A. Antipalindrome, 1 second, 256 megabytes, standard input, standard output

A string is a palindrome if it reads the same from the left to the right and from the right to the left. For example, the strings "kek", "abacaba", "r" and "papicipap" are palindromes, while the strings "abb" and "iq" are not.

A substring $s[l \ldots r]$ ($1 \leq l \leq r \leq |s|$) of a string $s = s_1 s_2 \ldots s_{|s|}$ is the string $s_l s_{l+1} \ldots s_r$.

Anna does not like palindromes, so she makes her friends call her Ann. She also changes all the words she reads in a similar way. Namely, each word s is changed into its longest substring that is not a palindrome. If all the substrings of s are palindromes, she skips the word at all.

Some time ago Ann read the word s. What is the word she changed it into?

Input

The first line contains a non-empty string s with length at most 50 characters, containing lowercase English letters only.

Output

If there is such a substring in s that is not a palindrome, print the maximum length of such a substring. Otherwise print 0.

Note that there can be multiple longest substrings that are not palindromes, but their length is unique.

input
mew
output
3

input
wuffuw
output
5

input
qqqqqqqq
output
0

"mew" is not a palindrome, so the longest substring of it that is not a palindrome, is the string "mew" itself. Thus, the answer for the first example is 3.

The string "uffuw" is one of the longest non-palindrome substrings (of length 5) of the string "wuffuw", so the answer for the second example is 5.

All substrings of the string "qqqqqqqq" consist of equal characters so they are palindromes. This way, there are no non-palindrome substrings. Thus, the answer for the third example is 0.

B. Businessmen Problems, 2 seconds, 256 megabytes, standard input, standard output

Two famous competing companies *ChemForces* and *TopChemist* decided to show their sets of recently discovered chemical elements on an exhibition. However they know that no element should be present in the sets of both companies.

In order to avoid this representatives of both companies decided to make an agreement on the sets the companies should present. The sets should be chosen in the way that maximizes the total income of the companies.

All elements are enumerated with integers. The *ChemForces* company has discovered n distinct chemical elements with indices $a_1, a_2, \ldots, a_n$, and will get an income of x_i Berland rubles if the i-th element from this list is in the set of this company.

The *TopChemist* company discovered m distinct chemical elements with indices $b_1, b_2, \ldots, b_m$, and it will get an income of y_j Berland rubles for including the j-th element from this list to its set.

In other words, the first company can present any subset of elements from $\{a_1, a_2, \ldots, a_n\}$ (possibly empty subset), the second company can present any subset of elements from $\{b_1, b_2, \ldots, b_m\}$ (possibly empty subset). There shouldn't be equal elements in the subsets.

Help the representatives select the sets in such a way that no element is presented in both sets and the total income is the maximum possible.

Input

The first line contains a single integer n ($1 \leq n \leq 10^5$) — the number of elements discovered by *ChemForces*.

The i-th of the next n lines contains two integers a_i and x_i ($1 \leq a_i \leq 10^9$, $1 \leq x_i \leq 10^9$) — the index of the i-th element and the income of its usage on the exhibition. It is guaranteed that all a_i are distinct.

The next line contains a single integer m ($1 \leq m \leq 10^5$) — the number of chemicals invented by *TopChemist*.

The j-th of the next m lines contains two integers b_j and y_j, ($1 \leq b_j \leq 10^9$, $1 \leq y_j \leq 10^9$) — the index of the j-th element and the income of its usage on the exhibition. It is guaranteed that all b_j are distinct.

Output

Print the maximum total income you can obtain by choosing the sets for both companies in such a way that no element is presented in both sets.

input

```
3
1 2
7 2
3 10
4
1 4
2 4
3 4
4 4
```

output

```
24
```

input

```
1
1000000000 239
3
14 15
92 65
35 89
```

output

```
408
```

In the first example *ChemForces* can choose the set $(3, 7)$, while *TopChemist* can choose $(1, 2, 4)$. This way the total income is $(10 + 2) + (4 + 4 + 4) = 24$.

In the second example *ChemForces* can choose the only element 10^9, while *TopChemist* can choose $(14, 92, 35)$. This way the total income is $(239) + (15 + 65 + 89) = 408$.

C. Useful Decomposition, 1 second, 256 megabytes, standard input, standard output

Ramesses knows a lot about problems involving trees (undirected connected graphs without cycles)!

He created a new useful tree decomposition, but he does not know how to construct it, so he asked you for help!

The decomposition is the splitting the edges of the tree in some simple paths in such a way that each two paths have at least one common vertex. Each edge of the tree should be in exactly one path.

Help Remesses, find such a decomposition of the tree or derermine that there is no such decomposition.

Input

The first line contains a single integer n ($2 \leq n \leq 10^5$) the number of nodes in the tree.

