A. SMSC

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

Some large corporation where Polycarpus works has its own short message service center (SMSC). The center's task is to send all sorts of crucial information. Polycarpus decided to check the efficiency of the SMSC.

For that, he asked to give him the statistics of the performance of the SMSC for some period of time. In the end, Polycarpus got a list of n tasks that went to the SMSC of the corporation. Each task was described by the time it was received by the SMSC and the number of text messages to send. More formally, the i-th task was described by two integers t_i and c_i — the receiving time (the second) and the number of the text messages, correspondingly.

Polycarpus knows that the SMSC cannot send more than one text message per second. The SMSC uses a queue to organize its work. Consider a time moment x, the SMSC work at that moment as follows:

1. If at the time moment x the queue is not empty, then SMSC sends one message from the queue (SMSC gets the message from the head of the queue). Otherwise it doesn't send messages at the time moment x.
2. If at the time moment x SMSC receives a task, then it adds to the queue all the messages from this task (SMSC adds messages to the tail of the queue). Note, that the messages from the task cannot be send at time moment x. That's because the decision about sending message or not is made at point 1 before adding these messages to the queue.

Given the information about all n tasks, Polycarpus wants to count two values: the time when the last text message was sent and the maximum size of the queue at some time. Help him count these two characteristics he needs to evaluate the efficiency of the SMSC.

Input

The first line contains a single integer n $(1 \le n \le 10^3)$ — the number of tasks of the SMSC. Next n lines contain the tasks' descriptions: the i-th line contains two space-separated integers t_i and c_i $(1 \le t_i, c_i \le 10^6)$ — the time (the second) when the i-th task was received and the number of messages to send, correspondingly.

It is guaranteed that all tasks were received at different moments of time. It is guaranteed that the tasks are sorted in the chronological order, that is, $t_i < t_{i+1}$ for all integer i $(1 \le i < n)$.

Output

In a single line print two space-separated integers — the time when the last text message was sent and the maximum queue size at a certain moment of time.

Sample test(s)

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

input
```
1
1000000 10
```
output
```
1000010 10
```

input
```
3
3 3
4 3
5 3
```
output
```
12 7
```

Note

In the first test sample:

- second 1: the first message has appeared in the queue, the queue's size is 1;

- second 2: the first message is sent, the second message has been received, the queue's size is 1;
- second 3: the second message is sent, the queue's size is 0,

Thus, the maximum size of the queue is 1, the last message was sent at the second 3.

B. Network Topology

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

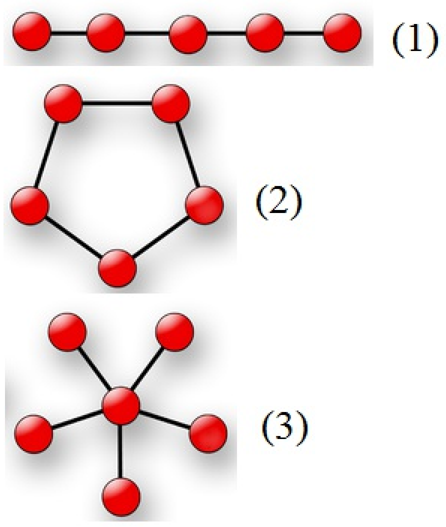
This problem uses a simplified network topology model, please read the problem statement carefully and use it as a formal document as you develop the solution.

Polycarpus continues working as a system administrator in a large corporation. The computer network of this corporation consists of n computers, some of them are connected by a cable. The computers are indexed by integers from 1 to n. It's known that any two computers connected by cable directly or through other computers

Polycarpus decided to find out the network's topology. A network topology is the way of describing the network configuration, the scheme that shows the location and the connections of network devices.

Polycarpus knows three main network topologies: bus, ring and star. A bus is the topology that represents a shared cable with all computers connected with it. In the ring topology the cable connects each computer only with two other ones. A star is the topology where all computers of a network are connected to the single central node.

Let's represent each of these network topologies as a connected non-directed graph. A bus is a connected graph that is the only path, that is, the graph where all nodes are connected with two other ones except for some two nodes that are the beginning and the end of the path. A ring is a connected graph, where all nodes are connected with two other ones. A star is a connected graph, where a single central node is singled out and connected with all other nodes. For clarifications, see the picture.



(1) — bus, (2) — ring, (3) — star

You've got a connected non-directed graph that characterizes the computer network in Polycarpus' corporation. Help him find out, which topology type the given network is. If that is impossible to do, say that the network's topology is unknown.

Input

The first line contains two space-separated integers n and m $(4 \le n \le 10^5; 3 \le m \le 10^5)$ — the number of nodes and edges in the graph, correspondingly. Next m lines contain the description of the graph's edges. The i-th line contains a space-separated pair of integers x_i, y_i $(1 \le x_i, y_i \le n)$ — the numbers of nodes that are connected by the i-the edge.

It is guaranteed that the given graph is connected. There is at most one edge between any two nodes. No edge connects a node with itself.

Output

In a single line print the network topology name of the given graph. If the answer is the bus, print "`bus topology`" (without the quotes), if the answer is the ring, print "`ring topology`" (without the quotes), if the answer is the star, print "`star topology`" (without the quotes). If no answer fits, print "`unknown topology`" (without the quotes).

Sample test(s)

input

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

output

```
bus topology
```

input

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

```
3 4
4 1
```

output

```
ring topology
```

input

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

output

```
star topology
```

input

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

output

```
unknown topology
```

C. Beautiful IP Addresses

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

The problem uses a simplified TCP/IP address model, please read the statement carefully.

