Problem A. Maximum bitwise OR

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You have an array A with N integers $A[1], A[2], ..A[N]$. You are given Q queries. Each query consists of two integers L and R. Consider a new array B of length $R - L + 1$, such that $B[i] = A[L + i - 1]$ for all $1 \leq i \leq R - L + 1$. In one move, you can do the following in order:

1. Choose an index j such that $1 \leq j \leq R - L + 1$
2. Choose an integer i satisfying $2^i \leq B[j]$
3. Replace $B[j]$ with $B[j] \oplus (B[j] - 2^i)$, where $\oplus$ denotes the bitwise-xor operator.

The answer for the query is the maximum possible bitwise OR of all values in B, and the minimum number of moves required to obtain this value.

Input

The first line contains T, the number of testcases. Then the testcases follow.

The first line of each testcase contains two integers, N and Q.

The second line contains N space separated integers $A[1], A[2], ..A[N]$.

Each of the next Q lines contains two space separated integers L and R.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq N \leq 10^5$
- $1 \leq Q \leq 10^5$
- $0 \leq A[i] \leq 10^9$
- $1 \leq L \leq R \leq N$
- The sum of N over all testcases doesn't exceed 10^5
- The sum of Q over all testcases doesn't exceed 10^5

Output

For each testcase print Q lines, each line should contain 2 space separated integers, denoting maximum possible OR and the minimum number of moves required.

Example

standard input	standard output
1 3 2 10 10 5 1 2 1 3	15 2 15 0

Note

In the first query, $B = [10, 10]$. We make two moves:

1. Choose $j = 1, i = 0$. B becomes $[3, 10]$
2. Choose $j = 2, i = 2$. B becomes $[3, 12]$

The bitwise OR equals 15 after the two moves.

In the second query, $B = [10, 10, 5]$ and the bitwise OR equals 15, which can be shown to be optimal. So, no move needs to be made.

Problem B. Minimize Median

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given an array A containing N integers, each between 1 and M. N is **odd**.

You are also given an array *cost* of length M.

In one move, you can do the following:

- Pick an index i ($1 \le i \le N$) and an integer x ($1 \le x \le M$)
- Replace $A[i]$ with $\lfloor A[i]/x \rfloor$, for a cost of $cost[x]$.

Here, $\lfloor\rfloor$ denotes the floor function, i.e, $\lfloor y \rfloor$ is the largest integer that doesn't exceed y.

You can perform operations as long as their total cost doesn't exceed K.

Under this condition, find the minimum possible value of $median(A)$ that can be achieved.

As a reminder, $median(A)$ is the middle element of A when it is sorted. For example, $median([3, 1, 2]) = 2$.

Input

The first line contains a single integer T, the number of testcases. Then the testcases follow.

The first line of each test case contains three space-separated integers N, M, K.

The second line of each test case contains N space-separated integers $A[1], A[2], \cdots, A[N]$.

The third line of each test case contains M space-separated integers $cost[1], cost[2], \cdots, cost[M]$.

Constraints

- $1 \le T \le 10^5$
- $1 \le N \le 10^6$
- N is odd.
- $2 \le M \le 10^6$
- $0 \le K \le 10^9$
- $1 \le A[i] \le M$
- $1 \le cost[i] \le 10^9$
- The sum of N across all testcases doesn't exceed 10^6.
- The sum of M across all testcases doesn't exceed 10^6.

Output

For each testcase, print a single integer, the minimum possible median of A.

Example

standard input	standard output
3 3 5 0 2 5 2 3 2 4 6 13 3 5 3 2 5 3 3 2 4 6 13 3 5 6 2 5 2 3 2 4 6 13	2 2 1

Note

Test case 1: No moves can be made, so the answer is $median([2, 5, 2]) = 2$.

Test case 2: Perform the following move:

- Divide $A[3] = 3$ by $x = 2$. This sets $A[3] = 1$ for a cost of 2.

