A. Johnny and Contribution

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Today Johnny wants to increase his contribution. His plan assumes writing n blogs. One blog covers one topic, but one topic can be covered by many blogs. Moreover, some blogs have references to each other. Each pair of blogs that are connected by a reference has to cover different topics because otherwise, the readers can notice that they are split just for more contribution. Set of blogs and bidirectional references between some pairs of them is called blogs network.

There are n different topics, numbered from 1 to n sorted by Johnny's knowledge. The structure of the blogs network is already prepared. Now Johnny has to write the blogs in some order. He is lazy, so each time before writing a blog, he looks at it's already written neighbors (the blogs referenced to current one) and chooses the topic with the smallest number which is not covered by neighbors. It's easy to see that this strategy will always allow him to choose a topic because there are at most $n-1$ neighbors.

For example, if already written neighbors of the current blog have topics number 1, 3, 1, 5, and 2, Johnny will choose the topic number 4 for the current blog, because topics number 1, 2 and 3 are already covered by neighbors and topic number 4 isn't covered.

As a good friend, you have done some research and predicted the best topic for each blog. Can you tell Johnny, in which order he has to write the blogs, so that his strategy produces the topic assignment chosen by you?

Input

The first line contains two integers n $(1 \leq n \leq 5 \cdot 10^5)$ and m $(0 \leq m \leq 5 \cdot 10^5)$ — the number of blogs and references, respectively.

Each of the following m lines contains two integers a and b ($a \neq b$; $1 \leq a, b \leq n$), which mean that there is a reference between blogs a and b. It's guaranteed that the graph doesn't contain multiple edges.

The last line contains n integers $t_1, t_2, \ldots, t_n$, i-th of them denotes desired topic number of the i-th blog ($1 \leq t_i \leq n$).

Output

If the solution does not exist, then write -1. Otherwise, output n distinct integers $p_1, p_2, \ldots, p_n$ $(1 \leq p_i \leq n)$, which describe the numbers of blogs in order which Johnny should write them. If there are multiple answers, print any.

Examples

input

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

output

```
2 1 3
```

input

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

output

```
-1
```

input

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

output

```
2 5 1 3 4
```

Note

In the first example, Johnny starts with writing blog number 2, there are no already written neighbors yet, so it receives the first topic. Later he writes blog number 1, it has reference to the already written second blog, so it receives the second topic. In the end, he writes blog number 3, it has references to blogs number 1 and 2 so it receives the third topic.

Second example: There does not exist any permutation fulfilling given conditions.

Third example: First Johnny writes blog 2, it receives the topic 1. Then he writes blog 5, it receives the topic 1 too because it doesn't have reference to single already written blog 2. Then he writes blog number 1, it has reference to blog number 2 with topic 1, so it receives the topic 2. Then he writes blog number 3 which has reference to blog 2, so it receives the topic 2. Then he ends with writing blog number 4 which has reference to blog 5 and receives the topic 2.

B. Johnny and Grandmaster

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Johnny has just found the new, great tutorial: "*How to become a grandmaster?*". The tutorial tells many strange and unexpected for Johnny things, such as you have to be patient or that very important is solving many harder and harder problems.

The boy has found an online judge with tasks divided by topics they cover. He has picked p^{k_i} problems from i-th category (p is his favorite number). He wants to solve them in two weeks (the patience condition is too hard for Johnny, so for simplicity, he looks only at easy tasks, which can be solved in such a period). Now our future grandmaster has to decide which topics to cover first and which the second week. Help him assign topics in such a way, that workload is balanced.

Formally, given n numbers p^{k_i}, the boy wants to divide them into two disjoint sets, minimizing the absolute difference between sums of numbers in each set. Find the minimal absolute difference. Output the result modulo $10^9 + 7$.

Input

Input consists of multiple test cases. The first line contains one integer t $(1 \leq t \leq 10^5)$ — the number of test cases. Each test case is described as follows:

The first line contains two integers n and p $(1 \leq n, p \leq 10^6)$. The second line contains n integers k_i $(0 \leq k_i \leq 10^6)$.

The sum of n over all test cases doesn't exceed 10^6.

Output

Output one integer — the reminder of division the answer by $1\,000\,000\,007$.

