A. Magical Sticks

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

A penguin Rocher has n sticks. He has exactly one stick with length i for all $1 \le i \le n$.

He can connect some sticks. If he connects two sticks that have lengths a and b, he gets one stick with length $a + b$. Two sticks, that were used in the operation disappear from his set and the new connected stick appears in his set and can be used for the next connections.

He wants to create the maximum number of sticks that have the same length. It is not necessary to make all sticks have the same length, some sticks can have the other length. How many sticks with the equal length he can create?

Input

The input consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. Next t lines contain descriptions of test cases.

For each test case, the only line contains a single integer n ($1 \le n \le 10^9$).

Output

For each test case, print a single integer — the answer to the problem.

Example

input

```
4
1
2
3
4
```

output

```
1
1
2
2
```

Note

In the third case, he can connect two sticks with lengths 1 and 2 and he will get one stick with length 3. So, he will have two sticks with lengths 3.

In the fourth case, he can connect two sticks with lengths 1 and 3 and he will get one stick with length 4. After that, he will have three sticks with lengths $\{2, 4, 4\}$, so two sticks have the same length, and one stick has the other length.

B. Magical Calendar

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

A competitive eater, Alice is scheduling some practices for an eating contest on a magical calendar. The calendar is unusual because a week contains not necessarily 7 days!

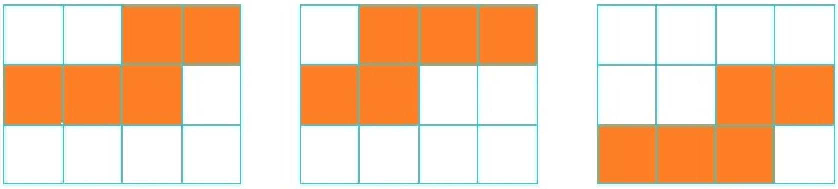
In detail, she can choose any integer k which satisfies $1 \le k \le r$, and set k days as the number of days in a week.

Alice is going to paint some n consecutive days on this calendar. On this calendar, dates are written from the left cell to the right cell in a week. If a date reaches the last day of a week, the next day's cell is the leftmost cell in the next (under) row.

She wants to make **all of the painted cells to be connected by side**. It means, that for any two painted cells there should exist at least one sequence of painted cells, started in one of these cells, and ended in another, such that any two consecutive cells in this sequence are connected by side.

Alice is considering the shape of the painted cells. Two shapes are the same if there exists a way to make them exactly overlapped **using only parallel moves, parallel to the calendar's sides**.

For example, in the picture, a week has 4 days and Alice paints 5 consecutive days. [1] and [2] are different shapes, but [1] and [3] are equal shapes.



[1] [2] [3]

Alice wants to know **how many possible shapes** exists **if she set how many days a week has and choose consecutive n days and paints them in calendar started in one of the days of the week**. As was said before, she considers only shapes, there all cells are connected by side.

Input

The input consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. Next t lines contain descriptions of test cases.

For each test case, the only line contains two integers n, r ($1 \le n \le 10^9, 1 \le r \le 10^9$).

Output

For each test case, print a single integer — the answer to the problem.

Please note, that the answer for some test cases won't fit into 32-bit integer type, so you should use at least 64-bit integer type in your programming language.

Example

input

```
5
3 4
3 2
3 1
13 7
1010000 9999999
```

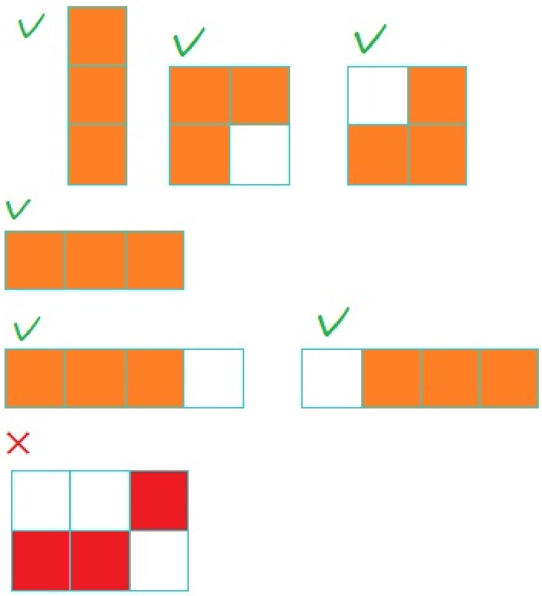
output

```
4
3
1
28
510049495001
```

Note

In the first test case, Alice can set $1, 2, 3$ or 4 days as the number of days in a week.

There are 6 possible paintings shown in the picture, but there are only 4 different shapes. So, the answer is 4. Notice that **the last example in the picture is an invalid painting** because all cells are not connected by sides.