The answer is $median([2, 5, 1]) = 2$, which is optimal.

Test case 3: Perform the following moves:

- Divide $A[2] = 5$ by $x = 3$. This sets $A[2] = 1$ for a cost of 4.
- Divide $A[3] = 2$ by $x = 2$. This sets $A[3] = 1$ for a cost of 2.

The answer is $median([2, 1, 1]) = 1$, which is optimal.

Problem C. Exam Requirements

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 megabytes

Your university semester exams are coming up. There are N exams where each exam i ($1 \le i \le N$) happens continuously from time S_i to E_i (both inclusive). To pass an exam, you need to attend it in its entirety (full attendance for the entire duration of the exam is enough to pass the exam). You can only attend non-overlapping exams. Formally, for any exams i and j , you can attend both the exams only if the **closed** intervals $[S_i, E_i]$ and $[S_j, E_j]$ do not overlap. For example $[1, 3]$ and $[2, 5]$ overlap. Similarly, $[1, 3]$ and $[3, 10]$ overlap. But $[1, 3]$ and $[4, 5]$ don't overlap.

To graduate, there M requirements which you need to fullfil, each requirement is of the form: pass at least one of exams A or B ($1 \le A$, $B \le N$ and $A \ne B$).

You need to fullfil ALL requirements while only attending non-overlapping exams. Check whether it is possible for you to graduate (output `YES`/`NO` , **case-sensitive**).

Input

The first line contains T, the number of test cases. Then the testcases follow.

The first line of each testcase contains two integers N and M.

N lines follow, each containing 2 integers. The i-th of these lines contains S_i and E_i.

M lines follow, each containing 2 integers A, B (i.e. you MUST pass at least one of A, B).

Constraints

- $1 \le T \le 100$
- $1 \le N \le 100000$
- $0 \le M \le 100000$
- $0 \le S_i \le E_i \le 1000000000$
- $1 \le A, B \le N$, and $A \ne B$.
- The sum of N over all testcases doesn't exceed 100000.
- The sum of M over all testcases doesn't exceed 100000.

Output

For each testcase, print in a new line - `YES` if it is possible for you to graduate, otherwise print `NO` (**case-sensitive**).

Example

standard input	standard output
2 3 1 1 5 2 7 10 11 2 1 3 3 1 5 2 7 5 7 1 2 2 3 3 1	YES NO

Note

- Test case 1: There are 3 exams, and 1 requirement. You can pass any of the exams 1 or 2 to fulfill this requirement, and graduate.

- Test case 2: There are 3 exams, and 3 requirements. All 3 exams overlap with each other, so you will only be able to attend at most 1 exam. But to fullfil all 3 requirements, you would need to attend at least 2 of these exams. Hence it is not possible to graduate.

Problem D. Central Subset

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given a undirected **connected** graph on N vertices and M edges. The vertices are numbered from 1 to N. You have to find a subset of vertices S, such that both conditions are satisfied:

- $|S| \leq \left\lceil \sqrt{N} \right\rceil$ - the number of vertices in S should be less than or equal to $\left\lceil \sqrt{N} \right\rceil$
- For every vertex u in the graph, there exists a vertex v in S such that $dist(u, v) \leq \left\lceil \sqrt{N} \right\rceil$

If there is no such subset then print `-1`.

Note:

- $\lceil x \rceil$ is the smallest integer greater than or equal to x.
- $dist(u, v)$ is the number of edges in the shortest path from u to v.

Input

First line contains a single integer T denoting the number of test cases.

The first line of each test case contains two space separated integers N and M denoting the number of vertices and the number of edges respectively.

The next M lines each contains two space separated integers u and v denoting that there is an edge between u and v.

There are no self-loops or multi-edges.

Constraints

- $1 \leq T \leq 2 \cdot 10^4$.
- $1 \leq N \leq 2 \cdot 10^5$.
- $0 \leq M \leq 10^6$.
- $1 \leq u, v \leq N$.
- Sum of N over all test cases does not exceed $2 \cdot 10^5$.
- Sum of M over all test cases does not exceed 10^6.
- The graph is connected.

