# A. Disk Tree

You are given a set of `n` disks in the plane. The disks can have different radii. No two disks intersect or touch each other.

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 on the circumference or inside the areas of any of the `n` disks, or on the line segments.

There are some constraints on the line segments:

- The line segments should have integer coordinate endpoints in the range $[0, 10^9]$.
- 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 Format

The first line contains an integer `n`, 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]`, denoting a disk with centre point `(x[i], y[i])` and a radius of `r[i]`.

# Constraints

- $2 <= n <= 2 \times 10^5$
- $0 <= x[i], y[i] <= 10^9$
- $1 <= r[i] <= 10^9$
- It is guaranteed that no two disks touch or intersect.

# Output Format

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 the points `(x[1],y[1])` and `(x[2],y[2])`. These two points should not be the same. Further more it must hold that $0 <= x[1], y[1], x[2], y[2] <= 10^9$. It can be proven that if the answer is `YES`, then there exists a solution in which the coordinates of the line segments are bounded like this.

## Sample 1

### Input

```
3
1 0 3
10 10 6
0 5 1
```

### Output

```
YES
0 4 7 12
0 0 16 8
```

## Sample 2

### Input

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

### Output

```
YES
2 1 3 2
```

## Sample 3

### Input

```
5
10 10 10
2 0 1
20 20 1
3 20 1
20 0 1
```

### Output

```
YES
1 0 10 10
20 0 10 10
20 10 20 22
3 20 10 10
```

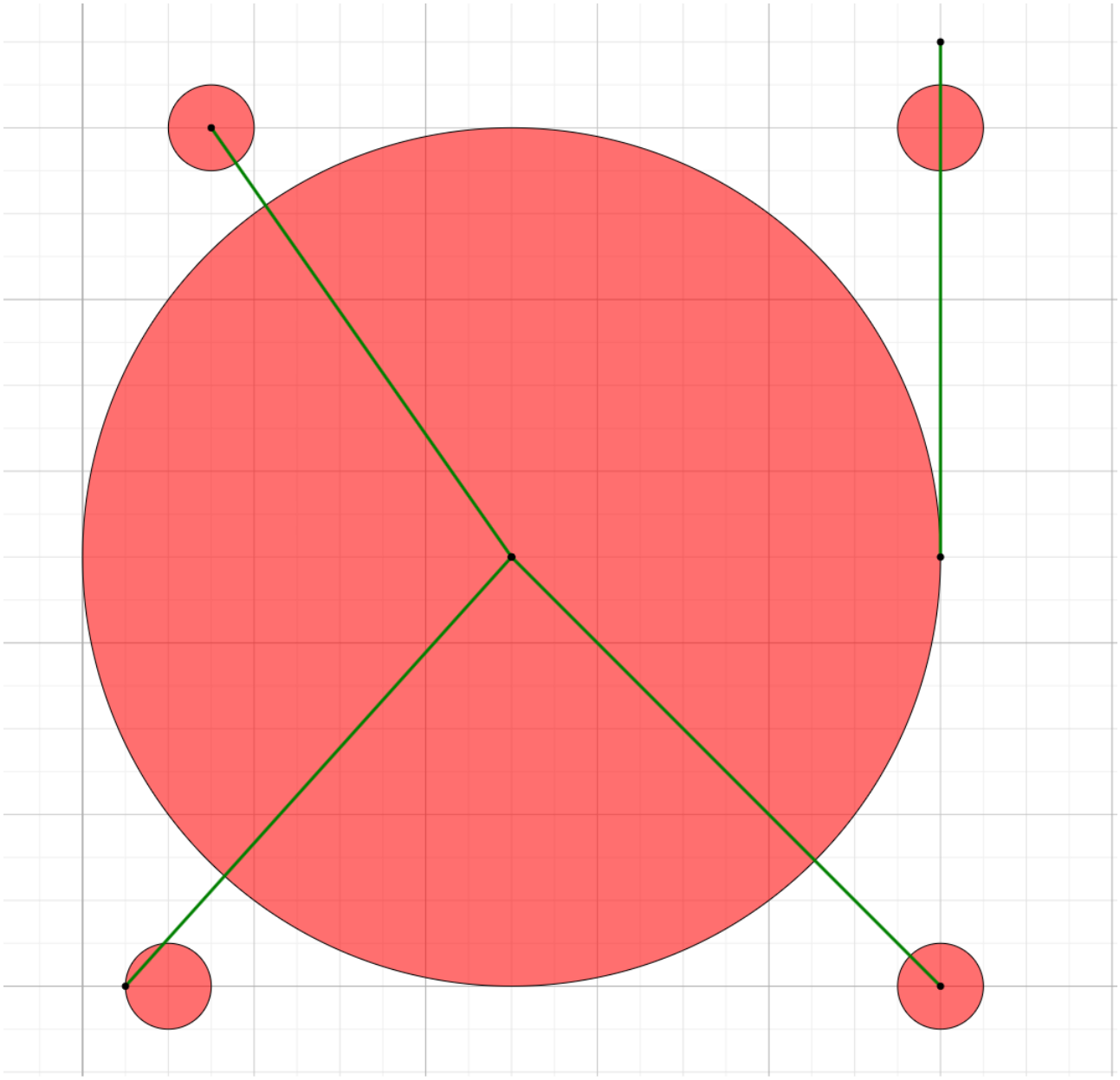

Figure 1: sample explanation

# B. Perfect Strings

Consider a character set $\sigma$ of size `c`. There are $c^(2n)$ strings of length $2n$, each of whose characters lies in $\sigma$. 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 Format

- 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

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

## Sample 0

### Input

```
2
3 1
2 2
```

### Output

```
1
6
```

### Explanation

- 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)}`

# C. Red Black Grid

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 $1, 2, ..., N$ from left to right and the columns be numbered $1, 2, ..., N$ 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 Format

- 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 Format

