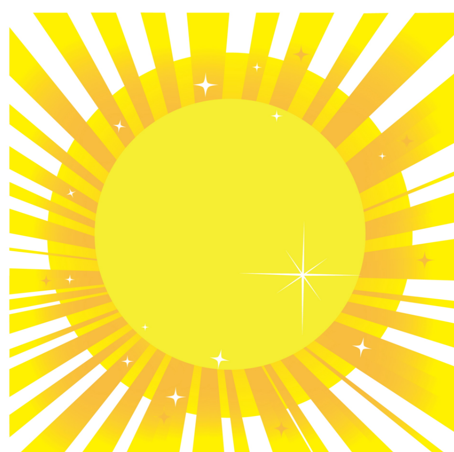
COMPUTER AND INFORMATION SCIENCE

UNIVERSITY OF OREGON

Twenty-third Annual
University of Oregon
Eugene Luks Programming Competition

2019 April 13, Saturday
am10:00 – pm2:00



Problem Contributors
Ramakrishnan Durairajan, Eugene Luks,
Joe Sventek, Chris Wilson

Technical Assistance and Organization
Lauradel Collins, Jonathan Kerr

SERVER NUMBER

The CIS Department at the University of Oregon has a fancy new data center that services computational requests from its classes. There are *n* servers in this center and they will be providing service to *c* class requests. Each class request has a different service time: the i^{th} request needs t_i units of time (here seconds) to be completed.

These requests are in a queue, and must be served in that order (1, 2, …, c). The request in the front of the queue is to be handled immediately by the first available server, and request *i* will cause the server to be busy for t_i seconds. At the start, all servers are available, and if two or more servers become available at the same time, the request goes to the lowest numbered available server.

Your job is to write a program to determine, for each of the *c* requests, the number of the server that will process it.

input

The first line of input contains two space-separated integers `N C` $(1 \leq N \leq C \leq 1000)$, where `N` is the number of servers and `C` is the number of class requests. The next line contains `C` space-separated integers, where the i^{th} integer is t_i , the time used by request *i*.

output

The output should be a single line consisting of `C` space separated integers. Each number represents the number of the server that processes that class request.

Sample Input

```
3 10
406 424 87 888 871 915 516 81 275 578
```

Sample Output

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

Wide Ways

Willamette Walk-Ways (WWW) is renowned for its integration of scenic pedestrian/bicycle paths into forested parks for new communities. Once a park region is set aside, WWW is contracted to position a promenade through the trees.

The promenade is to be a straight road (with parallel edges) subject to the following conditions

- Construction should not require removal of existing trees, i.e, no trees should obstruct the promenade (though a tree may remain on the edge).
- At least $1/3$ of the trees should be positioned to each side of the promenade.
- The promenade should be as wide as possible given the above constraints.

In this problem, you are to determine the widths achieved for WWW promenades.

For each park instance, you are given the positions of the trees to be preserved and must determine the maximum feasible width for a promenade.

The first line of the input will declare the number N of communities to be considered. The second line will be blank. There will also be a blank line between successive instances.

Each instance will start with a number $m \leq 25$ of trees within the park boundary. Each of the following m lines will indicate the Cartesian coordinates of a tree, i.e., two numbers separated by a space. Since the trees will still be saplings at this time, we assume they have negligible diameter.

The output for each instance should be the achievable path-width rounded to the nearest tenth numerically. (All I/O lengths are assumed to be given in meters, so only the numerical values need appear.)

See reverse for sample I/O.

Sample Input

```
5

2
1.5 1.5
4.5 4.5

4
1.0 1.0
6.0 1.0
7.0 1.0
8.0 1.0

5
1.0 1.0
6.0 1.0
7.0 1.0
8.0 1.0
2.0 4.0

6
1.0 1.0
6.0 1.0
7.0 1.0
8.0 1.0
2.0 4.0
4.0 4.0

7
1.0 1.0
6.0 1.0
7.0 1.0
8.0 1.0
2.0 4.0
4.0 4.0
4.0 2.0
```

Sample Output

```
4.2
1.0
4.7
3.5
2.2
```

Edges of the promenades for the Sample Output:

i. $y = -x + 3$ and $y = -x + 9$

ii. $x = 6$ and $x = 7$

iii. $y = 3x - 2$ and $y = 3x - 17$

iv. $y = x$ and $y = x - 5$

v. $y = 2x - 6$ and $y = 2x - 11$

DATACENTER CONNECTIVITY

One of the fundamental problems in cloud computing is establishing connectivity between servers across geo-distributed datacenters. In this problem, we will focus on server connectivity within a single datacenter.