Example

input

```
4
5 2
2 3 4 4 3
3 1
2 10 1000
4 5
0 1 1 100
1 8
89
```

output

```
4
1
146981438
747093407
```

Note

You have to minimize the difference, not it's remainder. For example, if the minimum difference is equal to 2, but there is also a distribution where the difference is $10^9 + 8$, then the answer is 2, not 1.

In the first test case of the example, there're the following numbers: 4, 8, 16, 16, and 8. We can divide them into such two sets: $4, 8, 16$ and $8, 16$. Then the difference between the sums of numbers in sets would be 4.

C. Johnny and Megan's Necklace

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Johnny's younger sister Megan had a birthday recently. Her brother has bought her a box signed as "*Your beautiful necklace — do it yourself!*". It contains many necklace parts and some magic glue.

The necklace part is a chain connecting two pearls. Color of each pearl can be defined by a non-negative integer. The magic glue allows Megan to merge two pearls (possibly from the same necklace part) into one. The beauty of a connection of pearls in colors u and v is defined as follows: let 2^k be the greatest power of two dividing $u \oplus v$ — exclusive or of u and v. Then the beauty equals k.

If $u = v$, you may assume that beauty is equal to 20.

Each pearl can be combined with another at most once. Merging two parts of a necklace connects them. Using the glue multiple times, Megan can finally build the necklace, which is a cycle made from connected necklace parts (so every pearl in the necklace is combined with precisely one other pearl in it). The beauty of such a necklace is the minimum beauty of a single connection in it. The girl wants to use all available necklace parts to build **exactly one** necklace consisting of **all of them** with the largest possible beauty. Help her!

Input

The first line contains n $(1 \le n \le 5 \cdot 10^5)$ — the number of necklace parts in the box. Each of the next n lines contains two integers a and b $(0 \le a, b < 2^{20})$, which denote colors of pearls presented in the necklace parts. Pearls in the i-th line have indices $2i - 1$ and $2i$ respectively.

Output

The first line should contain a single integer b denoting the maximum possible beauty of a necklace built from all given parts.

The following line should contain $2n$ distinct integers p_i $(1 \le p_i \le 2n)$ — the indices of initial pearls in the order in which they appear on a cycle. Indices of pearls belonging to the same necklace part have to appear at neighboring positions in this permutation (so $1\,4\,3\,2$ is not a valid output, whereas $2\,1\,4\,3$ and $4\,3\,1\,2$ are). If there are many possible answers, you can print any.

Examples

input

```
5
13 11
11 1
3 5
17 1
9 27
```

output

```
3
8 7 9 10 5 6 1 2 3 4
```

input

```
5
13 11
11 1
3 5
17 1
7 29
```

output

```
2
8 7 10 9 5 6 4 3 2 1
```

input

```
1
1 1
```

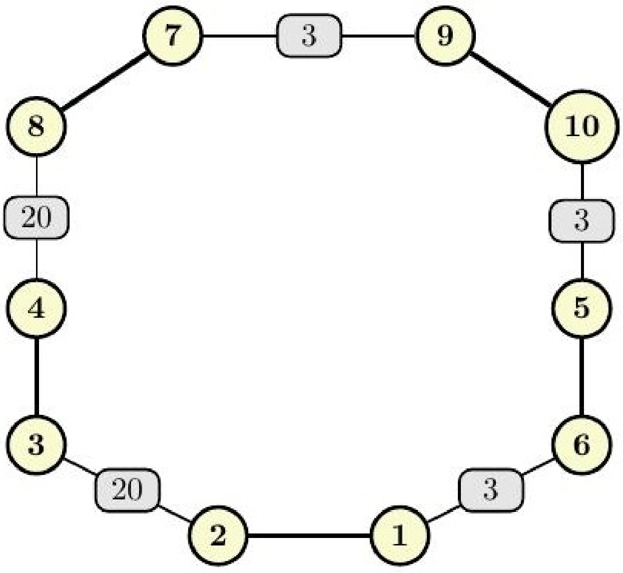
output

```
20
2 1
```

Note

In the first example the following pairs of pearls are combined: $(7, 9)$, $(10, 5)$, $(6, 1)$, $(2, 3)$ and $(4, 8)$. The beauties of connections equal correspondingly: 3, 3, 3, 20, 20.

The following drawing shows this construction.