- 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`.

## Sample 0

### Input

```
2
3 6
3 1
```

### Output

```
Possible
RRB
BBB
RRR
Impossible
```

### Explanation

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)`

# D. Minimize Median

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 Format

- 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], ..., A[N]`.
- The third line of each test case contains `M` space-separated integers `cost[1], cost[2], ..., cost[M]`.

## Output Format

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

## 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 $10^6$.

## Sample Input

```
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
```

## Sample Output

```
2
2
1
```

## Explanation

**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.

# E. Same Sum

You are given a binary string `S` of length `N`.

Find four **distinct** indices $1 \leq i, j, k, l \leq N$ such that

- $S_i = S_j = 0$,
- $S_k = S_l = 1$, and
- $i + j = k + l$.

If no such indices exist, you should report about it.

You need to answer `T` independent test cases.

## Input Format

- The first line of the input contains an integer `T` — the number of testcases.
- The first line of each test case contains an integer `N` — the length of the string.
- The second line of each test case contains a binary string `S` of length `N` consisting of characters `0` and `1`.

## Output Format

For each test case, output the following:

- If there exist four indices `i, j, k, l` satisfying the conditions in the statement, output them in that order on a single line.
- Otherwise, output only the integer `-1` in a single line.

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

## Constraints

- $1 \leq T \leq 1000$
- $1 \leq N \leq 2 \times 10^5$
- The sum of `N` across all test cases does not exceed $2 \times 10^5$.

## Sample 0

### Input

```
5
10
0101000110
1
0
9
111000011
4
1111
8
10101100
```

### Output

```
5 6 2 9
-1
4 6 8 2
-1
7 4 6 5
```

### Explanation

In the first testcase, `S = 0101000110`. A valid solution is `i = 5`, `j = 6`, `k = 2` and `l = 9`, since

- $S_i = S_5 = 0$, $S_j = S_6 = 0$,
- $S_k = S_2 = 1$, $S_l = S_9 = 1$,

- $i + j = 5 + 6 = 11$ and $k + l = 2 + 9 = 11$.

In the second testcase there do not exist four distinct indices $1 \leq i, j, k, l \leq 1$.

