ICPC Asia Pacific Regionals

icpc global sponsor
programming tools

icpc.foundation

icpc diamond
multi-regional sponsor

The 2021 ICPC Asia Yokohama Regional Contest

Official Problem Set

Problem A
Loop of Chocolate
Time Limit: 2 seconds

Let's make sweets of a fancy shape that is a loop of chocolate.

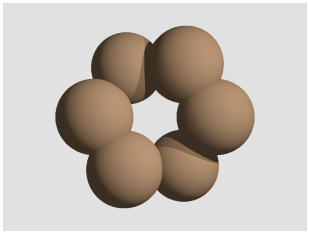
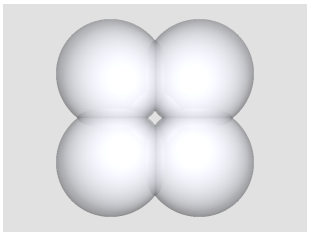
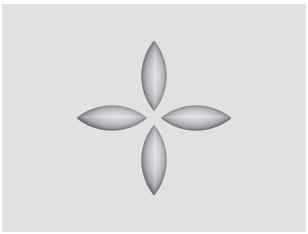


Figure A.1. A loop of chocolate formed by a union of six spheres

The shape of a loop is formed by a union of a number of spheres of the same size, where every sphere intersects with exactly two others.



(a) Union of four spheres



(b) Four intersections of the four spheres in (a)

Figure A.2. A loop of chocolate formed by a union of four spheres

Your job is to write a program that, for given the size and the positions of spheres, computes the total volume of the union of the spheres, i.e., the amount of chocolate required for filling the loop formed by the union.

[Hints] Two spheres of the same radius r intersect each other when the distance between their centers, d, is less than $2r$. The volume of the intersection is known to be

$$\frac{2}{3}\pi(r - d/2)^2(2r + d/2).$$

The volume of the sphere of radius r is $4\pi r^3/3$.

Input

The input consists of a single test case of the following format.

$n\ r$
$x_1\ y_1\ z_1$
$\vdots$
$x_n\ y_n\ z_n$

n and r are integers. n is the number of spheres ($4 \leq n \leq 100$). All the spheres has the same radius r ($2 \leq r \leq 100$). (x_k, y_k, z_k) indicates the coordinates of the center of the k-th sphere ($k = 1, \ldots, n$). All of x_k, y_k, and z_k are integers between -100 and 100, inclusive.

The k-th and the $k + 1$-th spheres intersect each other for $1 \leq k < n$. The 1-th and the n-th spheres also intersect. No other pairs of spheres intersect.

Output

Output in a line the volume of the union of the spheres. Relative error of the output should be within 10^{-7}.

Sample Input 1

```
6 9
20 0 10
20 10 0
10 20 0
0 20 10
0 10 20
10 0 20
```

Sample Output 1

```
17149.528141
```

Sample Input 2

```
4 12
10 10 0
10 -10 0
-10 -10 0
-10 10 0
```

Sample Output 2

```
27813.56696
```

Sample Input 3

```
6 9
23 3 13
20 10 0
10 20 0
3 23 13
0 10 20
10 0 20
```

Sample Output 3

```
17470.837758
```

Sample Input 4

```
4 2
0 0 0
3 0 0
3 3 0
0 3 0
```

Sample Output 4

```
122.52211349
```

Sample Input 5

```
8 70
100 100 0
0 100 0
-100 100 0
-100 0 0
-100 -100 0
0 -100 0
100 -100 0
100 0 0
```

Sample Output 5

```
10220648.1
```

Problem B
Lottery Fun Time

Time Limit: 2 seconds

The biggest fun of lottery is not in receiving the (usually a tiny amount of) prize money, but in dreaming of the big fortune you may possibly (that is, virtually never) receive.

You have a number of lottery tickets at hand, each with a six-digit number. All the numbers are different, of course. Tomorrow is the drawing day and the prizes are the following.

- The first prize of 300,000 yen is won by the ticket with exact match of all the six digits with the six-digit first prize winning number.
- The second prizes of 4,000 yen are won by all of the tickets with their last four digits matching the four-digit second prize winning number.
- The third prizes of 500 yen are won by all of the tickets with their last two digits matching any of the three two-digit third prize winning numbers.

The six digits on the lottery tickets and all of the winning numbers may start with zeros.