D. Johnny and James

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

James Bond, Johnny's favorite secret agent, has a new mission. There are n enemy bases, each of them is described by its coordinates so that we can think about them as points in the Cartesian plane.

The bases can communicate with each other, sending a signal, which is the ray directed from the chosen point to the origin or in the opposite direction. The exception is the central base, which lies at the origin and can send a signal in any direction.

When some two bases want to communicate, there are two possible scenarios. If they lie on the same line with the origin, one of them can send a signal directly to the other one. Otherwise, the signal is sent from the first base to the central, and then the central sends it to the second base. We denote the distance between two bases as the total Euclidean distance that a signal sent between them has to travel.

Bond can damage all but some k bases, which he can choose arbitrarily. A damaged base can't send or receive the direct signal but still can pass it between two working bases. In particular, James can damage the central base, and the signal **can still be sent** between any two undamaged bases as before, so the distance between them remains the same. What is the maximal sum of the distances between all pairs of remaining bases that 007 can achieve by damaging exactly $n - k$ of them?

Input

The first line contains two integers n and k $(2 \leq k \leq n \leq 5 \cdot 10^5)$ — the total number of bases and number of bases that have to remain, respectively.

Each of the next n lines contains two integers x and y $(-10^9 \leq x, y \leq 10^9)$, i-th line contains coordinates of the i-th base. You can assume that no two points coincide and that one of them is $(0, 0)$.

Output

You should output one number — the maximal possible sum of distances between all pairs of some k from given bases. Your answer will be accepted if the absolute or relative error is less than 10^{-6}.

Examples

input

```
6 2
0 0
1 1
2 2
3 3
0 1
0 2
```

output

```
6.24264069
```

input

```
6 5
0 0
1 1
2 2
```

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

output

```
32.62741700
```

input

```
13 10
0 0
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
0 -2
0 1
0 2
```

output

```
237.00000000
```

input

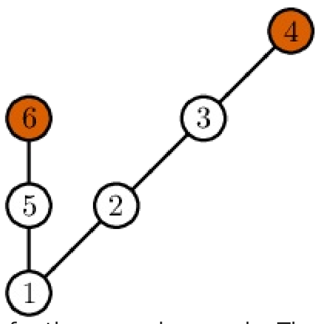
```
10 5
2 2
4 4
3 5
6 10
0 5
0 0
5 0
10 0
0 10
4 7
```

output

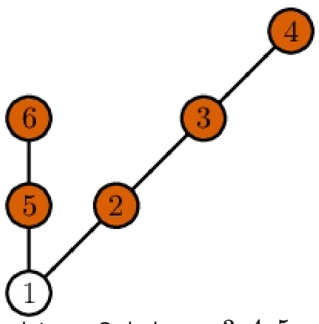
```
181.52406315
```

Note

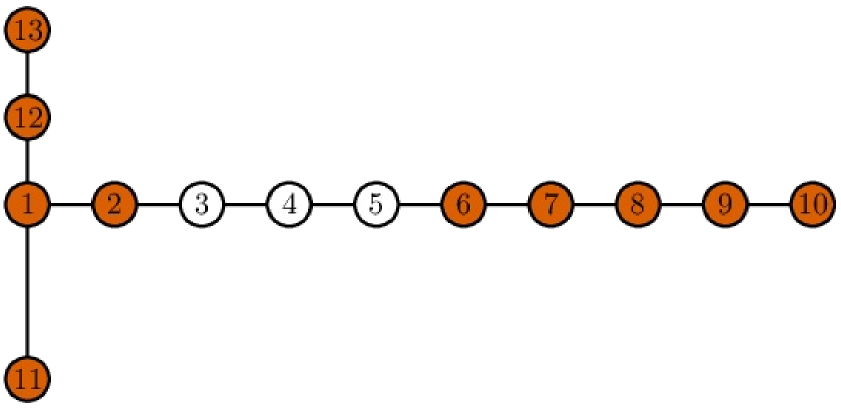
In the first example, in an optimal solution Bond **doesn't** destroy bases with indices 4 and 6 (marked in orange):



The following picture represents an optimal solution for the second example. These bases are are **not** destroyed: $2, 3, 4, 5, 6$ (marked in orange).



An optimal solution for the third test is visible in the picture. Only bases $3, 4, 5$ are destroyed. Again, the **not** destroyed bases are marked in orange.