In the last test case, be careful with the overflow issue, described in the output format.

C. A Cookie for You

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

output: standard output

Anna is a girl so brave that she is loved by everyone in the city and citizens love her cookies. She is planning to hold a party with cookies. Now she has a vanilla cookies and b chocolate cookies for the party.

She invited n guests of the first type and m guests of the second type to the party. They will come to the party in some order. After coming to the party, each guest will choose the type of cookie (vanilla or chocolate) to eat. There is a difference in the way how they choose that type:

If there are v vanilla cookies and c chocolate cookies at the moment, when the guest comes, then

- if the guest of the first type: if $v > c$ the guest selects a **vanilla** cookie. Otherwise, the guest selects a **chocolate** cookie.
- if the guest of the second type: if $v > c$ the guest selects a **chocolate** cookie. Otherwise, the guest selects a **vanilla** cookie.

After that:

- If there is at least one cookie of the selected type, the guest eats one.
- Otherwise (there are no cookies of the selected type), the guest gets angry and returns to home.

Anna wants to know if there exists some order of guests, such that **no one guest gets angry**. Your task is to answer her question.

Input

The input consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. Next t lines contain descriptions of test cases.

For each test case, the only line contains four integers a, b, n, m ($0 \le a, b, n, m \le 10^{18}, n + m \neq 0$).

Output

For each test case, print the answer in one line. If there exists at least one valid order, print "Yes". Otherwise, print "No".

You can print each letter in any case (upper or lower).

Example

input

```
6
2 2 1 2
0 100 0 1
12 13 25 1
27 83 14 25
0 0 1 0
1000000000000000000 1000000000000000000 1000000000000000000 1000000000000000000
```

output

```
Yes
No
No
Yes
No
Yes
```

Note

In the first test case, let's consider the order $\{1, 2, 2\}$ of types of guests. Then:

- The first guest eats a chocolate cookie. After that, there are 2 vanilla cookies and 1 chocolate cookie.
- The second guest eats a chocolate cookie. After that, there are 2 vanilla cookies and 0 chocolate cookies.
- The last guest selects a chocolate cookie, but there are no chocolate cookies. So, the guest gets angry.

So, this order can't be chosen by Anna.

Let's consider the order $\{2, 2, 1\}$ of types of guests. Then:

- The first guest eats a vanilla cookie. After that, there is 1 vanilla cookie and 2 chocolate cookies.
- The second guest eats a vanilla cookie. After that, there are 0 vanilla cookies and 2 chocolate cookies.
- The last guest eats a chocolate cookie. After that, there are 0 vanilla cookies and 1 chocolate cookie.

So, the answer to this test case is "Yes".

In the fifth test case, it is illustrated, that the number of cookies ($a + b$) can be equal to zero, but the number of guests ($n + m$) can't be equal to zero.

In the sixth test case, be careful about the overflow of 32-bit integer type.

D. Grid-00100

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

A mad scientist Dr.Jubal has made a competitive programming task. Try to solve it!

You are given integers n, k. Construct a grid A with size $n \times n$ **consisting of integers 0 and 1**. The very important condition should be satisfied: the sum of all elements in the grid is exactly k. In other words, the number of 1 in the grid is equal to k.

Let's define:

- $A_{i,j}$ as the integer in the i-th row and the j-th column.
- $R_i = A_{i,1} + A_{i,2} + \ldots + A_{i,n}$ (for all $1 \le i \le n$).
- $C_j = A_{1,j} + A_{2,j} + \ldots + A_{n,j}$ (for all $1 \le j \le n$).
- In other words, R_i are row sums and C_j are column sums of the grid A.
- For the grid A let's define the value $f(A) = (\max(R) - \min(R))^2 + (\max(C) - \min(C))^2$ (here for an integer sequence X we define $\max(X)$ as the maximum value in X and $\min(X)$ as the minimum value in X).

Find any grid A, which satisfies the following condition. Among such grids find any, for which the value $f(A)$ is the minimum possible. Among such tables, you can find any.

Input

The input consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases. Next t lines contain descriptions of test cases.

For each test case the only line contains two integers n, k $(1 \le n \le 300, 0 \le k \le n^2)$.

It is guaranteed that the sum of n^2 for all test cases does not exceed 10^5.

Output

For each test case, firstly print the minimum possible value of $f(A)$ among all tables, for which the condition is satisfied.

After that, print n lines contain n characters each. The j-th character in the i-th line should be equal to $A_{i,j}$.

If there are multiple answers you can print any.

Example

input

```
4
2 2
3 8
1 0
4 16
```

output

```
0
10
01
2
111
111
101
0
0
0
1111
1111
1111
1111
```

Note

In the first test case, the sum of all elements in the grid is equal to 2, so the condition is satisfied. $R_1 = 1, R_2 = 1$ and $C_1 = 1, C_2 = 1$. Then, $f(A) = (1-1)^2 + (1-1)^2 = 0$, which is the minimum possible value of $f(A)$.