The last two digits of all the prize winning numbers are made different so that tickets winning the third prize cannot also win the first nor the second prizes. Note that this rule also made the last two digits of the first and the second prize winning numbers different.

To enjoy the climax of the lottery fun time, you decided to calculate the possible maximum amount you may win with your tickets. You have too many tickets to hand-calculate it, but it should also be your joy to write a program for making the calculation.

Input

The first line of the input has a positive integer n ($n \leq 10^5$), which is the number of tickets you have at hand. Each of the following n lines has the six-digit number on one of your tickets. All the n numbers are different from one another.

Output

Output in a line a single integer, which is the maximum possible total of winning prizes with the tickets you have.

Sample Input 1	Sample Output 1
7 034207 924837 372745 382947 274637 083907 294837	309500

Sample Input 2	Sample Output 2
10 012389 456789 234589 678989 890189 567889 123489 263784 901289 345689	304500

For the first sample, the following combination of prize winners allows the maximum total amount of 309500 yen.

- The first prize winner of 382947 makes one ticket with that number win 300000 yen.
- The second prize winner of 4837 makes two tickets, 924837 and 294837, win 4000 yen each.
- The third prize winners 07 and 45 make three tickets, 034207, 083907, and 372745, win 500 yen each. 37 cannot be a third prize winner, as the second prize winner, 4837, has the final two digits of 37. The ticket 274637, thus, wins nothing. You have no more tickets to win whatever the remaining third prize winner may be.

For the second sample, nine out of the ten tickets have the same last two digits, 89, and thus the third prize winner of 89 allows nine third prizes, totaling 4500 yen. This is more than the second prize of 4000 yen possibly won by one of these nine tickets. The only remaining ticket 263784 should, of course, win the first prize.

Problem C
Reversible Compression

Time Limit: 2 seconds

Data compression is an essential technology in our information society. It is to encode a given string into a (preferably) more compact code string so that it can be stored and/or transferred efficiently.

You are asked to design a novel compression algorithm such that even a code string is *reversed* it can be decoded into the given string.

Currently, you are considering the following specification as a candidate.

- A given string is a sequence of decimal digits, namely, `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, and `9`.
- A code string is a sequence of code words, and a code word consists of two decimal digits `A` and `L`. So, a code string is a sequence of even number of decimal digits.
- A code string $\mathtt{A}_1\mathtt{L}_1 \cdots \mathtt{A}_k\mathtt{L}_k$ is decoded into a string by the following procedure. Here, for brevity, a decimal digit ($\mathtt{A}_i$ or $\mathtt{L}_i$) is also treated as the single digit decimal integer it represents.

  1. $i \leftarrow 1$
  2. while (i is not greater than k) {
  3. if ($\mathtt{A}_i$ is zero) { output $\mathtt{L}_i$ }
  4. else if ($\mathtt{L}_i$ is zero) { do nothing }
  5. else if ($\mathtt{A}_i$ is larger than the number of digits output so far) { raise an error }
  6. else {
  7. repeat $\mathtt{L}_i$ times {
  8. output the $\mathtt{A}_i$-th of the already output digits counted backwards
  9. }
  10. }
  11. $i \leftarrow i + 1$
  12. }

For example, a code string `000125` is decoded into a string `0101010` as follows:

1. The first code word `00` outputs `0`.
2. The second code word `01` outputs `1`.

3. The first digit `2` of the last code word `25` means that the second digit in the already decoded digits, counted backwards, should be output. This should be repeated five times. For the first repetition, the decoded digits so far are `0` and `1`, and thus the second last digit is `0`, which is output. For the second repetition, digits decoded so far are `0`, `1`, and `0`, and the second last is `1`, which is output. Repeating this three more times outputs `0`, `1`, and `0`.

A sequence of code words that raises no error is a valid code string. A valid code string is said to be *reversible* when its reverse is also valid and both the original and its reverse are decoded into the same string.

For example, the code string `000125` is not reversible, because its reverse, `521000`, raises an error and thus is not valid. The code string `0010` is not reversible though its reverse is valid. It is decoded into `0`, but its reverse `0100` is decoded into `10`. On the other hand, `0015599100` is reversible, because this and its reverse, `0019955100`, are both decoded into `00000000000000000`.

You want to evaluate the performance of this compression method when applied to a variety of cases. Your task is to write a program that, for an arbitrary given digit string, finds the shortest reversible code string that is decoded into the given string.

