Problem A. Burning Bridges

Input file:	`bridges.in`
Output file:	`bridges.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Ferry Kingdom is a nice little country located on N islands that are connected by M bridges. All bridges are very beautiful and are loved by everyone in the kingdom. Of course, the system of bridges is designed in such a way that one can get from any island to any other one.

But recently the great sorrow has come to the kingdom. Ferry Kingdom was conquered by the armies of the great warrior Jordan and he has decided to burn all the bridges that connected the islands. This was a very cruel decision, but the wizards of Jordan have advised him no to do so, because after that his own armies would not be able to get from one island to another. So Jordan decided to burn as many bridges as possible so that is was still possible for his armies to get from any island to any other one.

Now the poor people of Ferry Kingdom wonder what bridges will be burned. Of course, they cannot learn that, because the list of bridges to be burned is kept in great secret. However, one old man said that you can help them to find the set of bridges that certainly will not be burned.

So they came to you and asked for help. Can you do that?

Input

The first line of the input file contains N and M — the number of islands and bridges in Ferry Kingdom respectively ($2 \le N \le 10\,000$, $1 \le M \le 100\,000$). Next M lines contain two different integer numbers each and describe bridges. Note that there can be several bridges between a pair of islands.

Output

On the first line of the output file print K — the number of bridges that will certainly not be burned. On the second line print K integers — the numbers of these bridges. Bridges are numbered starting from one, as they are given in the input file.

Example

bridges.in	bridges.out
6 7 1 2 2 3 2 4 5 4 1 3 4 5 3 6	2 3 7

Problem B. Hyperhuffman

Input file:	`huffman.in`
Output file:	`huffman.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

You might have heard about Huffman encoding — that is the coding system that minimizes the expected length of the text if the codes for characters are required to consist of an integral number of bits.

Let us recall codes assignment process in Huffman encoding. First the *Huffman tree* is constructed. Let the alphabet consist of N characters, i-th of which occurs P_i times in the input text. Initially all characters are considered to be active nodes of the future tree, i-th being marked with P_i. On each step take two active nodes with smallest marks, create the new node, mark it with the sum of the considered nodes and make them the children of the new node. Then remove the two nodes that now have parent from the set of active nodes and make the new node active. This process is repeated until only one active node exists, it is made the root of the tree.

Note that the characters of the alphabet are represented by the leaves of the tree. For each leaf node the length of its code in the Huffman encoding is the length of the path from the root to the node. The code itself can be constrcuted the following way: for each internal node consider two edges from it to its children. Assign 0 to one of them and 1 to another. The code of the character is then the sequence of 0s and 1s passed on the way from the root to the leaf node representing this character.

In this problem you are asked to detect the length of the text after it being encoded with Huffman method. Since the length of the code for the character depends only on the number of occurences of this character, the text itself is not given — only the number of occurences of each character. Characters are given from most rare to most frequent.

Note that the alphabet used for the text is quite huge — it may contain up to 500 000 characters.

Input

The first line of the input file contains N — the number of different characters used in the text ($2 \le N \le 500\,000$). The second line contains N integer numbers P_i — the number of occurences of each character ($1 \le P_i \le 10^9$, $P_i \le P_{i+1}$ for all valid i).

Output

Output the length of the text after encoding it using Huffman method, in bits.

Example

huffman.in	huffman.out
3 1 1 4	8

Problem C. Think Positive

Input file:	`positive.in`
Output file:	`positive.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

It is well known, that the year on planet Eisiem has n days. Of course, some days are very good for people, while some others are just horrible. Long observations have shown for each day of the year whether this day is good for most people, or bad.

The new president of the Planet Federation wants all people to be happy. He knows that good emotions have a tendency to accumulate, just like bad ones do. The New Year however is a special event and all emotions accumulated by this moment just disappear. Therefore the president wants to change the calendar on Eisiem and choose the new first day of the year, so that the positive emotions would prevail the whole year.

More precisely, for all i from 1 to n let a_i be 1 if i-th day is good for most people and -1 if it is bad. Let s_{jk} be the sum of a_i for all days from the j-th day of the year to the k-th, that is:

$$s_{jk} = \begin{cases} \sum_{i=j}^{k} a_i, \text{ if } k \geq j; \\ \sum_{i=j}^{n} a_i + \sum_{i=1}^{k} a_i, \text{ if } k < j. \end{cases}$$

President wants to find such j to order the j-th day to be the first day of the year, that s_{jk} is positive for all k from 1 to n. Since he wants several variants to choose from, he asks you to find all such j. Since he doesn't want to get too much information at once, first of all he wants to know the number of such j. That is exactly your task.

Input

The first line of the input file contains n — the number of days ($1 \leq n \leq 200\,000$). Next line contains n integer numbers — a_i.

Output

Output the number of different indices j, such that s_{jk} is positive for all k.

Example

positive.in	positive.out
5 1 -1 1 -1 1	1

Problem D. Quantization Problem

Input file:	`quant.in`
Output file:	`quant.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

When entering some analog data into a computer, this information must be quantized. Quantization transforms each measured value x to some other value $l(x)$ selected from the predefined set L of levels. Sometimes to reduce the influence of the levels set to the information, the group of levels sets L_i is used. The number of levels sets is usually chosen to be the power of 2.

