`
	`2`

`6`
`y 4`
`n 5`
`y 4`
`y 6`
`n 3`
`y 6`
	`2`

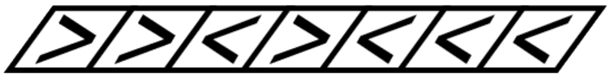
Problem D. Linear Gimmick

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

You are in front of a linear gimmick of a game. It consists of N panels in a row, and each of them displays a right or a left arrow.

You can step in this gimmick from any panel. Once you get on a panel, you are forced to move following the direction of the arrow shown on the panel and the panel will be removed immediately. You keep moving in the same direction until you get on another panel, and if you reach a panel, you turn in (or keep) the direction of the arrow on the panel. The panels you passed are also removed. You repeat this procedure, and when there is no panel in your current direction, you get out of the gimmick.

For example, when the gimmick is the following image



and you first get on the 2nd panel from the left, your moves are as follows.

- Move right and remove the 2nd panel.
- Move left and remove the 3rd panel.
- Move right and remove the 1st panel.
- Move right and remove the 4th panel.
- Move left and remove the 5th panel.
- Get out of the gimmick.

You are given a gimmick with N panels. Compute the maximum number of removed panels after you get out of the gimmick.

Input

The input consists of two lines. The first line contains an integer N ($1 \leq N \leq 10^5$) which represents the number of the panels in the gimmick. The second line contains a character string S of length N, which consists of '>' or '<'. The i-th character of S corresponds to the direction of the arrow on the i-th panel from the left. '<' and '>' denote left and right directions respectively.

Output

Output the maximum number of removed panels after you get out of the gimmick.

Example

standard input	standard output
7 >><><<<	7
5 >><<<	5
6 ><<><<	6
7 <<><<>>	5

Problem E. Shifting a Matrix

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

You are given $N \times N$ matrix A initialized with $A_{i,j} = (i-1) \cdot N + j$, where $A_{i,j}$ is the entry of the i-th row and the j-th column of A. Note that i and j are 1-based.

You are also given an operation sequence which consists of the four types of shift operations: left, right, up, and down shifts. More precisely, these operations are defined as follows:

- Left shift with i: circular shift of the i-th row to the left, i.e., setting previous $A_{i,k}$ to new $A_{i,k-1}$ for $2 \le k \le N$, and previous $A_{i,1}$ to new $A_{i,N}$.
- Right shift with i: circular shift of the i-th row to the right, i.e., setting previous $A_{i,k}$ to new $A_{i,k+1}$ for $1 \le k \le N-1$, and previous $A_{i,N}$ to new $A_{i,1}$.
- Up shift with j: circular shift of the j-th column to the above, i.e., setting previous $A_{k,j}$ to new $A_{k-1,j}$ for $2 \le k \le N$, and previous $A_{1,j}$ to new $A_{N,j}$.
- Down shift with j: circular shift of the j-th column to the below, i.e., setting previous $A_{k,j}$ to new $A_{k+1,j}$ for $1 \le k \le N-1$, and previous $A_{N,j}$ to new $A_{1,j}$.

An operation sequence is given as a string. You have to apply operations to a given matrix from left to right in a given string. Left, right, up, and down shifts are referred as 'L', 'R', 'U', and 'D' respectively in a string, and the following number indicates the row/column to be shifted. For example, "R25" means we should perform right shift with 25. In addition, the notion supports repetition of operation sequences. An operation sequence surrounded by a pair of parentheses must be repeated exactly m times, where m is the number following the close parenthesis. For example, "(L1R2)10" means we should repeat exactly 10 times the set of the two operations: left shift with 1 and right shift with 2 in this order.

Given operation sequences are guaranteed to follow the following BNF:

```
<sequence> := <sequence><rep> | <sequence><op> | <rep> | <op>
<rep> := '('<sequence>')'<number>
<op> := <shift><number>
<shift> := 'L' | 'R' | 'U' | 'D'
<number> := <nonzero_digit> |<number><digit>
<digit> := '0' | <nonzero_digit>
<nonzero_digit> := '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
```

Given N and an operation sequence as a string, make a program to compute the $N \times N$ matrix after operations indicated by the operation sequence.

