Disconnected Network

In a small town called Sparta, there is a war happening. People need metals for the war, because they can't get metals anywhere else, they start to think if they can cut the network wires to get more. However, they don't want to disconnect and separate the town too much. Therefore, they want to know which network wire is the most important and make sure they don't cut that. There are V network nodes and E network wires to connect them. It is guaranteed to be a connected network system. We want to help them find the most important wire that disconnect the most numbers of network nodes if cut.

Input The first line will be an integer V, which is the number of network nodes The second line will be an integer E, which is the number of edges The following E lines will have two distinct integers v1 and v2 separated by an empty space. v1 and v2 will be the indexes that represents the edges. (0 <= v1 < V, 0 <= v2 < V)

Output

Three integers on two lines: first two on the first line (each integer should be followed by a single space), third integer on the second line (with no space afterwards).

The first and second value will be the indexes of the edges `v1` and `v2` (`v1 < v2`) The third will be the the minimum value of `b - a` where `b >= a` a is the number of vertices in one part of the disconnected graph, b is the number of vertices in the other part of the disconnected graph. If there is a tie, choose the one with the smaller `v1` . If the `v1` are the same, break the tie with the smaller `v2` . If there is no critical connection, print `No critical connection`

Example 1

```
Input:
4
4
0 3
1 0
1 2
3 1

Output:
1 2
2
```

Spell Check

How does a spell checker know which word we’re actually trying to spell?

One way to think about this problem is to consider the idea of "closeness." Here, we make a word "closer" to another word by one of three operations:

- inserting a gap, incurring a 1 point penalty
- matching 2 different letters, incurring a 2 point penalty
- matching 2 identical letters, incurring no penalty

For example, if we wanted to make the word "see" closer to the word "splice," we could try the following set of operations (a gap is represented as "-"):

```
splice

see--- -> penalty = 0 + 2 + 2 + 1 + 1 + 1 = 7
```

Here is a set of operations with a smaller penalty:

```
splice

s---ee -> penalty = 0 + 1 + 1 + 1 + 2 + 0 = 5
```

Write an algorithm that, given a target word and a dictionary of possible matches, finds the word in the dictionary that is "closest" to the target. The closest word is the word with the smallest penalty (using an optimal set of operations). If there are multiple closest words, choose the lexicographically smallest.

Input

The first line of the input contains the target word. The next line contains an integer `n`, where `1 <= n <= 50`. `n` lines follow, each containing one possible match.

All words are non-empty strings with lengths no greater than 50. All words are composed of only lowercase alphabetical characters.

Output

Print the closest match to the target word.

Example 1

```
Input:
splice
4
see
pie
splendid
sec

Output:
pie
```

Explanation:

One possible optimal variation of "see" to match with splice:

splice

s—ee -> penalty = 0 + 1 + 1 + 1 + 2 + 0 = 5

Other words:

splice

-p-i-e -> penalty = 1 + 0 + 1 + 0 + 1 + 0 = 3

spl—i-ce

splendid-- -> penalty = 0 + 0 + 0 + 1 + 1 + 1 + 0 + 1 + 1 + 1 = 6

splice

s–ec- -> penalty = 0 + 1 + 1 + 2 + 0 + 1 = 5

"pie" has the smallest penalty, so that is the word we are actually trying to spell.

Example 2

```
Input:
algorithm
4
lgoitm
gorim
alortm
algo

Output:
alortm
```

Ayu’s Course Registration

Ayu is tired of asking her friends to write programs to solve her problems (and her friends are tired too). So, she decided to enroll in the Computer Science program at NYU. She has asked her friends to write her a program one last time. Right now, she needs help figuring out the order she has to take all of her classes. Some classes have prerequisites, and she can only take a certain number of classes in a semester. In order to plan for her future, she wants to know the how she should enroll in her courses so that she can finish the Computer Science program.

Given all of the classes and each class’s prerequisite (or lack-thereof), your task is to find the order of classes Ayu needs to take to finish all the requirements.

It is important to note that the level of difficulty for each class is indicated by its course id number. For example, class 2 is more difficult than class 1. The Computer Science program encourages students to enroll in the easier classes first, if they have multiple options for classes to choose from in a given semester. Ayu plans to abide by this rule. For example, if Ayu is qualified to take classes 1, 4, 7, and 8 but the semester is limited to only 2 classes, she will take classes 1 and 4.