An IP address is a 32-bit integer, represented as a group of four decimal 8-bit integers (without leading zeroes), separated by commas. For example, record `0.255.1.123` shows a correct IP address and records `0.256.1.123` and `0.255.1.01` do not. In the given problem an arbitrary group of four 8-bit integers is a correct IP address.

Our hero Polycarpus still works as a system administrator in some large corporation. He likes beautiful IP addresses. To check if some IP address is beautiful, he should do the following:

1. write out in a line four 8-bit numbers of the IP address, without the commas;
2. check if the resulting string is a palindrome.

Let us remind you that a palindrome is a string that reads the same from right to left and from left to right.

For example, IP addresses `12.102.20.121` and `0.3.14.130` are beautiful (as strings "`1210220121`" and "`0314130`" are palindromes), and IP addresses `1.20.20.1` and `100.4.4.1` are not.

Polycarpus wants to find all beautiful IP addresses that have the given set of digits. Each digit from the set must occur in the IP address at least once. IP address must not contain any other digits. Help him to cope with this difficult task.

Input

The first line contains a single integer n $(1 \le n \le 10)$ — the number of digits in the set. The second line contains the set of integers $a_1, a_2, ..., a_n$ $(0 \le a_i \le 9)$. It is guaranteed that all digits in the set are distinct.

Output

In the first line print a single integer k — the number of beautiful IP addresses that contain the given set of digits. In the following k lines print the IP addresses, one per line in the arbitrary order.

Sample test(s)

input

```
6
0 1 2 9 8 7
```

output

```
6
78.190.209.187
79.180.208.197
87.190.209.178
89.170.207.198
97.180.208.179
98.170.207.189
```

input

```
1
4
```

output

```
16
4.4.4.4
4.4.4.44
4.4.44.4
4.4.44.44
4.44.4.4
4.44.4.44
4.44.44.4
4.44.44.44
44.4.4.4
44.4.4.44
44.4.44.4
44.4.44.44
44.44.4.4
44.44.4.44
44.44.44.4
44.44.44.44
```

D. Connected Components

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

We already know of the large corporation where Polycarpus works as a system administrator. The computer network there consists of n computers and m cables that connect some pairs of computers. In other words, the computer network can be represented as some non-directed graph with n nodes and m edges. Let's index the computers with integers from 1 to n, let's index the cables with integers from 1 to m.

Polycarpus was given an important task — check the reliability of his company's network. For that Polycarpus decided to carry out a series of k experiments on the computer network, where the i-th experiment goes as follows:

1. Temporarily disconnect the cables with indexes from l_i to r_i, inclusive (the other cables remain connected).
2. Count the number of connected components in the graph that is defining the computer network at that moment.
3. Re-connect the disconnected cables with indexes from l_i to r_i (that is, restore the initial network).

Help Polycarpus carry out all experiments and for each print the number of connected components in the graph that defines the computer network through the given experiment. Isolated vertex should be counted as single component.

Input

The first line contains two space-separated integers n, m $(2 \le n \le 500;\ 1 \le m \le 10^4)$ — the number of computers and the number of cables, correspondingly.

The following m lines contain the cables' description. The i-th line contains space-separated pair of integers x_i, y_i $(1 \le x_i, y_i \le n;\ x_i \ne y_i)$ — the numbers of the computers that are connected by the i-th cable. Note that a pair of computers can be connected by multiple cables.

The next line contains integer k $(1 \le k \le 2 \cdot 10^4)$ — the number of experiments. Next k lines contain the experiments' descriptions. The i-th line contains space-separated integers l_i, r_i $(1 \le l_i \le r_i \le m)$ — the numbers of the cables that Polycarpus disconnects during the i-th experiment.

Output

Print k numbers, the i-th number represents the number of connected components of the graph that defines the computer network during the i-th experiment.

Sample test(s)

input

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

output

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

E. Copying Data

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

We often have to copy large volumes of information. Such operation can take up many computer resources. Therefore, in this problem you are advised to come up with a way to copy some part of a number array into another one, quickly.

More formally, you've got two arrays of integers $a_1, a_2, ..., a_n$ and $b_1, b_2, ..., b_n$ of length n. Also, you've got m queries of two types:

1. Copy the subsegment of array a of length k, starting from position x, into array b, starting from position y, that is, execute $b_{y+q} = a_{x+q}$ for all integer q $(0 \le q < k)$. The given operation is correct — both subsegments do not touch unexistent elements.
2. Determine the value in position x of array b, that is, find value b_x.

For each query of the second type print the result — the value of the corresponding element of array b.

Input

The first line contains two space-separated integers n and m $(1 \le n, m \le 10^5)$ — the number of elements in the arrays and the number of queries, correspondingly. The second line contains an array of integers $a_1, a_2, ..., a_n$ $(|a_i| \le 10^9)$. The third line contains an array of integers $b_1, b_2, ..., b_n$ $(|b_i| \le 10^9)$.

Next m lines contain the descriptions of the queries. The i-th line first contains integer t_i — the type of the i-th query $(1 \le t_i \le 2)$. If $t_i = 1$, then the i-th query means the copying operation. If $t_i = 2$, then the i-th query means taking the value in array b. If $t_i = 1$, then the query type is followed by three integers x_i, y_i, k_i $(1 \le x_i, y_i, k_i \le n)$ — the parameters of the copying query. If $t_i = 2$, then the query type is followed by integer x_i $(1 \le x_i \le n)$ — the position in array b.

All numbers in the lines are separated with single spaces. It is guaranteed that all the queries are correct, that is, the copying borders fit into the borders of arrays a and b.

Output

For each second type query print the result on a single line.

Sample test(s)

input

```
5 10
1 2 0 -1 3
3 1 5 -2 0
2 5
1 3 3 3
2 5
2 4
2 1
1 2 1 4
2 1
2 4
1 4 2 1
2 2
```

output

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