Assume there are N servers in a datacenter and M server-to-server connections need to be made. Also, there are three mandatory conditions and additional constraints the connections should fulfill.

The first requirement is that any server in the datacenter should be able to communicate with any other server through the connections, possibly, through some other servers.

Attacks are possible. So the second requirement is that even if any one server from the datacenter gets disabled so that the rest of the servers are unable to communicate with it, the rest of the servers can still communicate with each other. In other words, the first requirement still holds for any subset of (N-1) servers.

The third requirement is cost: there shouldn't be any irrelevant connections in the datacenter. We will call a connection irrelevant if and only if after its' removal, the above two requirements are still held.

Constraints include

- 1 <= T <= 1000
- 1 <= M <= N * (N - 1) / 2
- 1 <= Sum of all N <= 1000
- Subtask 1: 1 <= N <= 4
- Subtask 2: 1 <= N <= 100

Given N, M, please build a network with N servers and M connections, or state that it is impossible.

input

The first line of the input contains an integer T denoting the number of test cases. The description of T test cases follow. The first and only line of each test case contains a pair of space-separated integers N and M, denoting the number of computers and the number of connections respectively.

output
Output T blocks.

If it is impossible to construct a network with the given parameters for the corresponding test case, then output -1 -1. Otherwise, output M lines, each of which contains a space-separated pair of integers denoting the IDs of the servers that should be connected. Note that multiple connections between any pair of servers and connections connecting a server to itself are (implicitly) not allowed due to the third requirement.

Sample Input

```
2
10 1
5 5
```

Sample Output

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

D

COUNT SUBSEQUENCES

A common problem is to determine whether a string S contains a string T as a **subsequence**. That is, is it possible to remove some of the characters of S and end up with T? For example, if S=ABCABC and T=AC, then the answer is "yes". Here we consider a related question: in how many distinct ways is T a subsequence of S? Using S=ABCABC and T=AC again, the answer should be 3: ABCABC, ABCABC, and ABCABC.

input

The first line of input contains an integer `C` (≤ 1000), where `C` is the number of test cases. The next `C` lines each contain 2 space-separated strings, `S T`. You can assume that `S` and `T` consist of lower case characters (`a-z`) and that $1 \leq len(T) \leq len(S) \leq 33$.

output

The output `C` integers on separate lines, each indicating the number of distinct ways in which `T` can be a subsequence of `S`.

Sample Input

```
4
abcabc ac
babgbag bag
aaaaaaaaaaaaaaaaaaaa aaaaaaaaaa
abc cba
```

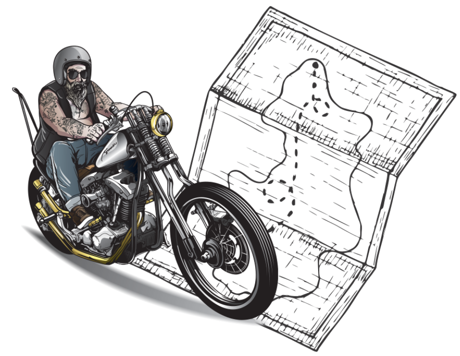
Sample Output

```
3
5
184756
0
```

Note: The output in the third case can be explained as "20 choose 10"=C(20,10)=184756.

Motorcycle Tour

E



There is a closed loop connecting gas stations, each with a single pump. You wish to tour all of the gas stations in the loop.

The stations are numbered 0 through N – 1, and since the stations form a loop, you can only travel from station i to station i + 1(*mod* N). Since it takes some amount of gasoline to get from one station to another, it is possible that you could run out of gasoline before completing the cycle.

Two pieces of information are associated with each station:

1. the amount of gasoline that particular pump will dispense (in liters); and
2. the distance from that station to the next station, in miles.

Initially you have a tank of infinite capacity with no gasoline. You can start the tour at any of the stations. Since the motorcycle has an infinite capacity tank, it consumes 1 liter of gasoline for each 10 miles covered. You need to determine the number of the first station at which you can start traveling and complete the loop.

The first line of the input will be an integer $1 \le N \le 10^9$ giving the number of stations. Following will be N lines, each containing two integers $1 \le L, D \le 10^9$ separated by a space; the integer L is the number of liters that you will receive at that station, and D is the number of miles to the next station. The output is the lowest numbered station at which you can start your journey and complete the cycle through the loop.

Sample Input

```
4
4 20
1 110
10 30
18 60
```

Sample Output

```
2
```