Output

For every test case:

- If there is no valid subset then print `-1` in a new line.
- If there exists a subset S, then print the size of the subset in a new line. In the next line print $|S|$ space separated distinct vertices in any order. If there are multiple valid subsets then print any.

Example

standard input	standard output
2 4 3 1 2 2 3 3 4 6 7 1 2 2 3 3 1 1 4 4 5 5 6 6 4	1 2 3 5 2 6

Note

- For the first test case, $\lceil\sqrt{4}\rceil = 2$. The valid subsets are $\{2\}$,$\{3\}$,$\{1,2\}$,$\{1,3\}$,$\{1,4\}$,$\{2,3\}$,$\{2,4\}$,$\{3,4\}$. Any one of them can be printed.
- For the second test case, $\lceil\sqrt{6}\rceil = 3$. One example of a valid subset is $\{2,5,6\}$.

Problem E. Strange Keyboard

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	512 megabytes

You have a strange keyboard with N regular keys and one backspace key. You start with an empty string. At any instant, you can press any of the $N+1$ keys. Pressing the $i-$th regular key appends the string S_i to the current string, and pressing the backspace key does nothing if the current string has length $< K$, and otherwise deletes the last K characters of the current string.

You want to form a string T. Is it possible to do so? If it is possible, what is the minimum number of key-presses required?

Input

The first line contains Q, the number of test cases. Then the test cases follow.

The first line of each test case contains N and K, the number of regular keys, and the number of characters deleted by the backspace key.

$i-th$ of the next N lines contains S_i, the string corresponding to the $i-th$ regular key.

The last line of the testcase contains the string T to be formed.

Constraints

- $1 \leq Q \leq 100$
- $1 \leq N \leq 10^6$
- $1 \leq K \leq 5000$
- The sum of the lengths of all the strings S_i over all the testcases doesn't exceed 10^6
- The sum of the length of T over all the testcases doesn't exceed 5000.
- Strings S_i and T contain English lowercase letters only.

Output

For each testcase:

If it is impossible to form the string T, print -1 on a new line.

Else, print the minimum number of key presses required to form the string T, on a new line.

Example

standard input	standard output
2 2 3 defgh abc abcde 1 1 a b	3 -1

Note

In the first testcase, we can do the following:

1. Press the second regular key. After this, we get `abc`.
2. Press the first regular key. We now have `abcdefgh`.
3. Press the backspace key. We now have `abcde` as required.

In the second testcase, it is impossible to form the required string.

Problem F. Longest Strictly Increasing Sequence

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Given an array b of length n, find an array a of length n such that, for each $1 \le i \le n$, the length of the longest strictly increasing subsequence of $[a[1], a[2], \cdots, a[i]]$ is equal to $b[i]$.

For an array c of length m, a subsequence $c[i_1], c[i_2], \cdots, c[i_k]$ where $1 \le i_1 < i_2 < \cdots < i_k \le m$ is called strictly increasing if $c[i_1] < c[i_2] < \cdots < c[i_k]$.

Input

The first line contains a single integer T, denoting the number of test cases.

Each test case contains two lines:

- First line contains n - the size of array b.
- Second line contains n space-separated integers where the i-th integer represents $b[i]$.

Constraints

- $1 \le T \le 4\,000$
- $1 \le n \le 10$
- $1 \le b[i] \le 10$
- $1 \le$ sum n of all tests in a testfile $\le 20\,000$
- $1 \le a[i] \le 100$

Output

For each test case, print `YES` if there exists an array a that satisfies the conditions, `NO` otherwise on a new line.

If `YES`, print n space-separated integers representing elements of the array a in a new line.

Example

standard input	standard output
2 6 1 2 3 2 5 7 2 1 2	NO YES 1 2

Note

In the first test case, we can prove that no array exists which satisfies the condition.