In the second test case, the sum of all elements in the grid is equal to 8, so the condition is satisfied. $R_1 = 3, R_2 = 3, R_3 = 2$ and $C_1 = 3, C_2 = 2, C_3 = 3$. Then, $f(A) = (3-2)^2 + (3-2)^2 = 2$. It can be proven, that it is the minimum possible value of $f(A)$.

E1. Asterism (Easy Version)

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

This is the easy version of the problem. The difference between versions is the constraints on n and a_i. You can make hacks only if all versions of the problem are solved.

First, Aoi came up with the following idea for the competitive programming problem:

Yuzu is a girl who collecting candies. Originally, she has x candies. There are also n enemies numbered with integers from 1 to n. Enemy i has a_i candies.

Yuzu is going to determine a permutation P. A permutation is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $\{2,3,1,5,4\}$ is a permutation, but $\{1,2,2\}$ is not a permutation (2 appears twice in the array) and $\{1,3,4\}$ is also not a permutation (because $n=3$ but there is the number 4 in the array).

After that, she will do n duels with the enemies with the following rules:

- If Yuzu has **equal or more** number of candies than enemy P_i, she wins the duel and **gets** 1 **candy**. Otherwise, she loses the duel and gets nothing.
- The candy which Yuzu gets will be used in the next duels.

Yuzu wants to **win all duels**. How many valid permutations P exist?

This problem was easy and wasn't interesting for Akari, who is a friend of Aoi. And Akari made the following problem from the above idea:

Let's define $f(x)$ as the number of valid permutations for the integer x.

You are given n, a and **a prime number** $p \le n$. Let's call a positive integer x **good**, if the value $f(x)$ is **not** divisible by p. Find **all** good integers x.

Your task is to solve this problem made by Akari.

Input

The first line contains two integers n, p $(2 \le p \le n \le 2000)$. It is guaranteed, that the number p is prime (it has exactly two divisors 1 and p).

The second line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 2000)$.

Output

In the first line, print the number of good integers x.

In the second line, output all good integers x **in the ascending order**.

It is guaranteed that the number of good integers x does not exceed 10^5.

Examples

input

```
3 2
3 4 5
```

output

```
1
3
```

input

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

output

```
2
3 4
```

input

```
4 3
9 1 1 1
```

output

```
0
```

Note

In the first test, $p=2$.

- If $x \le 2$, there are no valid permutations for Yuzu. So $f(x)=0$ for all $x \le 2$. The number 0 is divisible by 2, so all integers $x \le 2$ are not good.
- If $x=3$, $\{1,2,3\}$ is the only valid permutation for Yuzu. So $f(3)=1$, so the number 3 is good.
- If $x=4$, $\{1,2,3\}$, $\{1,3,2\}$, $\{2,1,3\}$, $\{2,3,1\}$ are all valid permutations for Yuzu. So $f(4)=4$, so the number 4 is not good.
- If $x \ge 5$, all 6 permutations are valid for Yuzu. So $f(x)=6$ for all $x \ge 5$, so all integers $x \ge 5$ are not good.

So, the only good number is 3.

In the third test, for all positive integers x the value $f(x)$ is divisible by $p=3$.

E2. Asterism (Hard Version)

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

This is the hard version of the problem. The difference between versions is the constraints on n and a_i. You can make hacks only if all versions of the problem are solved.

First, Aoi came up with the following idea for the competitive programming problem:

Yuzu is a girl who collecting candies. Originally, she has x candies. There are also n enemies numbered with integers from 1 to n. Enemy i has a_i candies.

Yuzu is going to determine a permutation P. A permutation is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $\{2,3,1,5,4\}$ is a permutation, but $\{1,2,2\}$ is not a permutation (2 appears twice in the array) and $\{1,3,4\}$ is also not a permutation (because $n=3$ but there is the number 4 in the array).

After that, she will do n duels with the enemies with the following rules:

- If Yuzu has **equal or more** number of candies than enemy P_i, she wins the duel and **gets 1 candy**. Otherwise, she loses the duel and gets nothing.
- The candy which Yuzu gets will be used in the next duels.

Yuzu wants to **win all duels**. How many valid permutations P exist?

This problem was easy and wasn't interesting for Akari, who is a friend of Aoi. And Akari made the following problem from the above idea:

Let's define $f(x)$ as the number of valid permutations for the integer x.

You are given n, a and **a prime number** $p \le n$. Let's call a positive integer x **good**, if the value $f(x)$ is **not** divisible by p. Find **all** good integers x.

Your task is to solve this problem made by Akari.

