A. Parliament of Berland

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n parliamentarians in Berland. They are numbered with integers from 1 to n. It happened that all parliamentarians with odd indices are Democrats and all parliamentarians with even indices are Republicans.

New parliament assembly hall is a rectangle consisting of $a \times b$ chairs — a rows of b chairs each. Two chairs are considered neighbouring if they share as side. For example, chair number 5 in row number 2 is neighbouring to chairs number 4 and 6 in this row and chairs with number 5 in rows 1 and 3. Thus, chairs have four neighbours in general, except for the chairs on the border of the hall

We know that if two parliamentarians from one political party (that is two Democrats or two Republicans) seat nearby they spent all time discussing internal party issues.

Write the program that given the number of parliamentarians and the sizes of the hall determine if there is a way to find a seat for any parliamentarian, such that no two members of the same party share neighbouring seats.

Input

The first line of the input contains three integers n, a and b ($1 \le n \le 10\,000$, $1 \le a, b \le 100$) — the number of parliamentarians, the number of rows in the assembly hall and the number of seats in each row, respectively.

Output

If there is no way to assigns seats to parliamentarians in a proper way print -1.

Otherwise print the solution in a lines, each containing b integers. The j-th integer of the i-th line should be equal to the index of parliamentarian occupying this seat, or 0 if this seat should remain empty. If there are multiple possible solution, you may print any of them.

Examples

input
```
3 2 2
```
output
```
0 3
1 2
```

input
```
8 4 3
```
output
```
7 8 3
0 1 4
6 0 5
0 2 0
```

input
```
10 2 2
```
output
```
-1
```

Note

In the first sample there are many other possible solutions. For example,

```
3 2
0 1
```
and

```
2 1
3 0
```

The following assignment

```
3 2
1 0
```

is incorrect, because parliamentarians 1 and 3 are both from Democrats party but will occupy neighbouring seats.

B. Processing Queries

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In this problem you have to simulate the workflow of one-thread server. There are n queries to process, the i-th will be received at moment t_i and needs to be processed for d_i units of time. All t_i are guaranteed to be distinct.

When a query appears server may react in three possible ways:

1. If server is free and query queue is empty, then server immediately starts to process this query.
2. If server is busy and there are less than b queries in the queue, then new query is added to the end of the queue.
3. If server is busy and there are already b queries pending in the queue, then new query is just rejected and will never be processed.

As soon as server finished to process some query, it picks new one from the queue (if it's not empty, of course). If a new query comes at some moment x, and the server finishes to process another query at exactly the same moment, we consider that first query is picked from the queue and only then new query appears.

For each query find the moment when the server will finish to process it or print -1 if this query will be rejected.

Input

The first line of the input contains two integers n and b ($1 \le n, b \le 200\,000$) — the number of queries and the maximum possible size of the query queue.

Then follow n lines with queries descriptions (in chronological order). Each description consists of two integers t_i and d_i ($1 \le t_i, d_i \le 10^9$), where t_i is the moment of time when the i-th query appears and d_i is the time server needs to process it. It is guaranteed that $t_{i-1} < t_i$ for all $i > 1$.

Output

Print the sequence of n integers $e_1, e_2, \ldots, e_n$, where e_i is the moment the server will finish to process the i-th query (queries are numbered in the order they appear in the input) or -1 if the corresponding query will be rejected.

Examples

input

```
5 1
2 9
4 8
10 9
15 2
19 1
```

output

```
11 19 -1 21 22
```

input

```
4 1
2 8
4 8
10 9
15 2
```

output

```
10 18 27 -1
```

Note

Consider the first sample.

1. The server will start to process first query at the moment 2 and will finish to process it at the moment 11.
2. At the moment 4 second query appears and proceeds to the queue.
3. At the moment 10 third query appears. However, the server is still busy with query 1, $b = 1$ and there is already query 2 pending in the queue, so third query is just rejected.
4. At the moment 11 server will finish to process first query and will take the second query from the queue.
5. At the moment 15 fourth query appears. As the server is currently busy it proceeds to the queue.
6. At the moment 19 two events occur simultaneously: server finishes to proceed the second query and the fifth query appears. As was said in the statement above, first server will finish to process the second query, then it will pick the fourth query from the queue and only then will the fifth query appear. As the queue is empty fifth query is proceed there.
7. Server finishes to process query number 4 at the moment 21. Query number 5 is picked from the queue.

8. Server finishes to process query number 5 at the moment 22.

C. Hostname Aliases

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are some websites that are accessible through several different addresses. For example, for a long time Codeforces was accessible with two hostnames `codeforces.com` and `codeforces.ru`.

You are given a list of page addresses being queried. For simplicity we consider all addresses to have the form `http://<hostname>[/<path>]`, where:

- `<hostname>` — server name (consists of words and maybe some dots separating them),
- `/<path>` — optional part, where `<path>` consists of words separated by slashes.

We consider two `<hostname>` to correspond to one website if for each query to the first `<hostname>` there will be exactly the same query to the second one and vice versa — for each query to the second `<hostname>` there will be the same query to the first one. Take a look at the samples for further clarifications.

Your goal is to determine the groups of server names that correspond to one website. Ignore groups consisting of the only server name.

Please note, that according to the above definition queries `http://<hostname>` and `http://<hostname>/` are different.

Input

The first line of the input contains a single integer n ($1 \le n \le 100\,000$) — the number of page queries. Then follow n lines each containing exactly one address. Each address is of the form `http://<hostname>[/<path>]`, where:

- `<hostname>` consists of lowercase English letters and dots, there are no two consecutive dots, `<hostname>` doesn't start or finish with a dot. The length of `<hostname>` is positive and doesn't exceed 20.
- `<path>` consists of lowercase English letters, dots and slashes. There are no two consecutive slashes, `<path>` doesn't start with a slash and its length doesn't exceed 20.

Addresses are not guaranteed to be distinct.

Output

First print k — the number of groups of server names that correspond to one website. You should count only groups of size greater than one.

Next k lines should contain the description of groups, one group per line. For each group print all server names separated by a single space. You are allowed to print both groups and names inside any group in arbitrary order.

Examples

input

```
10
http://abacaba.ru/test
http://abacaba.ru/
http://abacaba.com
http://abacaba.com/test
http://abacaba.de/
http://abacaba.ru/test
http://abacaba.de/test
http://abacaba.com/
http://abacaba.com/t
http://abacaba.com/test
```

output

```
1
http://abacaba.de http://abacaba.ru
```

input

```
14
http://c
http://ccc.bbbb/aba..b
http://cba.com
http://a.c/aba..b/a
http://abc/
http://a.c/
http://ccc.bbbb
http://ab.ac.bc.aa/
http://a.a.a/
http://ccc.bbbb/
http://cba.com/
http://cba.com/aba..b
http://a.a.a/aba..b/a
```

```
http://abc/aba..b/a
```

output

```
2
http://cba.com http://ccc.bbbb
http://a.a.a http://a.c http://abc
```