Each of the next $n - 1$ lines contains two integers a_i and b_i ($1 \leq a_i, b_i \leq n, a_i \neq b_i$) — the edges of the tree. It is guaranteed that the given edges form a tree.

Output

If there are no decompositions, print the only line containing "No".

Otherwise in the first line print "Yes", and in the second line print the number of paths in the decomposition m.

Each of the next m lines should contain two integers u_i, v_i ($1 \leq u_i, v_i \leq n, u_i \neq v_i$) denoting that one of the paths in the decomposition is the simple path between nodes u_i and v_i.

Each pair of paths in the decomposition should have at least one common vertex, and each edge of the tree should be presented in exactly one path. You can print the paths and the ends of each path in arbitrary order.

If there are multiple decompositions, print any.

input

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

output

```
Yes
1
1 4
```

input

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

output

```
No
```

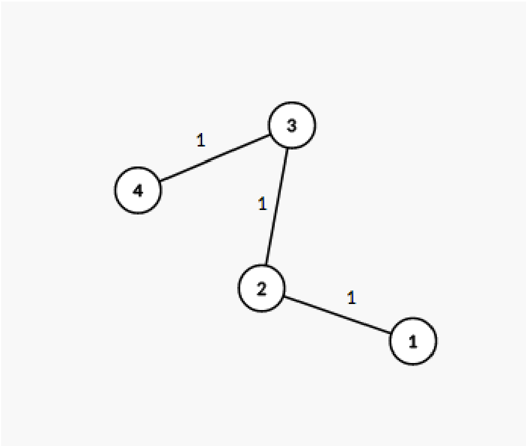
input

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

output

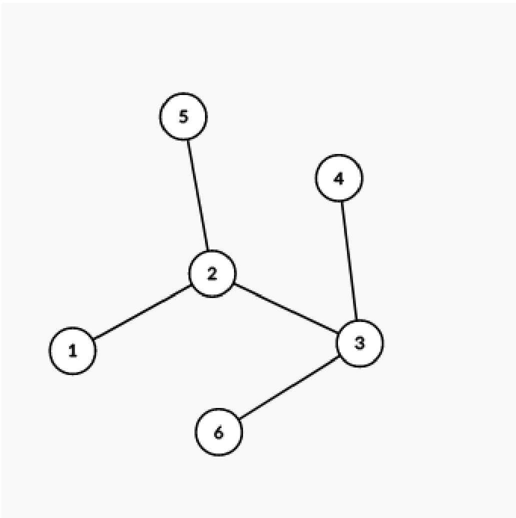
```
Yes
4
1 2
1 3
1 4
1 5
```

The tree from the first example is shown on the picture below:



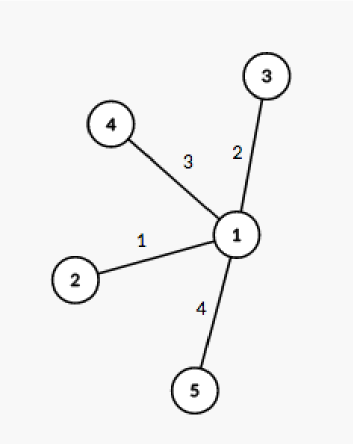
The number next to each edge corresponds to the path number in the decomposition. It is easy to see that this decomposition suits the required conditions.

The tree from the second example is shown on the picture below:



We can show that there are no valid decompositions of this tree.

The tree from the third example is shown on the picture below:



The number next to each edge corresponds to the path number in the decomposition. It is easy to see that this decomposition suits the required conditions.

D. Bookshelves, 1 second, 256 megabytes, standard input, standard output

Mr Keks is a typical white-collar in Byteland.

He has a bookshelf in his office with some books on it, each book has an integer positive price.

Mr Keks defines the value of a shelf as the sum of books prices on it.

Miraculously, Mr Keks was promoted and now he is moving into a new office.

He learned that in the new office he will have not a single bookshelf, but exactly k bookshelves. He decided that the beauty of the k shelves is the bitwise AND of the values of all the shelves.

He also decided that he won't spend time on reordering the books, so he will place several first books on the first shelf, several next books on the next shelf and so on. Of course, he will place at least one book on each shelf. This way he will put all his books on k shelves in such a way that the beauty of the shelves is as large as possible. Compute this maximum possible beauty.

Input

The first line contains two integers n and k ($1 \leq k \leq n \leq 50$) — the number of books and the number of shelves in the new office.

The second line contains n integers $a_1, a_2, \ldots a_n$, ($0 < a_i < 2^{50}$) — the prices of the books in the order they stand on the old shelf.