Input

The first line of the input contains two integers N and L, where N $(1 \le N \le 100)$ is the size of the given matrix and L $(2 \le L \le 1,000)$ is the length of the following string. The second line contains a string S representing the given operation sequence. You can assume that S follows the above BNF. You can also assume numbers representing rows and columns are no less than 1 and no more than N, and the number of each repetition is no less than 1 and no more than 10^9 in the given string.

Output

Output the matrix after the operations in N lines, where the i-th line contains single-space separated N integers representing the i-th row of A after the operations.

Examples

standard input	standard output

`3 2`
`R1`
	`3 1 2`
	`4 5 6`
	`7 8 9`

`3 7`
`(U2)300`
	`1 2 3`
	`4 5 6`
	`7 8 9`

`3 7`
`(R1D1)3`
	`3 4 7`
	`1 5 6`
	`2 8 9`

Problem F. Modern Announce Network

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Today, modern teenagers use social networks to communicate with each other.

In a high school, N students are using an social network called ICPC (International Community for Programming Contest). Some pairs of these N students are "friends" on this system, and can send messages to each other. Among these N students, A first grade students, B second grade students, and C third grade students are members of the Programming Society. Note that there may be some students who are not the members of the Programming Society, so $A + B + C$ can be less than N.

There are good relationships between members of the same grade in the Society. Thus, there is a chat group in the social network for each grade, and the Society members of the same grade can communicate with each other instantly via their group chat. On the other hand, the relationships between any different grades are not so good, and there are no chat group for the entire Society and the entire high school.

In order to broadcast a message to all the Society members on the social network, the administrator of the Society came up with a method: the administrator tells the message to one of the N students and have them spread the message on the social network via the chat groups and their friends. (The administrator itself does not have an account for the social network.) As the members of the same grade can broadcast the message by the chat group for the grade, we can assume that if one of a grade gets the message, all other members of that grade also get the message instantly. Therefore, if the message is told to at least one member of each grade, we can assume that the message is broadcasted to the all members of the Society on the social network.

Because it is bothering to communicate between friends, we want to minimize the number of communications between friends. What is the minimum number of communications between friends to broadcast a message to all the Society members? Who is the first person to whom the administrator should tell the message to achieve the minimum communications?

Input

The first line of the input contains four integers N, A, B, and C. N ($3 \le N \le 10^4$) denotes the number of students in the social network, and A, B and C ($1 \le A, B, C \le N$ and $A+B+C \le N$) denote the number of members of the first, second, and third grade respectively.

Each student on the social network is represented by a unique numeric ID between 1 and N. The second line contains A integers $a_1, \ldots, a_A$ ($1 \le a_1, \ldots, a_A \le N$), which are the IDs of members of the first grade. The third line contains B integers $b_1, \ldots, b_B$ ($1 \le b_1, \ldots, b_B \le N$), which are the IDs of members of the second grade. The fourth line contains C integers $c_1, \ldots, c_C$ ($1 \le c_1, \ldots, c_C \le N$), which are the IDs of members of the third grade. You can assume that $a_1, \ldots, a_A, b_1, \ldots, b_B, c_1, \ldots, c_C$ are distinct.

The fifth line contains an integer M ($2 \le M \le 5 \cdot 10^5$). M denotes the number of pairs of students who are friends in the social network. The i-th line of the following M lines contains two integers x_i and y_i ($1 \le x_i, y_i \le N$), which means the student with ID x_i and the student with ID y_i are friends in the social network. You can assume that $x_i \ne y_i$ ($1 \le i \le M$), and ($x_i \ne x_j$ or $y_i \ne y_j$)

and ($x_i \neq y_j$ or $y_i \neq x_j$) if $i \neq j$ ($1 \leq i, j \leq M$). You can also assume that a message can be delivered from any student to any student in the social network via the friends and group chat.

Output

Output the minimum number of communications between friends (not including group chat) to broadcast a message among all the Society members, and the ID of the student to whom the administrator should tell the message in order to achieve the minimum number of communications, separated by a single space. If there are multiple students that satisfy the above constraints, output the minimum ID of such students.