In the second test case, $[4, 9]$ satisfies all conditions. LIS of $[4]$ is $[4]$ and its length is 1, and LIS of $[4, 9]$ is $[4, 9]$ and its length is 2. Other acceptable answers include $[5, 20]$ and $[25, 26]$. On the other hand, $[5, 5]$ and $[10, 5]$ are incorrect answers.

Problem G. Perfect Strings

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

Consider a character set σ of size c. There are c^{2n} strings of length $2n$, each of whose characters lies in σ. Let's call such a string perfect if the set of its indices $\{1, 2, .., 2n\}$ can be partitioned into n pairs, such that:

- Each index is a part of exactly one pair
- For each pair (i, j), $S[i] = S[j]$
- No two pairs are entangled, that is, for any two pairs (i, j) and (k, l), $i < k < j < l$ must NOT be true.

Given n and c, count the number of perfect strings of length $2n$, modulo $10^9 + 7$.

Input

The first line contains T, the number of testcases. Then the testcases follow.

Each testcase consists of two space separated integers, n and c.

Constraints

- $1 \leq T \leq 10^5$
- $1 \leq n, c \leq 10^7$
- The sum of n over all testcases doesn't exceed 10^7.

Example

standard input	standard output
2 3 1 2 2	1 6

Note

In the first testcase, there is only one string and it is clearly perfect

In the second testcase, let the character set be $\{a, b\}$. The perfect strings are (along with a partition of their indices into pairs):

aaaa	$\{(1, 4), (2, 3)\}$
aabb	$\{(1, 2), (3, 4)\}$
abba	$\{(1, 4), (2, 3)\}$
baab	$\{(1, 4), (2, 3)\}$
bbaa	$\{(1, 2), (3, 4)\}$
bbbb	$\{(1, 2), (3, 4)\}$

Problem H. Treelection

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

The company X has N employees numbered from 1 through N. For every $2 \le u \le N$, the employee numbered P_u $(1 \le P_u < u)$ is the manager of the employee numbered u. Employee 1 is the CEO and has no mananger. Employee v is said to be a leader of employee u if v is the manager of u or there is an employee w such that v is the manager of w and w is a leader of u.

The company X wants to setup a work council through elections, in which every employee except the CEO will vote. Unfortunately the elections are rigged, and employees can only vote for one of their leaders.

Find out which employees can end up being the sole winner of the election. An employee is the sole winner if they get **strictly** more votes than any other employee.

Input

The first line contains T, the number of testcases. Then the testcases follow

Each testcase consists of two lines.

The first line which contains N.

The second line contains $N-1$ space separated integers, $P_2, P_3, ..P_N$, where P_i is the manager of the employee i. It is guaranteed that $1 \le P_i < i$ for all valid i.

Constraints

- $1 \le T$
- $2 \le N \le 10^6$
- The sum of N over all testcases doesn't exceed 10^6.
- $1 \le P_i < i$ for all $2 \le i \le N$.

Output

For each testcase, print a single line containing a string of length N, whose $i-th$ character is 1 if the employee i can be the sole winner of the election, and 0 otherwise.

Example

standard input	standard output
2 4 1 2 3 5 1 1 2 2	1100 10000

Note

In the first testcase, employee 2 will be the sole winner if employee 2 votes for employee 1 and employees 3 and 4 vote for employee 2. In this case employee 1 gets 1 vote and employee 2 gets 2 votes.

Problem I. Disk Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

You are given a set of n disks in the plane. The disks can have different radii. The disks are not intersecting, and not touching. The edge of a disk is included in its area. Your job is to draw $n-1$ straight line segments to connect the disks, such that there exists a path from any disk to any other disk, that only walks inside the areas of the disks and on the line segments.

There are some constraints on the line segments.

- The line segments should have integer coordinate endpoints.
- A line segment can only touch or intersect with at most two disks.
- Any two line segments cannot intersect, and cannot touch. The only exception is that it is allowed for two line segments to share an endpoint.