Output

Print the maximum possible beauty of k shelves in the new office.

input
```
10 4
9 14 28 1 7 13 15 29 2 31
```
output
```
24
```

input
```
7 3
3 14 15 92 65 35 89
```
output
```
64
```

In the first example you can split the books as follows:

$$(9 + 14 + 28 + 1 + 7) \& (13 + 15) \& (29 + 2) \& (31) = 24.$$

In the second example you can split the books as follows:

$$(3 + 14 + 15 + 92) \& (65) \& (35 + 89) = 64.$$

E. Addition on Segments, 2 seconds, 256 megabytes, standard input, standard output

Grisha come to a contest and faced the following problem.

You are given an array of size n, initially consisting of zeros. The elements of the array are enumerated from 1 to n. You perform q operations on the array. The i-th operation is described with three integers l_i, r_i and x_i ($1 \leq l_i \leq r_i \leq n$, $1 \leq x_i \leq n$) and means that you should add x_i to each of the elements with indices $l_i, l_i + 1, \ldots, r_i$. After all operations you should find the maximum in the array.

Grisha is clever, so he solved the problem quickly.

However something went wrong inside his head and now he thinks of the following question: "consider we applied some subset of the operations to the array. What are the possible values of the maximum in the array?"

Help Grisha, find all integers y between 1 and n such that if you apply some subset (possibly empty) of the operations, then the maximum in the array becomes equal to y.

Input

The first line contains two integers n and q ($1 \leq n, q \leq 10^4$) — the length of the array and the number of queries in the initial problem.

The following q lines contain queries, one per line. The i-th of these lines contains three integers l_i, r_i and x_i ($1 \leq l_i \leq r_i \leq n$, $1 \leq x_i \leq n$), denoting a query of adding x_i to the segment from l_i-th to r_i-th elements of the array, inclusive.

Output

In the first line print the only integer k, denoting the number of integers from 1 to n, inclusive, that can be equal to the maximum in the array after applying some subset (possibly empty) of the given operations.

In the next line print these k integers from 1 to n — the possible values of the maximum. Print these integers in **increasing order**.

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

input
```
7 2
1 5 1
3 7 2
```
output
```
3
1 2 3
```

input
10 3 1 1 2 1 1 3 1 1 6
output
6 2 3 5 6 8 9

Consider the first example. If you consider the subset only of the first query, the maximum is equal to 1. If you take only the second query, the maximum equals to 2. If you take the first two queries, the maximum becomes 3. If you take only the fourth query, the maximum becomes 4. If you take the fourth query and something more, the maximum becomes greater that n, so you shouldn't print it.

In the second example you can take the first query to obtain 1. You can take only the second query to obtain 2. You can take all queries to obtain 3.

In the third example you can obtain the following maximums:

- You can achieve the maximim of 2 by using queries: (1).
- You can achieve the maximim of 3 by using queries: (2).
- You can achieve the maximim of 5 by using queries: $(1, 2)$.
- You can achieve the maximim of 6 by using queries: (3).
- You can achieve the maximim of 8 by using queries: $(1, 3)$.
- You can achieve the maximim of 9 by using queries: $(2, 3)$.

F. Round Marriage,

3 seconds, 256 megabytes, standard input, standard output

It's marriage season in Ringland!

Ringland has a form of a circle's boundary of length L. There are n bridegrooms and n brides, and bridegrooms decided to marry brides.

Of course, each bridegroom should choose exactly one bride, and each bride should be chosen by exactly one bridegroom.

All objects in Ringland are located on the boundary of the circle, including the capital, bridegrooms' castles and brides' palaces. The castle of the i-th bridegroom is located at the distance a_i from the capital in clockwise direction, and the palace of the i-th bride is located at the distance b_i from the capital in clockwise direction.

Let's define the inconvenience of a marriage the maximum distance that some bride should walk along the circle from her palace to her bridegroom's castle in the shortest direction (in clockwise or counter-clockwise direction).

Help the bridegrooms of Ringland to choose brides in such a way that the inconvenience of the marriage is the smallest possible.

Input

The first line contains two integers n and L ($1 \le n \le 2 \cdot 10^5$, $1 \le L \le 10^9$) — the number of bridegrooms and brides and the length of Ringland.

The next line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < L$) — the distances from the capital to the castles of bridegrooms in clockwise direction.

The next line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i < L$) — the distances from the capital to the palaces of brides in clockwise direction.

Output

In the only line print the smallest possible inconvenience of the wedding, where the inconvenience is the largest distance traveled by a bride.

input
2 4 0 1 2 3
output
1

input
10 100 3 14 15 92 65 35 89 79 32 38 2 71 82 81 82 84 5 90 45 23
output
27

In the first example the first bridegroom should marry the second bride, the second bridegroom should marry the first bride. This way, the second bride should walk the distance of 1, and the first bride should also walk the same distance. Thus, the inconvenience is equal to 1.