Input

The input consists of a single line containing a non-empty string s of decimal digits. The length of s does not exceed 500.

Output

Output the shortest reversible code string that is decoded into s. If there are multiple solutions with the same shortest length, output the earliest in the lexicographic order. Note that it is easily shown that a reversible code string always exists for any input string (e.g., see Sample Output 4).

Sample Input 1	**Sample Output 1**
00000000000000000	0015599100

Sample Input 2	**Sample Output 2**
0101010	000122221000

Sample Input 3	**Sample Output 3**
123123123123123123123123123123	01020336699993302010

Sample Input 4	**Sample Output 4**
123456789	0102030405060708099080706050403020 10

Problem D
Wireless Communication Network
Time Limit: 5 seconds

We are setting up a wireless communication network in a mountain range. Communication stations are to be located on the summits. The summits are on a straight line and have different altitudes.

To minimize the cost, our communication network should have a tree structure, which is a connected graph with the least number of edges. The structure of the network, that is, which stations to communicate with which other stations, should be decided in advance.

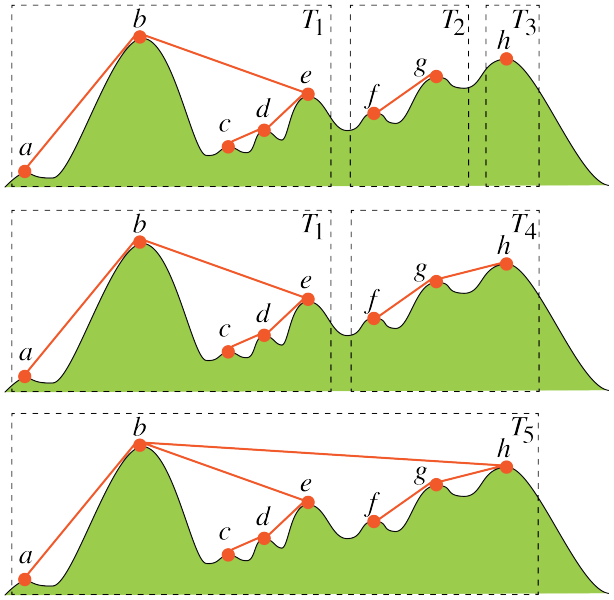
We construct the communication network as follows.

1. Initially, each station forms a tree consisting of only that station.
2. In each step, we merge two trees that are adjacent by making a link between two stations in different trees. Two trees are called *adjacent* when all the summits in between a summit in one tree and a summit in the other belong to one of these two trees. Stations to link are those on the highest summits of the two trees; they are uniquely determined because the altitudes of the summits are distinct.
3. Repeat the step 2 until all the stations are connected, directly or indirectly.

Figure D.1 depicts an example of the tree formation for Sample 1.

When a station detects an emergency event, an alert message should be broadcast to all the stations. On receiving a message, each station relays the message to all the stations with direct links, except for one from which it has come. The diameter of the network, that is, how many hops are sufficient to distribute messages initiated at any stations, depends on the choice of the two trees in the step 2 above.

To estimate the worst case delay of broadcast, we want to find the largest diameter of the network possibly constructed by the above procedure.



Obtained after some repetitions of step 2 with a certain series of choices. There are three trees, T_1, T_2, and T_3. Here, the station h forms the tree T_3 consisting of only the station h.

Obtained by linking stations g and h in step 2. Trees T_2 and T_3, with top summits g and h respectively, are adjacent.

Obtained by linking stations b and h in the next step 2, merging two adjacent trees. Now all the stations form a single tree.

Figure D.1. The tree formation for Sample 1

Input

The input consists of a single test case of the following format.

$$n$$
$$h_1$$
$$\vdots$$
$$h_n$$

Here, n is the number of communication stations ($3 \leq n \leq 10^6$), and h_i is an integer representing the altitude of the i-th station ($1 \leq h_i \leq n$). The altitudes of the stations are distinct, that is, $h_i \neq h_j$ if $i \neq j$.

Output

Output in a line a single integer representing the largest possible diameter of the tree.