E. James and the Chase

time limit per test: 2.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

James Bond has a new plan for catching his enemy. There are some cities and **directed** roads between them, such that it is possible to travel between any two cities using these roads. When the enemy appears in some city, Bond knows her next destination but has no idea which path she will choose to move there.

The city a is called interesting, if for each city b, there is exactly one simple path from a to b. By a simple path, we understand a sequence of distinct cities, such that for every two neighboring cities, there exists a directed road from the first to the second city.

Bond's enemy is the mistress of escapes, so only the chase started in an interesting city gives the possibility of catching her. James wants to arrange his people in such cities. However, if there are not enough interesting cities, the whole action doesn't make sense since Bond's people may wait too long for the enemy.

You are responsible for finding all the interesting cities or saying that there is not enough of them. By not enough, James means strictly less than 20% of all cities.

Input

The first line contains one integer t ($1 \le t \le 2\,000$) — the number of test cases. Each test case is described as follows:

The first line contains two integers n and m ($1 \le n \le 10^5$, $0 \le m \le 2 \cdot 10^5$) — the number of cities and roads between them. Each of the following m lines contains two integers u, v ($u \neq v$; $1 \le u, v \le n$), which denote that there is a directed road from u to v.

You can assume that between each ordered pair of cities there is at most one road. The sum of n over all test cases doesn't exceed 10^5, and the sum of m doesn't exceed $2 \cdot 10^5$.

Output

If strictly less than 20% of all cities are interesting, print -1. Otherwise, let k be the number of interesting cities. Print k distinct integers in increasing order — the indices of interesting cities.

Example

input

```
4
3 3
1 2
2 3
3 1
3 6
1 2
2 1
2 3
3 2
1 3
3 1
7 10
1 2
2 3
3 1
1 4
4 5
5 1
4 6
6 7
7 4
6 1
6 8
1 2
2 3
3 4
4 5
5 6
6 1
6 2
```

```
5 1
```

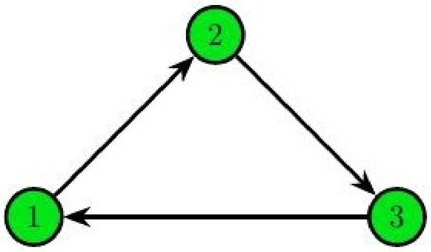
output

```
1 2 3
-1
1 2 3 5
-1
```

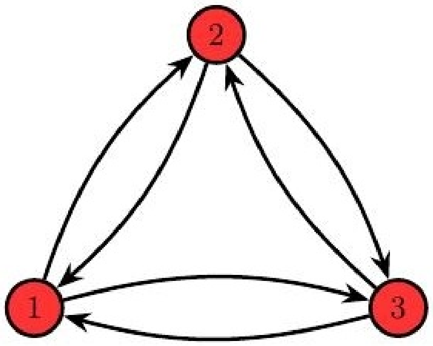
Note

In all drawings, if a city is colored green, then it is interesting; otherwise, it is colored red.

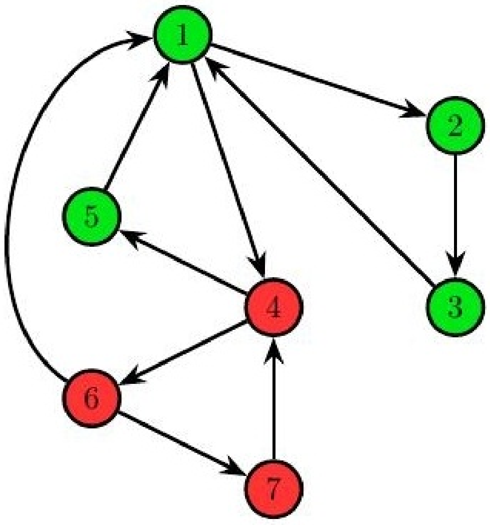
In the first sample, each city is interesting.



