CODEFORCES β
Sponsored by Telegram

Codeforces Round #387 (Div. 2)

A. Display Size

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

A big company decided to launch a new series of rectangular displays, and decided that the display must have exactly n pixels.

Your task is to determine the size of the rectangular display — the number of lines (rows) of pixels a and the number of columns of pixels b, so that:

- there are exactly n pixels on the display;
- the number of rows does not exceed the number of columns, it means $a \le b$;
- the difference $b - a$ is as small as possible.

Input

The first line contains the positive integer n $(1 \le n \le 10^6)$ — the number of pixels display should have.

Output

Print two integers — the number of rows and columns on the display.

Examples

input

```
8
```

output

```
2 4
```

input

```
64
```

output

```
8 8
```

input

```
5
```

output

```
1 5
```

input

```
999999
```

output

```
999 1001
```

Note

In the first example the minimum possible difference equals 2, so on the display should be 2 rows of 4 pixels.

In the second example the minimum possible difference equals 0, so on the display should be 8 rows of 8 pixels.

In the third example the minimum possible difference equals 4, so on the display should be 1 row of 5 pixels.

B. Mammoth's Genome Decoding

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

The process of mammoth's genome decoding in Berland comes to its end!

One of the few remaining tasks is to restore unrecognized nucleotides in a found chain s. Each nucleotide is coded with a capital letter of English alphabet: '`A`', '`C`', '`G`' or '`T`'. Unrecognized nucleotides are coded by a question mark '`?`'. Thus, s is a string consisting of letters '`A`', '`C`', '`G`', '`T`' and characters '`?`'.

It is known that the number of nucleotides of each of the four types in the decoded genome of mammoth in Berland should be equal.

Your task is to decode the genome and replace each unrecognized nucleotide with one of the four types so that the number of nucleotides of each of the four types becomes equal.

Input

The first line contains the integer n $(4 \leq n \leq 255)$ — the length of the genome.

The second line contains the string s of length n — the coded genome. It consists of characters '`A`', '`C`', '`G`', '`T`' and '`?`'.

Output

If it is possible to decode the genome, print it. If there are multiple answer, print any of them. If it is not possible, print three equals signs in a row: "`===`" (without quotes).

Examples

input

```
8
AG?C??CT
```

output

```
AGACGTCT
```

input

```
4
AGCT
```

output

```
AGCT
```

input

```
6
????G?
```

output

```
===
```

input

```
4
AA??
```

output

```
===
```

Note

In the first example you can replace the first question mark with the letter '`A`', the second question mark with the letter '`G`', the third question mark with the letter '`T`', then each nucleotide in the genome would be presented twice.

In the second example the genome is already decoded correctly and each nucleotide is exactly once in it.

In the third and the fourth examples it is impossible to decode the genom.

C. Servers

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

There are n servers in a laboratory, each of them can perform tasks. Each server has a unique id — integer from 1 to n.

It is known that during the day q tasks will come, the i-th of them is characterized with three integers: t_i — the moment in seconds in which the task will come, k_i — the number of servers needed to perform it, and d_i — the time needed to perform this task in seconds. All t_i are distinct.

To perform the i-th task you need k_i servers which are unoccupied in the second t_i. After the servers begin to perform the task, each of them will be busy over the next d_i seconds. Thus, they will be busy in seconds $t_i, t_i + 1, ..., t_i + d_i - 1$. For performing the task, k_i servers with the smallest ids will be chosen from all the unoccupied servers. If in the second t_i there are not enough unoccupied servers, the task is ignored.

Write the program that determines which tasks will be performed and which will be ignored.

Input

The first line contains two positive integers n and q $(1 \le n \le 100, 1 \le q \le 10^5)$ — the number of servers and the number of tasks.

Next q lines contains three integers each, the i-th line contains integers t_i, k_i and d_i $(1 \le t_i \le 10^6, 1 \le k_i \le n, 1 \le d_i \le 1000)$ — the moment in seconds in which the i-th task will come, the number of servers needed to perform it, and the time needed to perform this task in seconds. The tasks are given in a chronological order and they will come in distinct seconds.

Output

Print q lines. If the i-th task will be performed by the servers, print in the i-th line the sum of servers' ids on which this task will be performed. Otherwise, print `-1`.

Examples

input

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

output

```
6
-1
10
```

input

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

output

```
3
3
```

input

```
8 6
1 3 20
4 2 1
6 5 5
10 1 1
15 3 6
21 8 8
```

output

```
6
9
30
-1
15
36
```

Note

In the first example in the second 1 the first task will come, it will be performed on the servers with ids 1, 2 and 3 (the sum of the ids equals 6) during two seconds. In the second 2 the second task will come, it will be ignored, because only the server 4 will be unoccupied at that second. In the second 3 the third task will come. By this time, servers with the ids 1, 2 and 3 will be unoccupied again, so the third task will be done on all the servers with the ids 1, 2, 3 and 4 (the sum of the ids is 10).

In the second example in the second 3 the first task will come, it will be performed on the servers with ids 1 and 2 (the sum of the ids is 3) during three seconds. In the second 5 the second task will come, it will be performed on the server 3, because the first two servers will be busy performing the first task.

D. Winter Is Coming

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

The winter in Berland lasts n days. For each day we know the forecast for the average air temperature that day.