Sample Input 1	**Sample Output 1**
8 1 8 2 3 5 4 6 7	6

Sample Input 2	Sample Output 2
4 1 2 3 4	3

Problem E
Planning Railroad Discontinuation

Time Limit: 5 seconds

The ICPC Kingdom has a large round lake in its center, and all of its cities are developed around the lake, forming a circle of cities. Because all of them are developed under the same urban plan, the cities are made quite similar. They have exactly the same subway networks. Some of their stations are also stations of inter-city bullet trains connecting two corresponding stations of adjacent cities. All of the train routes provide two-way traffic between two terminal stations without any intermediate stations.

The cities have the same number of subway stations and subway routes. In each city, the stations are numbered sequentially as $0, 1, 2, \ldots$. Whether two stations in a city are connected by a subway route depends only on their station numbers and not on the city. If a station v is connected to a station u in one city, so is in all the other cities. The travel distance between two stations v and u are the same for all the cities.

All the cities have exactly the same list of station numbers of bullet train stations. If a station s is a bullet train station in one city, so is in all the other cities. All the pairs of two bullet train stations of the same station number in two adjacent cities are connected by a bullet train route. If a station s is one end of a bullet train route, the station s in an adjacent city is the other end. There exist no other bullet train routes. One can travel between any two stations in the Kingdom via one or more subway and/or bullet train services.

Due to financial difficulties in recent years, the Kingdom plans to discontinue some of the subway and bullet train services to reduce the maintenance cost of the railroad system. The maintenance cost is the sum of maintenance costs of the subway routes and the bullet train routes. The maintenance cost of a subway route is the sum of base cost depending on the city and cost proportional to the travel distance of the route. The maintenance cost of a bullet train route depends only on the two cities connected by the route.

You are asked to make a plan to discontinue some of the routes that minimizes the total maintenance cost. Of course, all the pairs of the stations in the Kingdom should still be connected through one or more subway and/or bullet train services after the discontinuation.

Input

The input consists of a single test case of the following format.

n m
v_1 u_1 d_1
$\vdots$
v_m u_m d_m
l
a_0 b_0
$\vdots$
a_{l-1} b_{l-1}
r
s_1
$\vdots$
s_r

Here, n is an integer satisfying $2 \leq n \leq 10^4$, which is the number of the subway stations in a city. The stations in each city are numbered from 0 to $n-1$. m is an integer satisfying $1 \leq m \leq 10^5$, which is the number of the subway routes in a city. The following m lines are information on the subway system in a city. The i-th of the m lines has three integers v_i, u_i, and d_i, satisfying $0 \leq v_i < n$, $0 \leq u_i < n$, $v_i \neq u_i$, and $1 \leq d_i \leq 10^9$. They mean that a subway route connects stations numbered v_i and u_i, and its maintenance cost proportional to the travel distance is d_i. There is at most one subway route between two stations. All the subway stations in a city are connected directly or indirectly via one or more subway routes.

l in the next line is an integer satisfying $3 \leq l \leq 10^5$, which is the number of cities in the Kingdom. The cities are numbered from 0 to $l-1$, and the city 0 is also called the city l. For each $0 \leq j \leq l-1$, the cities j and $j+1$ are adjacent. a_j and b_j in the following l lines are integers satisfying $1 \leq a_j \leq 10^9$ and $1 \leq b_j \leq 10^9$. a_j is the maintenance cost of a bullet train route connecting the cities j and $j+1$. b_j is the base cost of the subway routes in the city j.

r is an integer satisfying $1 \leq r \leq n$, which is the number of the bullet train stations in each city. $s_1, s_2, \ldots, s_r$ are the station numbers of the bullet train stations in each city.

Output

Output in a single line the integer representing the minimum maintenance cost of the railroad system after the discontinuation that minimizes the cost while keeping all the stations connected, directly or indirectly.

Sample Input 1

```
2 1
0 1 3
3
6 1
4 2
5 3
1
1
```

Sample Output 1

```
24
```

Sample Input 2

```
3 3
0 1 7
1 2 8
2 0 5
4
8 1
5 1
9 3
7 3
2
1
2
```

Sample Output 2

```
76
```

Sample Input 3	Sample Output 3
5 8 0 1 1 2 1 2 4 0 5 3 4 7 3 2 8 0 2 4 4 1 3 2 4 6 6 3 2 9 3 7 1 5 3 3 2 5 2 4 0 2 3 4	120

Problem F
It's Surely Complex
Time Limit: 30 seconds

As you know, a complex number is often represented as the sum of a real part and an imaginary part. $3 + 2i$ is such an example, where 3 is the real part, 2 is the imaginary part, and i is the imaginary unit.