Examples

standard input	standard output
`4 2 1 1`	`2 1`
`1 2`	
`3`	
`4`	
`3`	
`1 2`	
`2 4`	
`3 4`	
`4 1 1 1`	`3 1`
`2`	
`3`	
`4`	
`4`	
`1 2`	
`1 3`	
`1 4`	
`2 4`	
`8 1 2 2`	`2 3`
`5`	
`4 6`	
`3 8`	
`7`	
`1 2`	
`2 3`	
`3 4`	
`5 6`	
`6 7`	
`7 8`	
`2 6`	

Problem G. Cube Dividing

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Pablo Cubarson is a well-known cubism artist. He is producing a new piece of work using a cuboid which consists of $A \times B \times C$ unit cubes. He plans to make a beautiful shape by removing N units cubes from the cuboid. When he is about to begin the work, he has noticed that by the removal the cuboid may be divided into several parts disconnected to each other. It is against his aesthetics to divide a cuboid. So he wants to know how many parts are created in his plan.

Your task is to calculate the number of connected components in the cuboid after removing the N cubes. Two cubes are connected if they share one face.

Input

The first line of the input contains four integers A, B, C and N. A, B and C ($1 \le A, B, C \le 10^6$) denote the size of the cuboid — the cuboid has an A unit width from left to right and a B unit height from bottom to top, and a C unit depth from front to back. N ($0 \le N \le 2 \cdot 10^4$, $N \le A \times B \times C - 1$) denotes the number of the cubes removed in the Cubarson's plan. Each of the following N lines contains three integers X_i ($0 \le X_i \le A - 1$), Y_i ($0 \le Y_i \le B - 1$) and Z_i ($0 \le Z_i \le C - 1$). They denote that the cube located at the X_i-th position from the left, the Y_i-th from the bottom and the Z_i-th from the front will be removed in the plan. You may assume the given positions are distinct.

Output

Print the number of the connected components in the cuboid after removing the specified cubes.

Examples

standard input	standard output

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

`3 3 3 1`
`1 1 1`
	`1`

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

Problem H. Donut Decoration

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Donut maker's morning is early. Mr. D, who is also called Mr. Donuts, is an awesome donut maker. Also today, he goes to his kitchen and prepares to make donuts before sunrise.

In a twinkling, Mr. D finishes frying N donuts with a practiced hand. But these donuts as they are must not be displayed in a showcase. Filling cream, dipping in chocolate, topping somehow cute, colorful things, etc., several decoration tasks are needed. There are K tasks numbered 1 through K, and each of them must be done exactly once in the order $1, 2, \ldots, K$ to finish the donuts as items on sale.

Instantly, Mr. D arranges the N donuts in a row. He seems to intend to accomplish each decoration tasks sequentially at once. However, what in the world is he doing? Mr. D, who stayed up late at yesterday night, decorates only a part of the donuts in a consecutive interval for each task. It's clearly a mistake! Not only that, he does some tasks zero or several times, and the order of tasks is also disordered. The donuts which are not decorated by correct process cannot be provided as items on sale, so he should trash them.

Fortunately, there are data recording a sequence of tasks he did. The data contain the following information: for each task, the consecutive interval $[l, r]$ of the decorated donuts and the ID x of the task. Please write a program enumerating the number of the donuts which can be displayed in a showcase as items on sale for given recorded data.

Input

The first line of the input contains two integers N and K, where N ($1 \leq N \leq 2 \cdot 10^5$) is the number of the donuts fried by Mr. D, and K ($1 \leq K \leq 2 \cdot 10^5$) is the number of decoration tasks should be applied to the donuts. The second line contains a single integer T ($1 \leq T \leq 2 \cdot 10^5$), which means the number of information about tasks Mr. D did. Each of next T lines contains three integers l_i, r_i, and x_i representing the i-th task Mr. D did: the i-th task was applied to the interval $[l_i, r_i]$ ($1 \leq l_i \leq r_i \leq N$) of the donuts inclusive, and has ID x_i ($1 \leq x_i \leq K$).