Input The first line contains two integers, S and N, 1<= S<= 5 and 1<= N<= 15, where S is the number of courses Ayu can take in one semester and N is the number of courses Ayu must take to complete the program. Each of the next N lines contains information about the i-th course, 1<= i<= N. It contains one integer P, 0<= P$\l$N, followed by P integers on the same line. Each of P integers are the course ids of the prerequisites required for this course i.

It is guaranteed that there is at least one course to take per semester. It is also guaranteed that there will be no bidirectional dependency of courses, where course i and course j depend on each other, nor courses that depend on themselves.

Output There should be a line for every semester until Ayu is done with all of her classes in the program. Each line starts with "Sem" followed by an integer i, which is the semester number (1<= i<= N), followed by a colon and a whitespace ": ". What follows are the course ids, separated by a whitespace in between, that Ayu takes in that semester i, in the order described in the problem.

Example 1

```
Input:
2 4
0
1 1
1 1
1 2


Output:
Sem1: 1
Sem2: 2 3
Sem3: 4
```

Example 2

```
Input:
2 12
0
0
1 1
1 2
1 3
1 3
2 4 5
2 4 5
2 4 5
1 9
0
1 3


Output:
```

```
Sem1: 1 2
Sem2: 3 4
Sem3: 5 6
Sem4: 7 8
Sem5: 9 11
Sem6: 10 12
```

Example 3

```
Input:
3 12
0
0
1 1
1 2
1 3
1 3
2 4 5
2 4 5
2 4 5
1 9
0
1 3


Output:
Sem1: 1 2 11
Sem2: 3 4
Sem3: 5 6 12
Sem4: 7 8 9
Sem5: 10
```

Ayu’s_Shopping_V2.0

Uya, Ayu’s mother, has asked Ayu to go grocery shopping and buy food that has the highest nutritional value. Ayu must walk one hundred and three miles to Valhalla, where it is well known that the town possesses N number of merchants. Each merchant has an independent stall that is thirteen kilometers away from the next merchant. Ayu begins her journey to Valhalla with an empty rucksack that has a total capacity of T. While walking to the gates of Valhalla, Ayu meets a wise man called Odin. Odin advises Ayu to buy at least one item from each merchant’s stall as a sign of goodwill. Ayu is conflicted because she wants to respect her mothers’ orders but also the advice of Odin. Ayu decides that she will buy at least one item from each stall and also buy the one’s that add up to the highest nutritional value possible while the total weight of the items does not exceed T.

Given N number of merchants each having M items that have a weight, nutritional value, and a rucksack of capacity T, output the maximum nutritional value of the food items that doesn’t exceed the weight capacity of the rucksack and must have at least one food item from each merchant. If there is a group that we are not able to pick any item from, print `-1` as the result.

Input

The first line will have two integers T and N, where T is the maximum capacity of our rucksack and N is the number of merchant stalls. For the next N lines, we will have the first integer M, that describes the number of items in our merchant’s stall. For the next M pairs p1 and p2, where p1 describes the weight of item1 and p2 the nutritional value of item1.

T N M w1 v1 w2 v2 …

Constraint: `1 <= T <= 10000` `1 <= N <= 10` `1 <= M <= 100` `1 <= weight of each item <= 10000` `1 <= nutritional value of each item <= 10000`

Output

Print one integer (followed by a new line character) that is the maximum value we can have given the number of items in each group. Print `-1` (followed by a new line character) if there is a group that we are not able to pick any item from. The output should end with a single blank line after the line described above.

Example 1

```
Input:
4 3
3 1 100 999 1 999 2
2 999 1 1 200
4 999 2 999 3 1 300 1 400

Output:
1000
```

Example 2

```
Input:
100 2
1 1 100
1 999 1

Output:
-1
```

Friends

You are creating a social media app. Your users can add other users as friends. You want to find out the number of friends a person has, but realize that sometimes some friends forget to add each other, and some friends tangentially know of each other. Therefore, in order to compute the number of friends a person has, you decide on a friendship score. If the number of friends between two people is below that friendship score, then two people count as friends. Find the number of friends each person has, including those not directly on their friends list.

Input

An integer f, 1 <= f <= 1,000,000, that is the friendship score, and an integer n, 1 <= n <= 100, that represents the number of friend connections on the app. The next lines will be 2 integers k, l in range 1 <= k,l <= 1,000,000. The two integers each correspond to users that have added each other as friends.

Output

Print out the user and the number of friends in numerical order each user has within their friendship score range.