Given a prime number p and a positive integer n, your program for this problem should output the product of all the complex numbers satisfying the following conditions.

- Both the real part and the imaginary part are non-negative integers less than or equal to n.
- At least one of the real part and the imaginary part is not a multiple of p.

For instance, when $p = 3$ and $n = 1$, the complex numbers satisfying the conditions are $1\ (= 1 + 0i)$, $i\ (= 0 + 1i)$, and $1 + i\ (= 1 + 1i)$, and the product of these numbers, that is, $1 \times i \times (1 + i)$ is $-1 + i$.

Input

The input consists of a single test case of the following format.

$p\ n$

p is a prime number less than 5×10^5. n is a positive integer less than or equal to 10^{18}.

Output

Output two integers separated by a space in a line. When the product of all the complex numbers satisfying the given conditions is $a + bi$, the first and the second integers should be a modulo p and b modulo p, respectively. Here, x modulo y means the integer z between 0 and $y - 1$, inclusive, such that $x - z$ is divisible by y.

As exemplified in the main section, when $p = 3$ and $n = 1$, the product to be calculated is $-1 + i$. However, since -1 modulo 3 is 2, 2 and 1 are displayed in Sample Output 1.

Sample Input 1	Sample Output 1
3 1	2 1

Sample Input 2	Sample Output 2
5 5	0 0

Sample Input 3	Sample Output 3
499979 1000000000000000000	486292 0

Problem G
Genealogy of Puppets
Time Limit: 2 seconds

The *Inventive and Creative Puppets Corporation (ICPC)* sells a variety of educational *puppets* for children, and also for grown-ups who were once children and still remember it. To celebrate its centennial anniversary, ICPC has decided to offer for sale a collection of many of its most popular puppets in its history of one hundred years, which will be a collectors' envy for sure.

Each puppet has a ring attached to its head, with which it can be hung below one of two toes of another puppet. At most one puppet can be hung on one toe. As puppets do not feel comfortable in handstand positions, loops of puppets should be avoided. A *tree* of puppets can thus be made by hanging all the puppets to a toe of another puppet, leaving the ring of the topmost puppet to be hung on the wall.

You are enthusiastic in ICPC puppets since childhood. You like imagining genealogies of the puppets, assuming that puppets are hung on a parent puppet. You also imagine personalities of puppets, and decided to obey the following rules on forming the tree:

- each puppet has its own constraints on the number of children, that are its minimum and maximum, and
- if a puppet has any children, at least one of them should have release date later than the parent.

Note that, if a puppet has two children, one of them may have its release date earlier than the parent.

You want to write a program to calculate how many *different* trees can be made by the collection, satisfying the rules. Two trees are considered different if any of the puppets are hung on different parents, or hung on different toes of the same parent.

Input

The input consists of a single test case of the following format.

n
x_1 y_1
$\vdots$
x_n y_n

The first line contains an integer n ($2 \leq n \leq 300$), representing the number of puppets. The i-th line of the following n lines describes the personality of a puppet. Lines are in the order of the release dates of the puppets, from older to newer. Two integers in the line, x_i and y_i, are the minimum number and the maximum number of children of the puppet, respectively. $0 \leq x_i \leq y_i \leq 2$ holds.

Output

Output a single integer in a line which is the number of the different trees satisfying the rules modulo $10^9 + 7$.

Sample Input 1	**Sample Output 1**
3 0 1 1 2 0 2	6

Sample Input 2	**Sample Output 2**
2 0 2 1 2	0

For Sample Input 1, there are 6 possible trees satisfying the rules shown in Figure G.1.

For Sample Input 2, no trees can satisfy the rules.