Input

The first line contains two integers n, p $(2 \le p \le n \le 10^5)$. It is guaranteed, that the number p is prime (it has exactly two divisors 1 and p).

The second line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^9)$.

Output

In the first line, print the number of good integers x.

In the second line, output all good integers x **in the ascending order**.

It is guaranteed that the number of good integers x does not exceed 10^5.

Examples

input
```
3 2
3 4 5
```
output
```
1
3
```

input
```
4 3
2 3 5 6
```
output
```
2
3 4
```

input
```
4 3
9 1 1 1
```
output
```
0
```

input
```
3 2
1000000000 1 999999999
```
output
```
1
999999998
```

Note

In the first test, $p = 2$.

- If $x \le 2$, there are no valid permutations for Yuzu. So $f(x) = 0$ for all $x \le 2$. The number 0 is divisible by 2, so all integers $x \le 2$ are not good.
- If $x = 3$, $\{1, 2, 3\}$ is the only valid permutation for Yuzu. So $f(3) = 1$, so the number 3 is good.
- If $x = 4$, $\{1, 2, 3\}, \{1, 3, 2\}, \{2, 1, 3\}, \{2, 3, 1\}$ are all valid permutations for Yuzu. So $f(4) = 4$, so the number 4 is not good.
- If $x \ge 5$, all 6 permutations are valid for Yuzu. So $f(x) = 6$ for all $x \ge 5$, so all integers $x \ge 5$ are not good.

So, the only good number is 3.

In the third test, for all positive integers x the value $f(x)$ is divisible by $p = 3$.

F. Raging Thunder

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are a warrior fighting against the machine god Thor.

Thor challenge you to solve the following problem:

There are n conveyors arranged in a line numbered with integers from 1 to n from left to right. Each conveyor has a symbol "<" or ">". The initial state of the conveyor i is equal to the i-th character of the string s. There are $n+1$ holes numbered with integers from 0 to n. The hole 0 is on the left side of the conveyor 1, and for all $i \ge 1$ the hole i is on the right side of the conveyor i.

When a ball is on the conveyor i, the ball moves by the next rules:

If the symbol "<" is on the conveyor i, then:

- If $i = 1$, the ball falls into the hole 0.
- If the symbol "<" is on the conveyor $i-1$, the ball moves to the conveyor $i-1$.
- If the symbol ">" is on the conveyor $i-1$, the ball falls into the hole $i-1$.

If the symbol ">" is on the conveyor i, then:

- If $i = n$, the ball falls into the hole n.
- If the symbol ">" is on the conveyor $i+1$, the ball moves to the conveyor $i+1$.
- If the symbol "<" is on the conveyor $i+1$, the ball falls into the hole i.

You should answer next q queries, each query is defined by the pair of integers l, r ($1 \le l \le r \le n$):

- First, for all conveyors $l, l+1, \ldots, r$, the symbol "<" changes to ">" and vice versa. **These changes remain for the next queries.**
- After that, put **one ball** on each conveyor $l, l+1, \ldots, r$. Then, each ball falls into some hole. Find the maximum number of balls in one hole. **After the query all balls disappear and don't considered in the next queries.**

Input

The first line contains two integers n, q ($1 \le n \le 5 \times 10^5, 1 \le q \le 10^5$).

The second line contains a string s of length n. It consists of characters "<" and ">".

Next q lines contain the descriptions of the queries, i-th of them contains two integers l, r $(1 \le l \le r \le n)$, describing the i-th query.

Output

Print q lines, in the i-th of them print the answer to the i-th query.

Example

input

```
5 6
><>><
2 4
3 5
1 5
1 3
2 4
1 5
```

output

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

Note

- In the first query, the conveyors change to ">><<<". After that, put a ball on each conveyor $\{2, 3, 4\}$. All three balls fall into the hole 2. So the answer is 3.
- In the second query, the conveyors change to ">>>>>". After that, put a ball on each conveyor $\{3, 4, 5\}$. All three balls fall into the hole 5. So the answer is 3.
- In the third query, the conveyors change to "<<<<<". After that, put a ball on each conveyor $\{1, 2, 3, 4, 5\}$. All five balls fall into the hole 0. So the answer is 5.
- In the fourth query, the conveyors change to ">>><<". After that, put a ball on each conveyor $\{1, 2, 3\}$. All three balls fall into the hole 3. So the answer is 3.
- In the fifth query, the conveyors change to "><<><". After that, put a ball on each conveyor $\{2, 3, 4\}$. Two balls fall into the hole 1, and one ball falls into the hole 4. So, the answer is 2.
- In the sixth query, the conveyors change to "<>><>". After that, put a ball on each conveyor $\{1, 2, 3, 4, 5\}$. Three balls fall into the hole 3, one ball falls into the hole 0 and one ball falls into the hole 5. So, the answer is 3.