# F. Treelection

The company X has `N` employees numbered from `1` through `N`. For every $2 \leq u \leq N$, the employee numbered $P_u$ ($1 \leq P_u < u$) 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 Format

- 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 $1 \leq P_i < i$ for all valid `i`.

## Constraints

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

## Output Format

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.

## Sample 0

### Input

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

### Output

```
1100
10000
```

### Explanation

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.

# G. Intersecting Paths

You're given a tree with $N$ nodes numbered $1, 2, ..., N$. Initially you have an empty set $S$ of simple paths. There will be $Q$ queries, each containing two integers $u, v$ $(1 \leq u \leq v \leq N)$:

- Let $P$ be the shortest path between $u$ and $v$. If $P$ is in $S$, remove it from $S$, otherwise add it to $S$.

After each query, print the number of unordered pairs of paths in $S$, that intersect. Two paths are said to intersect if they contain a common node.

## Input Format

- The first line contains two integers, $N$ and $Q$.
- Each of the next $N-1$ lines contain two space separated integers, denoting the endpoints of an edge of the tree.
- Each of the next $Q$ lines contain two space separated integers $u$ and $v$, denoting the endpoints of the path $P$ to be added or removed from $S$.

## Constraints

- $1 \leq N, Q \leq 2 \times 10^5$
- The given graph is a tree.
- For any query, $1 \leq u \leq v \leq N$

## Output format

After each query, print on a new line, the number of unordered pairs of paths in $S$, that intersect.

## Sample 0

### Input

```
8 4
1 2
1 3
2 4
2 5
4 6
5 7
5 8
5 6
3 4
5 8
5 6
```

### Output

```
0
1
2
0
```

### Explanation

Let `(a, b)` denote the path with endpoints `a` and `b`. Then, the queries are:

1. Add `(5, 6)` to `S`. After this, , `S = {(5, 6)}`
2. Add `(3, 4)` to `S`. After this, , `S = {(3, 4), (5, 6)}`
3. Add `(5, 8)` to `S`. After this, , `S = {(3, 4), (5, 6), (5, 8)}`
4. Remove `(5, 6)` from `S`. After this, , `S = {(3, 4), (5, 8)}`

There are three possible unordered pairs of paths:

- `(3, 4)` and `(5, 6)` intersect.
- `(5, 6)` and `(5, 8)` intersect.

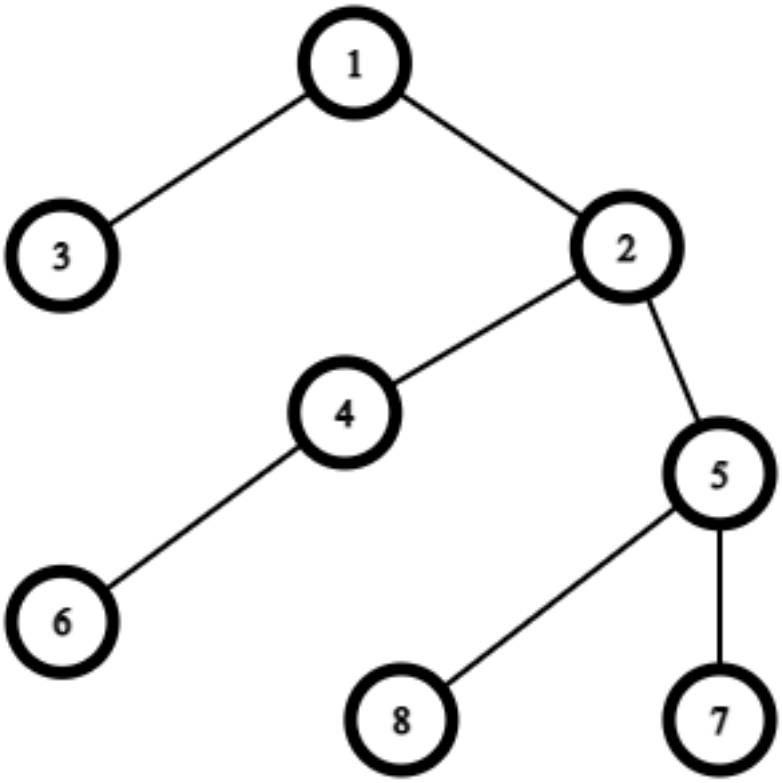


Figure 2: tc

- `(3, 4)` and `(5, 8)` don't intersect.