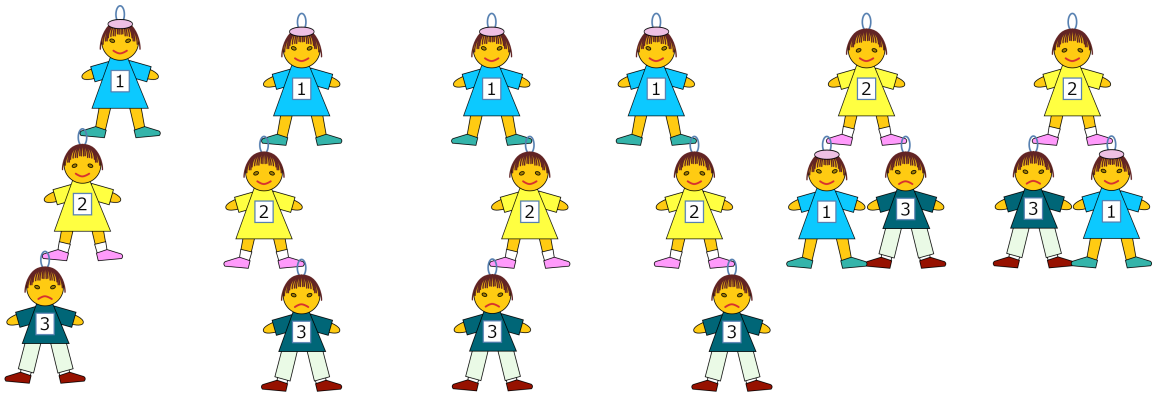


Figure G.1. Trees satisfying the rules for Sample Input 1. The numbers on the puppets represent the release order.

Problem H
Cancer DNA

Time Limit: 10 seconds

The *Investigation Center for Potential Cancer (ICPC)* found out patterns of a DNA sequence that cause cancer! We would like you to write a computer program that approximates the probability that a random DNA sequence matches one of the given patterns.

A DNA sequence can be represented by a string consisting of four letters, '`A`', '`G`', '`C`', and '`T`'. A *DNA pattern* is a string over the same four letters plus '`?`'. We say that a DNA pattern *matches* a DNA sequence of the same length if each of the characters in the pattern is either '`?`' or is the same as the character at the corresponding position in the DNA sequence. For example, a pattern "`AC?`" matches DNA sequences "`ACA`", "`ACG`", "`ACC`", and "`ACT`".

Your task is to write a program that, given a set of DNA patterns of the same length, computes the probability that a uniformly random DNA sequence of the same length matches any of the given patterns. A multiplicative error up to 5% is permissible.

Input

The input consists of a single test case of the following format.

n m
P_1
$\vdots$
P_m

The first line of the input contains two positive integers n and m such that $1 \leq n \leq 100$ and $1 \leq m \leq 30$ hold. The next m lines contain m patterns $P_1, \ldots, P_m$. Each pattern P_i is a string of length n over '`A`', '`G`', '`C`', '`T`', and '`?`'.

Output

Let S be a DNA sequence of length n chosen uniformly at random. Let w be the probability that S matches at least one of $P_1, \ldots,$ and P_m. The output is a real number v that approximates w.

The output v is judged to be correct if v approximates w within a multiplicative error of 5%, i.e.,

$$0.95 \times w \ \leq \ v \ \leq \ 1.05 \times w.$$

v should be represented either with or without exponent component. For example, 0.045 can be represented as `4.5e-2` or `0.045`.

Sample Input 1	Sample Output 1
3 1 AC?	0.0625

Sample Input 2	Sample Output 2
6 2 AC??A? A??A?T	0.0302734375

Sample Input 3	Sample Output 3
30 1 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	8.673617379884035e-19

In the first sample, there are 4^3 DNA sequences of length 3. There are 4 DNA sequences, "ACA", "ACG", "ACC", and "ACT", that match the pattern "AC?". Thus, the probability is $4/4^3 = 0.0625$. Any real number between 0.059375 and 0.065625 is accepted as a correct output.

As in the third sample, the probability can be a small real number. Note that "0" is not a correct output, as 0 is less than 95% of the precise probability.

Problem I
"Even" Division

Time Limit: 4 seconds

If you talk about an even division in the usual sense of the words, it means dividing a thing equally. Today, you need to think about something different. A graph is to be divided into subgraphs with their numbers of nodes being even, that is, multiples of two.

You are given an undirected connected graph with even number of nodes. The given graph is to be divided into its subgraphs so that all the subgraphs are connected and with even number of nodes, until no further such division is possible. Figure I.1 illustrates an example. The original graph of 8 nodes is divided into subgraphs with 4, 2, and 2 nodes. All of them have *even* numbers of nodes.