Input

The first line contains an integer n ($2 \le n \le 200\,000$), denoting the number of disks.

Each of the next n lines contains the description of a disk. Each line contains three integers x_i, y_i and r_i ($0 \le x_i, y_i \le 10^9$, $1 \le r_i \le 10^9$), denoting a disk with centre point x_i, y_i and with a radius of r_i.

It is guaranteed that no two disks touch or intersect.

Output

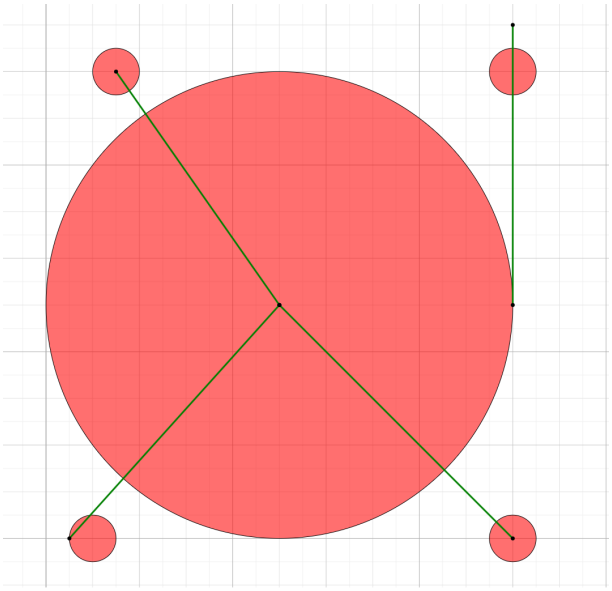
Output "`YES`" if there exists a solution, otherwise print "`NO`". If the answer is "`YES`", on the following lines output a description of the line segments.

The description contains $n-1$ lines. Each of the $n-1$ lines should contain 4 integers x_1, y_1, x_2, y_2, denoting a line segment that connects points (x_1, y_1) and (x_2, y_2). These two points should not be the same. Further more it must hold that $0 \le x_1, y_1, x_2, y_2 \le 10^9$. It can be proven that if there exists a solution, then there also exists a solution with the coordinates of the line segments being bounded like this.

Examples

standard input	standard output
3 1 0 3 10 10 6 0 5 1	YES 0 4 7 12 0 0 16 8
2 1 1 1 3 3 1	YES 2 1 3 2
5 10 10 10 2 0 1 20 20 1 3 20 1 20 0 1	YES 1 0 10 10 20 0 10 10 20 10 20 22 3 20 10 10

Note



Visualisation of sample 3

Problem J. Talk That Talk

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Gauri is a big fan of the K-pop group TWICE. Recently, TWICE has released a song called "Talk That Talk", and since then Gauri has been mesmerized by evenly-spaced triplets.

Given an integer t binary string s, where its indices are labelled from 1 to $|s|$, we define its t-value as the number of TTT-triplets. A triplet (i, j, k) is a TTT-triplet if and only if following conditions are met:

1. $1 \le i < j < k \le |s|$
2. $j - i = k - j$, and $1 \le j - i \le t$
3. $s_i = s_j = s_k$

Today Gauri received an integer t and a string w of length $p - 1$ as a present, where p is a prime. She noticed that for all $1 \le x \le p - 1$, $w_x = 1$ if there exists an integer z such that $z^2 \equiv x \pmod{p}$, and 0 otherwise. Help Gauri compute the t-value of w.

Each test consist of multiple testcases. There are T test cases.

Input

The first line consists of an integer T, the number of testcases.

The next T lines consists of 2 integers p and t.

Constraints

- $5 \le p \le 10^{12}$, and p is a prime number.
- $1 \le t \le 10^6$
- $1 \le T \le 5 \cdot 10^5$
- Sum of t among all tests is at most 10^6.

Output

Output T lines, one for each test case denoting the t-value of w.