Therefore, the answers to the queries would be `0, 1, 2, 0` in order.

# H. Strange Keyboard

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 Format

- 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.

## Output Format

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.

## Sample 0

### Input

```
2
2 3
defgh
abc
abcde
1 1
a
b
```

### Output

```
3
-1
```

### Explanation

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.

# I. Coworkers

There is a large company with `n` employees working there. All employees are numbered from `1` to `n`. There are `m` pairs of coworkers in the company, the `i`-th pair is formed with employee `a[i]` and employee `b[i]`. That means that `b[i]` is one of the coworkers of `a[i]`, and `a[i]` is one of the coworkers of `b[i]`.

The founder of the company decided to check the level of loyalty of the employees. To do this, he gave all employees an assignment. For employee `i` he chose a number `t[i]` and asked a question: Name the top t[i] coworkers you work with. Of course, `t[i]` is not bigger than the total number of coworkers employee `i` has.

After this, a pair of coworkers are considered `friends` if they both chose each other as one of the top coworkers they are working with. Similarly, a pair of coworkers are considered `enemies` if they both didn't choose each other as one of the top coworkers they are working with.

The level of loyalty is counted as the number of pairs of coworkers who are `friends` minus the number of pairs of coworkers who are `enemies`.

Determine the maximum level of loyalty that can be achieved if each of the employees answers the question optimally.

### Input

- The first line of the input contains two integers `n` and `m`.
- The second line contains `n` integers `t[1], t[2], ..., t[n]`, the number of top coworkers that the employee `i` needs to choose.
- The next `m` lines contain two integers each. The `i`-th of them contain `a[i]` and `b[i]`, the `i`-th pair of coworkers.

### Output

Output the maximum level of loyalty that can be achieved.

### Constraints

- $1 \leq n \leq 2 \times 10^5$
- $0 \leq m \leq min(2 \times 10^5, n \times (n-1)/2)$
- $0 \leq t[i] < n$. It's guaranteed that `t[i]` does not exceed overall number of coworkers employee `i` has.
- $1 \leq a[i], b[i] \leq n, a[i]! = b[i]$. It's guaranteed that every pair of coworkers appears at most once.

## Sample 0

### Input

```
5 4
2 1 2 1 0
1 2
2 3
3 4
4 1
```

### Output

```
2
```

### Explanation

- For the employee `1`, choose employees `2` and `4` to be the top coworkers
- For the employee `2`, choose employee `1` to be the top coworker
- For the employee `3`, choose employees `2` and `4` to be the top coworkers
- For the employee `4`, choose employee `1` to be the top coworker.

Doing so, employees `1` and `2` are friends. Also, employees `1` and `4` are friends. There are no enemies. So, the answer is `2 - 0 = 2`. It can be verified that this is optimal.

## Sample 1

### Input

```
4 6
0 0 0 0
1 2
1 3
1 4
2 3
2 4
3 4
```

### Output

```
-6
```

# J. (1, 2) Nim

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 Format

- 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 Format

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`.

## Sample 0

### Input

```
3
2
1 2
1
5
4
1 7 2 9
```

### Output

```
Grundy
Sprague
Grundy
```

### Explanation

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.

# K. XOR Dice

You are given two integers `N` and `D`.

Find `N` six-sided 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, it can be shown that a solution always exists.

## Input Format

The only line contains the two integers `N` and `D`.

## Output Format

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.

## Constraints

- $1 \leq N \leq 100$
- $2 \leq D \leq 60$

## Sample 0

### Input

```
3 2
```

### Output

```
1 3 5 7 9 11
3 5 7 9 11 2023
0 2 4 6 1000000 10
```

### Explanation

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, 1000000, 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.