Vasya has a new set of winter tires which allows him to drive safely no more than k days at any average air temperature. After k days of using it (regardless of the temperature of these days) the set of winter tires wears down and cannot be used more. It is not necessary that these k days form a continuous segment of days.

Before the first winter day Vasya still uses **summer tires**. It is possible to drive safely on summer tires any number of days when the average air temperature is **non-negative**. It is impossible to drive on summer tires at days when the average air temperature is negative.

Vasya can change summer tires to winter tires and vice versa at the beginning of any day.

Find the minimum number of times Vasya needs to change summer tires to winter tires and vice versa to drive safely during the winter. At the end of the winter the car can be with any set of tires.

Input

The first line contains two positive integers n and k $(1 \le n \le 2 \cdot 10^5,\ 0 \le k \le n)$ — the number of winter days and the number of days winter tires can be used. It is allowed to drive on winter tires at any temperature, but no more than k days in total.

The second line contains a sequence of n integers $t_1, t_2, ..., t_n$ $(-20 \le t_i \le 20)$ — the average air temperature in the i-th winter day.

Output

Print the minimum number of times Vasya has to change summer tires to winter tires and vice versa to drive safely during all winter. If it is impossible, print `-1`.

Examples

input
```
4 3
-5 20 -3 0
```
output
```
2
```

input
```
4 2
-5 20 -3 0
```
output
```
4
```

input
```
10 6
2 -5 1 3 0 0 -4 -3 1 0
```
output
```
3
```

Note

In the first example before the first winter day Vasya should change summer tires to winter tires, use it for three days, and then change winter tires to summer tires because he can drive safely with the winter tires for just three days. Thus, the total number of tires' changes equals two.

In the second example before the first winter day Vasya should change summer tires to winter tires, and then after the first winter day change winter tires to summer tires. After the second day it is necessary to change summer tires to winter tires again, and after the third day it is necessary to change winter tires to summer tires. Thus, the total number of tires' changes equals four.

E. Comments

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

A rare article in the Internet is posted without a possibility to comment it. On a Polycarp's website each article has comments feed.

Each comment on Polycarp's website is a non-empty string consisting of uppercase and lowercase letters of English alphabet. Comments have tree-like structure, that means each comment except root comments (comments of the highest level) has exactly one parent comment.

When Polycarp wants to save comments to his hard drive he uses the following format. Each comment he writes in the following format:

- at first, the text of the comment is written;
- after that the number of comments is written, for which this comment is a parent comment (i. e. the number of the replies to this comments);
- after that the comments for which this comment is a parent comment are written (the writing of these comments uses the same algorithm).

All elements in this format are separated by single comma. Similarly, the comments of the first level are separated by comma.
For example, if the comments look like:

then the first comment is written as "`hello,2,ok,0,bye,0`", the second is written as "`test,0`", the third comment is written as "`one,1,two,2,a,0,b,0`". The whole comments feed is written as: "`hello,2,ok,0,bye,0,test,0,one,1,two,2,a,0,b,0`". For a given comments feed in the format specified above print the comments in a different format:

- at first, print a integer d — the maximum depth of nesting comments;
- after that print d lines, the i-th of them corresponds to nesting level i;
- for the i-th row print comments of nesting level i in the order of their appearance in the Policarp's comments feed, separated by space.

Input

The first line contains non-empty comments feed in the described format. It consists of uppercase and lowercase letters of English alphabet, digits and commas.

It is guaranteed that each comment is a non-empty string consisting of uppercase and lowercase English characters. Each of the number of comments is integer (consisting of at least one digit), and either equals 0 or does not contain leading zeros.

The length of the whole string does not exceed 10^6. It is guaranteed that given structure of comments is valid.

Output

Print comments in a format that is given in the statement. For each level of nesting, comments should be printed in the order they are given in the input.

Examples

input

```
hello,2,ok,0,bye,0,test,0,one,1,two,2,a,0,b,0
```

output

```
3
hello test one
ok bye two
a b
```

input

```
a,5,A,0,a,0,A,0,a,0,A,0
```

output

```
2
a
A a A a A
```

input

```
A,3,B,2,C,0,D,1,E,0,F,1,G,0,H,1,I,1,J,0,K,1,L,0,M,2,N,0,O,1,P,0
```

output

```
4
A K M
B F H L N O
C D G I P
E J
```

Note

The first example is explained in the statements.

F. Igor and Interesting Numbers

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

Igor likes hexadecimal notation and considers **positive** integer in the hexadecimal notation *interesting* if each digit and each letter in it appears no more than t times. For example, if $t = 3$, then integers `13a13322`, `aaa`, `abcdef0123456789` are interesting, but numbers `aaaa`, `abababab` and `1000000` are not interesting.

Your task is to find the k-th smallest *interesting* for Igor integer in the hexadecimal notation. The integer should not contain leading zeros.

Input

The first line contains the two integers k and t ($1 \le k \le 2 \cdot 10^9$, $1 \le t \le 10$) — the number of the required integer and the maximum number of times some integer or letter can appear in interesting integer.

It can be shown that the answer always exists for such constraints.

Output

Print in the hexadecimal notation the only integer that is the k-th smallest *interesting* integer for Igor.

Examples

input

```
17 1
```

output

```
12
```

input

```
1000000 2
```

output

```
fca2c
```

Note

The first 20 *interesting* integers if $t = 1$: `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, `a`, `b`, `c`, `d`, `e`, `f`, `10`, `12`, `13`, `14`, `15`. So the answer for the first example equals `12`.