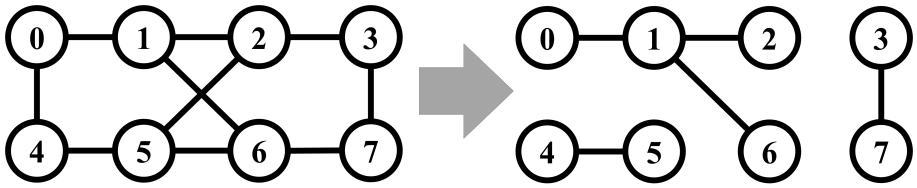


Figure I.1. Example of a division (Sample Input/Output 1)

To put it mathematically, an *even* division of a graph is the set of subsets $V_1, \ldots, V_k$ of the graph nodes satisfying the following conditions.

1. $V_1 \cup \cdots \cup V_k$ is the set of all the nodes of the input graph, and $V_i \cap V_j = \emptyset$ if $i \neq j$.
2. Each V_i is non-empty, and has an even number of elements.
3. Each V_i induces a connected subgraph. In other words, any nodes in V_i are reachable from each other by using only the edges of the input graph connecting two nodes in V_i.
4. There is no further division. For any $U_1 \cup U_2 = V_i$, the division obtained by replacing V_i with the two sets, U_1 and U_2, does not satisfy either of the conditions 1, 2, or 3.

Your task is to find an *even* division of the given graph.

Input

The input consists of a single test case of the following format.

$n\ m$
$x_1\ y_1$
$\vdots$
$x_m\ y_m$

The first line consists of two integers n and m. The first integer n ($2 \leq n \leq 10^5$) is an even number representing the number of the nodes of the graph to be divided. The second integer m ($n - 1 \leq m \leq 10^5$) is the number of the edges of the graph.

The nodes of the graph are numbered from 0 to $n - 1$.

The subsequent m lines represent the edges of the graph. A line $x_i\ y_i$ ($0 \leq x_i < y_i < n$) means that there is an edge connecting the two nodes x_i and y_i. There are no duplicated edges. The input graph is guaranteed to be connected.

Output

If there exists an *even* division of the node set of the given graph into subsets $V_1, \ldots, V_k$, print k in the first line of the output. The next k lines should describe the subsets $V_1, \ldots, V_k$. The order of the subsets does not matter. The i-th of them should begin with the size of V_i, followed by the node numbers of the elements of V_i, separated by a space. The order of the node numbers does not matter either.

If there are multiple even divisions, any of them are acceptable.

Sample Input 1

```
8 9
0 1
1 2
2 3
0 4
1 6
3 7
4 5
5 6
6 7
```

Sample Output 1

```
3
2 3 7
2 4 5
4 0 1 2 6
```

Sample Input 2

```
2 1
0 1
```

Sample Output 2

```
1
2 0 1
```

In the Sample 2, the singleton set of the set of the nodes of the original graph is already an even division.

Problem J
The Cross Covers Everything
Time Limit: 3 seconds

A cross-shaped infinite area on the x-y plane can be specified by two distinct points as depicted on the figure below.

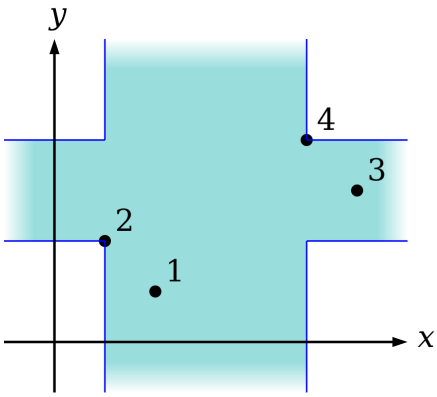


Figure J.1. The cross area specified by two points numbered 2 and 4

Given a set of points on the plane, you are asked to figure out how many pairs of the points form a cross-shaped area that covers all the points. To be more precise, when n points with coordinates (x_i, y_i) $(i = 1, \ldots, n)$ are given, the ordered pair $\langle p, q \rangle$ is said to cover a point (x, y) if $x_p \leq x \leq x_q$, $y_p \leq y \leq y_q$, or both hold. Your task is to find how many pairs $\langle p, q \rangle$ cover all the n points. No two given points have the same x-coordinate nor the same y-coordinate.