Output

Output the number of the donuts that can be provided as items on sale.

Examples

standard input	standard output
3 2	1
3	
1 2 1	
2 3 2	
3 3 1	
5 3	2
6	
2 3 1	
1 3 2	
4 5 1	
2 4 3	
3 5 2	
5 5 3	
10 1	5
2	
2 9 1	
5 7 1	

Problem I. Shortest Bridge

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

There is a city whose shape is a $1,000 \times 1,000$ square. The city has a big river, which flows from the north to the south and separates the city into just two parts: the west and the east.

Recently, the city mayor has decided to build a highway from a point s on the west part to a point t on the east part. A highway consists of a bridge on the river, and two roads: one of the roads connects s and the west end of the bridge, and the other one connects t and the east end of the bridge. Note that each road doesn't have to be a straight line, but the intersection length with the river must be zero.

In order to cut building costs, the mayor intends to build a highway satisfying the following conditions:

- Since bridge will cost more than roads, at first the length of a bridge connecting the east part and the west part must be as short as possible.
- Under the above condition, the sum of the length of two roads is minimum.

Your task is to write a program computing the total length of a highway satisfying the above conditions.

Input

At first, we refer to a point on the city by a coordinate (x, y): the distance from the west side is x and the distance from the north side is y.

The first line of the input contains four integers s_x, s_y, t_x, and t_y ($0 \leq s_x, s_y, t_x, t_y \leq 1,000$): points s and t are located at (s_x, s_y) and (t_x, t_y) respectively.

The next line contains an integer N ($2 \leq N \leq 20$), where N is the number of points composing the west riverside. Each of the following N lines contains two integers wx_i and wy_i ($0 \leq wx_i, wy_i \leq 1,000$): the coordinate of the i-th point of the west riverside is (wx_i, wy_i). The west riverside is a polygonal line obtained by connecting the segments between (wx_i, wy_i) and $(wx_i + 1, wy_i + 1)$ for all $1 \leq i \leq N - 1$.

The next line contains an integer M ($2 \leq M \leq 20$), where M is the number of points composing the east riverside. Each of the following M lines contains two integers ex_i and ey_i ($0 \leq ex_i, ey_i \leq 1,000$): the coordinate of the i-th point of the east riverside is (ex_i, ey_i). The east riverside is a polygonal line obtained by connecting the segments between (ex_i, ey_i) and $(ex_i + 1, ey_i + 1)$ for all $1 \leq i \leq M - 1$.

You can assume that test cases are under the following conditions.

- wy_1 and ey_1 must be 0, and wy_N and ey_M must be 1,000.
- Each polygonal line has no self-intersection.
- Two polygonal lines representing the west and the east riverside have no cross point.

- A point s must be on the west part of the city. More precisely, s must be on the region surrounded by the square side of the city and the polygonal line of the west riverside and not containing the east riverside points.
- A point t must be on the east part of the city. More precisely, t must be on the region surrounded by the square side of the city and the polygonal line of the east riverside and not containing the west riverside points.
- Each polygonal line intersects with the square only at the two end points. In other words, $0 < wx_i, wy_i < 1,000$ holds for $2 \le i \le N - 1$ and $0 < ex_i, ey_i < 1,000$ holds for $2 \le i \le M - 1$.

Output

Output single-space separated two numbers in a line: the length of a bridge and the total length of a highway (i.e. a bridge and two roads) satisfying the above mayor's demand. The output can contain an absolute or a relative error no more than 10^{-8}.

Examples

standard input	standard output

200 500 800 500
3
400 0
450 500
400 1000
3
600 0
550 500
600 1000
	100 600

300 300 700 100
5
300 0
400 100
300 200
400 300
400 1000
4
700 0
600 100
700 200
700 1000	

Problem J. Longest Shortest Path

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

You are given a directed graph and two nodes s and t. The given graph may contain multiple edges between the same node pair but not self loops. Each edge e has its initial length d_e and the cost c_e. You can extend an edge by paying a cost. Formally, it costs $x \cdot c_e$ to change the length of an edge e from d_e to $d_e + x$. (Note that x can be a non-integer). Edges cannot be shortened.