In the second example let p_i be the bride the i-th bridegroom will marry. One of optimal p is the following: $(6, 8, 1, 4, 5, 10, 3, 2, 7, 9)$.

G. Magic multisets,

4 seconds, 256 megabytes, standard input, standard output

In the School of Magic in Dirtpolis a lot of interesting objects are studied on Computer Science lessons.

Consider, for example, the magic multiset. If you try to add an integer to it that is already presented in the multiset, each element in the multiset duplicates. For example, if you try to add the integer 2 to the multiset $\{1, 2, 3, 3\}$, you will get $\{1, 1, 2, 2, 3, 3, 3, 3\}$.

If you try to add an integer that is not presented in the multiset, it is simply added to it. For example, if you try to add the integer 4 to the multiset $\{1, 2, 3, 3\}$, you will get $\{1, 2, 3, 3, 4\}$.

Also consider an array of n initially empty magic multisets, enumerated from 1 to n.

You are to answer q queries of the form "add an integer x to all multisets with indices $l, l + 1, \ldots, r$" and "compute the sum of sizes of multisets with indices $l, l + 1, \ldots, r$". The answers for the second type queries can be large, so print the answers modulo 998244353.

Input

The first line contains two integers n and q ($1 \le n, q \le 2 \cdot 10^5$) — the number of magic multisets in the array and the number of queries, respectively.

The next q lines describe queries, one per line. Each line starts with an integer t ($1 \le t \le 2$) — the type of the query. If t equals 1, it is followed by three integers l, r, x ($1 \le l \le r \le n$, $1 \le x \le n$) meaning that you should add x to all multisets with indices from l to r inclusive. If t equals 2, it is followed by two integers l, r ($1 \le l \le r \le n$) meaning that you should compute the sum of sizes of all multisets with indices from l to r inclusive.

Output

For each query of the second type print the sum of sizes of multisets on the given segment.

The answers can be large, so print them modulo 998244353.

input

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

output

```
10
```

input

```
3 7
1 1 1 3
1 1 1 3
1 1 1 2
1 1 1 1
2 1 1
1 1 1 2
2 1 1
```

output

```
4
8
```

In the first example after the first two queries the multisets are equal to $[\{1,2\},\{1,2\},\{\},\{\}]$, after the third query they are equal to $[\{1,1,2,2\},\{1,1,2,2\},\{1\},\{1\}]$.

In the second example the first multiset evolves as follows:

$$\{\} \to \{3\} \to \{3,3\} \to \{2,3,3\} \to \{1,2,3,3\} \to \{1,1,2,2,3,3,3,3$$

.

H. K Paths, 4 seconds, 256 megabytes, standard input, standard output

You are given a tree of n vertices. You are to select k (not necessarily distinct) simple paths in such a way that it is possible to split all edges of the tree into three sets: edges not contained in any path, edges that are a part of exactly one of these paths, and edges that are parts of all selected paths, and the latter set should be non-empty.

Compute the number of ways to select k paths modulo 998244353.

The paths are enumerated, in other words, two ways are considered distinct if there are such i ($1 \le i \le k$) and an edge that the i-th path contains the edge in one way and does not contain it in the other.

Input

The first line contains two integers n and k ($1 \le n, k \le 10^5$) — the number of vertices in the tree and the desired number of paths.

The next $n-1$ lines describe edges of the tree. Each line contains two integers a and b ($1 \le a, b \le n, a \ne b$) — the endpoints of an edge. It is guaranteed that the given edges form a tree.

Output

Print the number of ways to select k enumerated not necessarily distinct simple paths in such a way that for each edge either it is not contained in any path, or it is contained in exactly one path, or it is contained in all k paths, and the intersection of all paths is non-empty.

As the answer can be large, print it modulo 998244353.

input

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

output

```
7
```

input

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

output

```
10
```

input

```
29 29
1 2
1 3
1 4
1 5
5 6
5 7
5 8
8 9
8 10
8 11
11 12
11 13
11 14
14 15
14 16
14 17
17 18
17 19
17 20
20 21
20 22
20 23
23 24
23 25
23 26
26 27
26 28
26 29
```

output

```
125580756
```

In the first example the following ways are valid:

- $((1,2),(1,2))$,
- $((1,2),(1,3))$,
- $((1,3),(1,2))$,
- $((1,3),(1,3))$,
- $((1,3),(2,3))$,
- $((2,3),(1,3))$,
- $((2,3),(2,3))$.

In the second example $k=1$, so all $n \cdot (n-1)/2 = 5 \cdot 4/2 = 10$ paths are valid.

In the third example, the answer is ≥ 998244353, so it was taken modulo 998244353, don't forget it!