Example

standard input	standard output
7 7 32 13 1 13 2 67 11 2003 44 1000003 1984 999999999989 987654	0 2 2 146 21510 495014784 246913256130162788

Note

When $p = 13$, we get $w = 101100001101$, possible TTT-triplets are $(5, 6, 7)$, $(6, 7, 8)$, $(2, 5, 8)$, and $(5, 8, 11)$. Now if $t = 2$, the latter two triplets have $j - i > t$, violating condition 2. Thus, the answer for $p = 13$, $t = 2$ is 2.

Problem K. XOR Dice

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given two integers n and d.

Find n dice with faces labelled with nonnegative integers not more than 10^6 such that:

- for each die, the six numbers written on its faces are all distinct, and
- if you roll all dice, the bitwise XOR of the n numbers on top is **always** divisible by d.

Under the given constraints, we can prove that such dice always exist.

Input

The only line contains two integers n and d ($1 \le n \le 100$; $2 \le d \le 60$) — the number of dice and the number their XOR has to be divisible by, respectively.

Output

Output n lines, the i-th of which contains six distinct space-separated nonnegative integers at most 10^6 - the faces of the i-th die.

If there are multiple possible answers, output any of them.

Example

standard input	standard output
3 2	1 3 5 7 9 11 3 5 7 9 11 2023 0 2 4 6 100000 10

Note

There are three dice:

- Die 1 has faces $[1, 3, 5, 7, 9, 11]$.
- Die 2 has faces $[3, 5, 7, 9, 11, 2023]$.
- Die 3 has faces $[0, 2, 4, 6, 100000, 10]$.

Suppose we rolled the dice, and they landed on 7, 3, and 2. Then their bitwise XOR is $7 \oplus 3 \oplus 2 = 6$, which is a multiple of 2.

Problem L. (1, 2) Nim

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Sprague and Grundy are playing a game. There are N piles of stones numbered $1, 2, ..., N$. The $i-th$ pile contains $A[i]$ stones.

The following is defined as a **move**:

- Choose a non-empty pile, and remove any non-zero number of stones from it. Formally, choose some i with $A[i] > 0$ and choose $1 \leq j \leq A[i]$. Then replace $A[i]$ by $A[i] - j$.

The players take alternating turns starting with Sprague. In his turn, Sprague must make exactly one move. The rule for Grundy's turn is the following:

- First Grundy must make one move.
- After this move, if atleast one stone is remaining, he must make exactly one more move.

The player to remove the last remaining stone wins the game. Find out who wins if both play optimally.

Input

The first line contains T, the number of testcases. Then the testcases follow, each consisting of two lines:

- The first line of each testcase contains N.
- The second line contains N space separated integers $A[1], A[2], ..., A[N]$.

Constraints

- $1 \leq T \leq 10^4$
- $1 \leq N \leq 10^5$
- $1 \leq A[i] \leq 10^9$ for all $1 \leq i \leq N$
- The sum of N over all testcases doesn't exceed 10^5

Output

For each testcase, print a single line containing `Sprague` if Sprague wins the game and `Grundy` otherwise.

Please note that the checker is `case-sensitive`. Printing `sprague` or sPRAGuE instead of `Sprague` will give `Wrong Answer`.

Example

standard input	standard output
3 2 1 2 1 5 4 1 7 2 9	Grundy Sprague Grundy

Note

In the first testcase, one can verify that Grundy wins. For example, if Sprague removes 1 stone from the second pile, then in his turn, Grundy can remove 1 stone from the first pile in his first move and 1 stone from the second pile in his second move. If Sprague removes 2 stones from the second pile, Grundy can remove the only remaining stone in one move and win the game instantly.

Problem M. Graphs and Colors

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given a complete graph of N nodes. You have to color each edge of this graph using one of the K given colors. Colors are numbered from 1 to K.

- Let's denote d_j as the diameter of the graph formed using these N nodes and edges of color j.
- The value of d_j must be less than or equal to 4 for each color.