Your task is to maximize the length of the shortest path from node s to node t by lengthening some edges within cost P. You can assume that there is at least one path from s to t.

Input

The first line of the input contains five integers N, M, P, s, and t: N ($2 \le N \le 200$) and M ($1 \le M \le 2,000$) are the number of the nodes and the edges of the given graph respectively, P ($0 \le P \le 10^6$) is the cost limit that you can pay, and s and t ($1 \le s, t \le N$, $s \ne t$) are the start and the end node of objective path respectively. Each of the following M lines contains four integers v_i, u_i, d_i and c_i, which mean there is an edge from v_i to u_i ($1 \le v_i, u_i \le N$, $v_i \ne u_i$) with the initial length d_i ($1 \le d_i \le 10$) and the cost c_i ($1 \le c_i \le 10$).

Output

Output the maximum length of the shortest path from node s to node t by lengthening some edges within cost P. The output can contain an absolute or a relative error no more than 10^{-6}.

Examples

standard input	standard output

3 2 3 1 3
1 2 2 1
2 3 1 2
	6

3 3 2 1 3
1 2 1 1
2 3 1 1
1 3 1 1
	2.5000000

3 4 5 1 3
1 2 1 2
2 3 1 1
1 3 3 2
1 3 4 1
	4.25

Problem K. Optimal Tournament

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

In 21XX, an annual programming contest "Japan Algorithmist GrandPrix (JAG)" has been one of the most popular mind sport events. You, the chairperson of JAG, are preparing this year's JAG competition.

JAG is conducted as a knockout tournament. This year, N contestants will compete in JAG. A tournament is represented as a binary tree having N leaf nodes, to which the contestants are allocated one-to-one. In each match, two contestants compete. The winner proceeds to the next round, and the loser is eliminated from the tournament. The only contestant surviving over the other contestants is the champion. The final match corresponds to the root node of the binary tree.

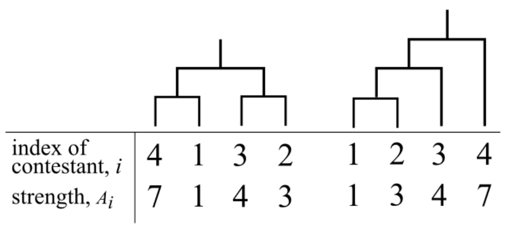
You know the strength of each contestant, $A_1, A_2, \ldots, A_N$, which is represented as an integer. When two contestants compete, the one having greater strength always wins. If their strengths are the same, the winner is determined randomly.

In the past JAG, some audience complained that there were too many boring one-sided games. To make JAG more attractive, you want to find a good tournament configuration.

Let's define the boringness of a match and a tournament. For a match in which the i-th contestant and the j-th contestant compete, we define the boringness of the match as the difference of the strength of the two contestants, $|A_i - A_j|$. And the boringness of a tournament is defined as the sum of the boringness of all matches in the tournament.

Your purpose is to minimize the boringness of the tournament.

You may consider any shape of the tournament, including unbalanced ones, under the constraint that the height of the tournament must be less than or equal to K. The height of the tournament is defined as the maximum number of the matches on the simple path from the root node to any of the leaf nodes of the binary tree.



This figure shows two possible tournaments for Sample Input 1. The height of the left one is 2, and the height of the right one is 3.

Write a program that calculates the minimum boringness of the tournament.

Input

The first line of the input contains two integers N ($2 \leq N \leq 1,000$) and K ($1 \leq K \leq 50$). You can assume that $N \leq 2K$. The second line contains N integers $A_1, A_2, \ldots A_N$. A_i ($1 \leq A_i \leq 10^5$) represents the strength of the i-th contestant.

Output

Output the minimum boringness value achieved by the optimal tournament configuration.

Examples

standard input	standard output
4 3 1 3 4 7	6
5 3 1 3 4 7 9	10

18 7
67 64 52 18 39 92 84 66 19 64 1 66
35 34 45 2 79 34
	114