Input

The input consists of a single test case of the following format.

n
x_1 y_1
$\vdots$
x_n y_n

The first line contains an integer n $(2 \leq n \leq 2 \times 10^5)$, which is the number of points given. Two integers x_i and y_i in the i-th line of the following n lines are the coordinates of the i-th point $(1 \leq x_i \leq 10^6, 1 \leq y_i \leq 10^6)$. You may assume that $x_j \neq x_k$ and $y_j \neq y_k$ hold for all $j \neq k$.

Output

Print in a line the number of ordered pairs of points that satisfy the condition.

Sample Input 1

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

Sample Output 1

```
4
```

Sample Input 2

```
20
15 9
14 13
2 7
10 5
11 17
13 8
9 3
8 12
6 4
19 18
12 1
3 2
5 10
18 11
4 19
20 16
16 15
1 14
7 6
17 20
```

Sample Output 2

```
9
```

Figure J.1 depicts the cross specified by two points numbered 2 and 4, that are the second and the fourth points of the Sample Input 1. This is one of the crosses covering all the points.

Problem K
Distributing the Treasure

Time Limit: 4 seconds

You are the leader of a treasure hunting team. Under your great direction, your team has made a big success in a quest, and got a lot of treasure. The only, but crucial, remaining issue is how to distribute the obtained treasure to the team members.

The excavated treasure includes a variety of precious items: gold ingots, jewelry with brilliant gem stones, exquisite craft works, etc. Each team member individually estimates the values of the items. The estimated values are consistent, in that, for any pair of two items, when some member estimates one to be strictly higher than the other, no member estimates oppositely, although some may give equal estimates.

All the members are sensible and thus understand the difficulty of even distribution of the items. Hence, no member will complain simply because, based on the member's own estimation, the sum of the values of the member's share is lower than that of another member's share. The members, however, may get angry if their own shares look unreasonably shabby compared to some other member's; what they cannot stand is when their own shares are estimated strictly lower than the share of that other member even after getting rid of one item with the least estimated value.

Your last mission as the leader is to decide who receives which items so that no member gets angry. Some of the members may receive nothing as long as they do not get angry.

Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
n\ m \\
v_{1,1}\ \cdots\ v_{1,m} \\
\ \ \vdots \\
v_{n,1}\ \cdots\ v_{n,m}
\end{array}
$$

The first line of the input has two positive integers n and m whose product does not exceed 2×10^5; n is the number of members of your treasure hunting team, and m is the number of treasure items. The members and items are numbered 1 through n and 1 through m, respectively. The i-th of the following n lines has m positive integers not exceeding 2×10^5 in descending order: $v_{i,1} \geq v_{i,2} \geq \cdots \geq v_{i,m}$. Here, $v_{i,j}$ is the value of the item j estimated by the member i.

Output

If you can distribute the items to the members without making any member angry, output such a distribution in a line as m positive integers $x_1, x_2, \ldots, x_m$ separated by spaces. Here, $x_j = i$ means that the member i receives the item j. If two or more such distributions exist, any of them shall be deemed correct. If there is no way to distribute the items without making any member angry, just output `0` in a line.

Sample Input 1

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

Sample Output 1

```
1 2 1
```

Sample Input 2

```
1 7
64 32 16 8 4 2 1
```

Sample Output 2

```
1 1 1 1 1 1 1
```

Let $V_i(X)$ denote the sum of the values of items in the set X estimated by the member i.

In Sample 1, $V_1(\{1,3\})$ is $4+1=5$, $V_1(\{2\})$ is 2, $V_2(\{1,3\})$ is $3+3=6$, and $V_2(\{2\})$ is 3. The distribution shown as output does not make the member 1 angry as $V_1(\{1,3\}) \geq V_1(\{2\})$. The member 2 does not get angry either even though $V_2(\{2\}) < V_2(\{1,3\})$ holds. If the member 1 waives one of the two items, the sum of the values of items received by the member 1 would become $V_2(\{1\}) = 3$ or $V_2(\{3\}) = 3$, neither of which is higher than $V_2(\{2\}) = 3$.

Note that their shares should not be exchanged. Suppose that the member 1 receives $\{2\}$ and the member 2 receives $\{1,3\}$. This makes the member 1 angry because $V_1(\{2\}) = 2 < 4 = V_1(\{1\})$ holds, meaning that, even if the member 2 waives the item 3, which is estimated to be the lesser of $\{1,3\}$, the value of the sole remaining item 1 is still estimated higher than $V_1(\{2\}) = 2$.

In Sample 2, you are the sole member of your team, so you can take them all. Congratulations!