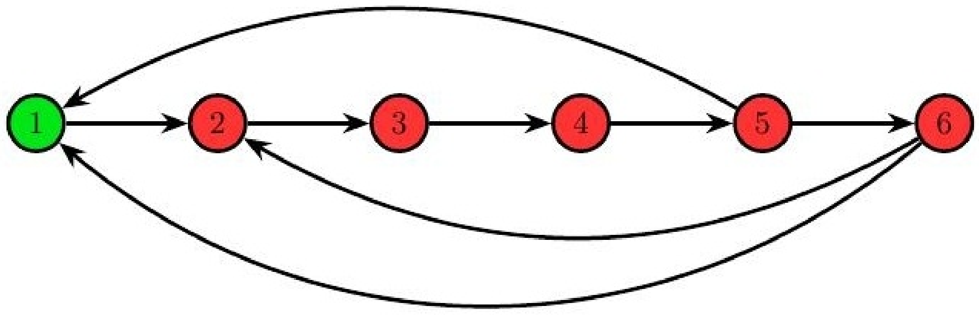
In the second sample, no city is interesting.



In the third sample, cities 1, 2, 3 and 5 are interesting.



In the last sample, only the city 1 is interesting. It is strictly less than 20% of all cities, so the answer is -1.



F. Johnny and New Toy

time limit per test: 15 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

Johnny has a new toy. As you may guess, it is a little bit extraordinary. The toy is a permutation P of numbers from 1 to n, written in one row next to each other.

For each i from 1 to $n-1$ between P_i and P_{i+1} there is a weight W_i written, and those weights form a permutation of numbers from 1 to $n-1$. There are also extra weights $W_0 = W_n = 0$.

The instruction defines subsegment $[L, R]$ as good if $W_{L-1} < W_i$ and $W_R < W_i$ for any i in $\{L, L+1, \ldots, R-1\}$. For such subsegment it also defines W_M as minimum of set $\{W_L, W_{L+1}, \ldots, W_{R-1}\}$.

Now the fun begins. In one move, the player can choose one of the good subsegments, cut it into $[L, M]$ and $[M+1, R]$ and swap the two parts. More precisely, before one move the chosen subsegment of our toy looks like:

$$W_{L-1}, P_L, W_L, \ldots, W_{M-1}, P_M, W_M, P_{M+1}, W_{M+1}, \ldots, W_{R-1}, P_R, W_R$$

and afterwards it looks like this:

$$W_{L-1}, P_{M+1}, W_{M+1}, \ldots, W_{R-1}, P_R, W_M, P_L, W_L, \ldots, W_{M-1}, P_M, W_R$$

Such a move can be performed multiple times (possibly zero), and the goal is to achieve the minimum number of inversions in P.

Johnny's younger sister Megan thinks that the rules are too complicated, so she wants to test her brother by choosing some pair of indices X and Y, and swapping P_X and P_Y (X might be equal Y). After each sister's swap, Johnny wonders, what is the minimal number of inversions that he can achieve, starting with current P and making legal moves?

You can assume that the input is generated **randomly**. P and W were chosen independently and equiprobably over all permutations; also, Megan's requests were chosen independently and equiprobably over all pairs of indices.

Input

The first line contains single integer n $(2 \leq n \leq 2 \cdot 10^5)$ denoting the length of the toy.

The second line contains n distinct integers $P_1, P_2, \ldots, P_n$ $(1 \leq P_i \leq n)$ denoting the initial permutation P. The third line contains $n-1$ distinct integers $W_1, W_2, \ldots, W_{n-1}$ $(1 \leq W_i \leq n-1)$ denoting the weights.

The fourth line contains single integer q $(1 \leq q \leq 5 \cdot 10^4)$ — the number of Megan's swaps. The following q lines contain two integers X and Y $(1 \leq X, Y \leq n)$ — the indices of elements of P to swap. The queries aren't independent; after each of them, the permutation is changed.

Output

Output q lines. The i-th line should contain exactly one integer — the minimum number of inversions in permutation, which can be obtained by starting with the P after first i queries and making moves described in the game's instruction.

Examples

input

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

output

```
0
1
0
```

input

```
5
4 3 2 5 1
3 2 1 4
7
5 4
5 2
1 5
2 4
2 4
4 3
3 3
```

output

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

Note

Consider the first sample. After the first query, P is sorted, so we already achieved a permutation with no inversions.

After the second query, P is equal to $[1, 3, 2]$, it has one inversion, it can be proven that it is impossible to achieve 0 inversions.

In the end, P is equal to $[2, 3, 1]$; we can make a move on the whole permutation, as it is a good subsegment itself, which results in P equal to $[1, 2, 3]$, which has 0 inversions.