When using the number of levels sets, some additional information should be used to specify which set was used for each quantization. However, providing this information may be too expensive — the better solution would be to choose more levels and use one set. To avoid the specification of the quantization set, the following technique is used. Suppose that n values $x_1, x_2, \ldots, x_n$ are to be quantized and the group of $m = 2^p$ levels sets $\{L_i\}_{i=0}^{m-1}$ each of size $s = 2^q$ is used to quantize it. After quantization x_j is replaced with some number $l_j =\in L_{f(j)}$. Instead of sending l_j, its ordinal number in $L_{f(j)}$ is usually sent, let k_j be the ordinal number of l_j in $L_{f(j)}$ (levels are numbered starting with 0). Take p least significant bits of k_j and say that the number $k_j\&(2^p - 1)$ is the number of the levels set that will be used for next quantization, that is $f(j + 1) = k_j\&(2^p - 1)$.

Since the least significant bits of k_j are usually distributed quite randomly, the sets used for quantization change often and weakly depend on values of quantized data, thus this technique provides the good way to perform the quantization.

Usually to perform the quantization the closest to the value level of the levels set is chosen. However, using the technique described, sometimes it pays off to choose not the optimal level, but some other one, the ordinal number of which has other least significant bits, thus choosing another levels set for next measure and providing better approximation of quantized values in the future. Let us call the *deviation* of quantization the value $\sum_{j=1}^{n} |x_j - l_j|$. Your task is given measures and levels sets to choose quantized value for each measure in such a way, that the deviation of quantization is minimal possible.

The first value is always quantized using set L_0.

Input

The first line of the input file contains n ($1 \le n \le 1000$). The second line contains n integer numbers x_i ranging from 1 to 10^6. The next line contains m and s ($1 \le m \le 128$, $m \le s \le 128$). Next m lines contain s integer numbers each — levels of the quantization sets given in increasing order for each set, all levels satisfy $1 \le level \le 10^6$.

Output

First output the minimal possible deviation of the quantization. Then output n integer numbers in range from 0 to $s - 1$. For each input value output the number of the level in the corresponding levels set (k_j) used for this number to achieve the quantization required.

Example

quant.in	quant.out
3 8 8 19 2 4 5 10 15 20 3 7 13 17	5 1 1 3

Problem E. Robbers

Input file:	`robbers.in`
Output file:	`robbers.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

N robbers have robbed the bank. As the result of their crime they chanced to get M golden coins. Before the robbery the band has made an agreement that after the robbery i-th gangster would get X_i/Y of all money gained. However, it turned out that M may be not divisible by Y.

The problem which now should be solved by robbers is what to do with the coins. They would like to share them fairly. Let us suppose that i-th robber would get K_i coins. In this case unfairness of this fact is $|X_i/Y - K_i/M|$. The total unfairness is the sum of all particular unfairnesses. Your task as the leader of the gang is to spread money among robbers in such a way that the total unfairness is minimized.

Input

The first line of the input file contains numbers N, M and Y ($1 \le N \le 1000, 1 \le M, Y \le 10000$). N integer numbers follow - X_i ($1 \le X_i \le 10000$, sum of all X_i is Y).

Output

Output N integer numbers — K_i (sum of all K_i must be M), so that the total unfairness is minimal.

Example

robbers.in	robbers.out
3 10 4 1 1 2	2 3 5

Problem F. Driving Straight

Input file:	straight.in
Output file:	straight.out
Time limit:	2 seconds
Memory limit:	64 megabytes

The city where Peter lives has the form of the rectangle with M streets running from east to west and N streets running from north to south. Recently, because of preparations for the celebration of the city's 3141-st birthday, some street sectors has been closed for driving.

Peter lives in the house next to point $(1, 1)$ and works next to the point (N, M). He always drives from home to work by his car using the shortest possible path. Of course, he can't drive through closed sectors. Since there can be many shortest pathes between his house and his work, he may choose any.

But Peter doesn't like to turn (he is an inexperienced driver), so he wants to choose the path using the following algorithm: starting from the point $(1, 1)$ he drives either northwards, or eastwards (wherever there is the shortest path available, if there are both, he may choose any). Whenever he comes to the junction he must decide where to go. If there is only one direction he can drive to stay on the shortest path, he must choose that direction. In the other case he would like to choose the direction giving priority to driving forward, that is, if he can drive forward and still stay on some shortest path, he would go forward. If he can't drive forward to stay on the shortest path, he would choose any available direction.

Help Peter to create the path from his house to his work using the rules described.

Input

The first line of the input file contains integer numbers M and N ($2 \le M, N \le 400$).

Next $2M - 1$ lines contain $2N - 1$ characters each, representing the city map. House blocks are marked with spaces, junctions with '+', open sectors with '-' and '|', closed sectors with spaces. Peter's house is at the lower-left corner of the map, his work is at the upper-right corner.

Output

On the first line of the output file print the direction Peter should choose first, 'N' or 'E'. On the second line output the sequence of latin letters 'F', 'L' and 'R' representing Peter's behaivour on junctions — go forward, turn left or right respectively. If there are several paths Peter can choose from, output any. You must output Peter's action on the junction even if he has no choice due to closed streets. It is guaranteed that there will always be the way for Peter to get from home to work.

Example

straight.in	straight.out
4 4 +-+ +-+ | | | + +-+-+ | | +-+-+-+ | | +-+-+-+	N RFLRL