Example 1

```
Input:
2
4
1 2
2 3
3 4
1 4

Output:
1 2
2 2
3 2
4 2
```

Example 2

```
Input:
2
6
1 2
2 3
3 4
4 5
3 6
6 7

Output:
1 2
2 4
3 6
4 4
5 2
6 4
7 2
```

Example 3

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

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

Interview Search

Mark is an excellent student and career star. He has applied to many companies, and all of them are looking to interview him 😨. Mark will never fail an interview, so he wants to interview as much as possible. Each company has at least one interview with a specified start time and end time. Companies also have "f" follow up interviews where 0 <= f <= 3, which you can only attend after completing the previous interview at the company. Each interview will be at least one full hour. Help Mark find the maximum time he can spend interviewing for the weekend (the next 48 hours).

Input The first line will have an integer "n" where 0 < n <= 48 will describe the number of interviews for the next two days. The next n lines will then list the company name and three integers, x, y, and z. For the two days, there are 48 hours total. Integer x will be the starting hour of the interview in the weekend, 0 <= x <= 47 . Integer y will be at what hour the interview ends, 1 <= y <= 48. Integer z will be what round the interview is. You're guaranteed to encounter the rounds in order (round 1 will always come before round 2, and so on)

Output

The maximum time he can spend completing his interviews.

Example 1

```
Input:
6
Bapple 0 1 1
Bapple 1 4 2
Besla 4 6 1
BabeStop 4 9 1
Besla 8 10 2
BabeStop 11 12 2

Output:
10
```

Example 2

```
Input:
Bucci 10 30 1

Output:
20
```

Hacker

You are a hacker who has access to the encrypted value of an RSA encrypted message between two people. Your job is to decrypt the conversations that a group of people have. If person a talks to person b and person b talks to person c, then a,b,c are all considered to be in the same group. Each person sends one character at a time, and the ASCI code of the character is being encrypted. RSA encryption scheme is given as such: choose 2 prime numbers p and q n := p*q phi := (p-1)*(q-1) Choose e such that e and phi are co-prime and e < phi. Choose d such that (d*e)%phi = 1 and d < phi Public key is (e,n) Private key is (d,n) To encrypt a message m : m^e % n = c To decrypt a message c : c^d % n (in this problem the smallest possible e and d are chosen for each encryption scheme)

The hacker knows the primes that are used in the encryption follow a specific constraint: The possible prime pairs have the form of a Sophie-Germain Prime p followed by the first Palindrome Prime q bigger than p. The next pair consists of first the Sophie-Germain prime p' bigger than q and the first Palindrome Prime q' bigger than p' and so on. The first two pairs are : (2,3) and (5,11). Sophie-Germain prime is a prime p such that 2*p+1 is also prime and a Palindrome prime is a prime that forms a palindrome. Another constraint is that the ith person uses the ith valid prime pair in their encryption scheme. Finally, a valid prime pair is one for which phi > 133 (chosen arbitrarily to make sure the primes are large enough to encrypt the alphabet).

Note: After you decrypt make sure you convert back the ASCI code to char **Input**

First line contains a digit n with how many people there are, 2<= n <= 10 The following lines are in form a b c. Where a and b are sender and receiver with 1< a,b <=n And c is the encrypted integer, 0 < c < 2147483647 Each character is converted to it's ASCI code and then encrypted, and this encrypted value is c. The original character that was encrypted is a letter from A to Z or from (_ , ? , . , !) (no comma) **Output** You want to print each group on a new line so that the groups are ordered in ascending order based on the minimum in each group. Each group should be printed sorted.

Finally for each person, if they've received data, write out what they've received by concatenating all of the characters they received in order. In the form: "Person i Received: " followed by the of character received in order concatenated. This should be printed in ascending order by person.

Example 1

```
Input:
2
1 2 4139
1 2 13604
1 2 3173
1 2 8171
1 2 584
1 2 2843
1 2 13604
1 2 8171
1 2 8698
1 2 9273
1 2 13604
1 2 3173
1 2 4540
1 2 12443
1 2 2843
1 2 7171
1 2 4540
1 2 3657
1 2 7391

Output:
[1, 2]
Person 2 Received: I_HATE_APS_HOMEWORK
```

Example 2

```
Input:
3
1 3 23964
```

```
2 2 534

Output:
[1, 3]
[2]
Person 2 Received: X
Person 3 Received: Y
```