Note: In a graph, if there does not exist a path between two nodes, the distance between them can be assumed to be infinity.

Input

The first line contains T, the number of testcases.

Each of the next T lines contains two integers N and K.

Constraints

- $2 \leq N \leq 100$
- $1 \leq K \leq N * (N-1)/2$
- $1 \leq$ Sum of N over all testcases $\leq 10\,000$

Output

For each testcase:

The first line should contain a string `YES` or `NO`. Print `NO` if coloring is impossible and `YES` otherwise.

If the answer is `YES`, print $N-1$ more lines, ith line should have i integers, where the jth integer on the ith line should contain the color of the edge from node j to node $i+1$.

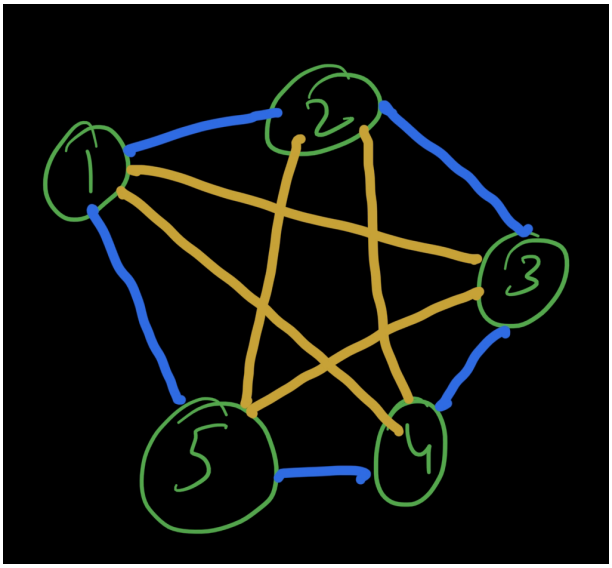
Example

standard input	standard output
2 5 10 5 2	NO YES 1 2 1 2 2 1 1 2 2 1

Note

In the first example, it can be proven that its impossible to add such coloring.

In the second example, one such coloring is shown below. Blue is color 1, and Yellow is color 2.


1
2
3
4
5

Problem N. Red Black Grid

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Given integers N and K, construct an $N \times N$ grid, each of whose cells is colored either red or black, such that there are exactly K unordered pairs of oppositely colored adjacent cells, or claim that no such grid exists.

A pair of cells is called adjacent if they share an edge. Formally, let the rows be numbered $1, 2, \cdots N$ from left to right and the columns be numbered $1, 2, \cdots N$ from top to bottom. The cell (i, j) denotes the cell with row number i and column number j. Two cells (a, b) and (c, d) are adjacent iff $|c - a| + |d - b| = 1$.

If there are multiple valid grids, you can output any of them.

Input

The first line contains T, the number of testcases. Then, the testcases follow.

Each testcase consists of two space separated integers N and K.

Constraints

- $1 \leq T \leq 10^4$
- $1 \leq N \leq 10^3$
- $0 \leq K \leq 2 \times N \times (N - 1)$
- The sum of N^2 over all testcases doesn't exceed 10^6.

Output

For each testcase, if no valid grid exists, print `Impossible` on a new line.

Else print $N + 1$ lines. Print `Possible` on the first line. Then, print N lines, the $i - th$ of which contains the $i - th$ row of the grid. For each cell of the row from left to right, if it is colored red, print R and if it is colored black, print B.

Example

standard input	standard output
2 3 6 3 1	Possible BRB RBB BBB Impossible

Note

In the first testcase, the pairs of adjacent oppositely colored cells are:

- $(1, 1)$ and $(2, 1)$
- $(1, 2)$ and $(1, 3)$
- $(1, 2)$ and $(2, 2)$

- $(2, 1)$ and $(3, 1)$
- $(2, 2)$ and $(3, 2)$
- $(2, 3)$ and $(3, 3)$