Codeforces Round #685 (Div. 2)

A. Subtract or Divide

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

Ridbit starts with an integer n.

In one move, he can perform one of the following operations:

- divide n by one of its **proper** divisors, or
- subtract 1 from n if n is greater than 1.

A proper divisor is a divisor of a number, excluding itself. For example, 1, 2, 4, 5, and 10 are proper divisors of 20, but 20 itself is not.

What is the minimum number of moves Ridbit is required to make to reduce n to 1?

Input
The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases.

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

Output
For each test case, output the minimum number of moves required to reduce n to 1.

Example

input
6 1 2 3 4 6 9
output
0 1 2 2 2 3

Note
For the test cases in the example, n may be reduced to 1 using the following operations in sequence

1

$2 \rightarrow 1$

$3 \rightarrow 2 \rightarrow 1$

$4 \rightarrow 2 \rightarrow 1$

$6 \rightarrow 2 \rightarrow 1$

$9 \rightarrow 3 \rightarrow 2 \rightarrow 1$

B. Non-Substring Subsequence

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

Hr0d1y has q queries on a binary string s of length n. A binary string is a string containing only characters '0' and '1'.

A query is described by a pair of integers l_i, r_i $(1 \le l_i < r_i \le n)$.

For each query, he has to determine whether there exists a good subsequence in s that is equal to the substring $s[l_i \ldots r_i]$.

- A substring $s[i \ldots j]$ of a string s is the string formed by characters $s_i s_{i+1} \ldots s_j$.
- String a is said to be a subsequence of string b if a can be obtained from b by deleting some characters without changing the order of the remaining characters.
- A subsequence is said to be **good** if it is not contiguous and has length ≥ 2. For example, if s is "1100110", then the subsequences $s_1 s_2 s_4$ ("**11**0**0**110") and $s_1 s_5 s_7$ ("**1**100**11**0") are good, while $s_1 s_2 s_3$ ("**110**0110") is not good.

Can you help Hr0d1y answer each query?

Input
The first line of the input contains a single integer t ($1 \leq t \leq 100$) — the number of test cases. The description of each test case is as follows.

The first line contains two integers n ($2 \leq n \leq 100$) and q ($1 \leq q \leq 100$) — the length of the string and the number of queries.

The second line contains the string s.

The i-th of the next q lines contains two integers l_i and r_i ($1 \leq l_i < r_i \leq n$).

Output
For each test case, output q lines. The i-th line of the output of each test case should contain "YES" if there exists a good subsequence equal to the substring $s[l_i \ldots r_i]$, and "NO" otherwise.

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

Example

input
2 6 3 001000 2 4 1 3 3 5 4 2 1111 1 4 2 3
output
YES NO YES NO YES

Note
In the first test case,

- $s[2 \ldots 4]$ = "010". In this case $s_1 s_3 s_5$ ("**0**0**1**0**0**0") and $s_2 s_3 s_6$ ("0**01**0**0**0") are good suitable subsequences, while $s_2 s_3 s_4$ ("0**01**00") is not good.
- $s[1 \ldots 3]$ = "001". No suitable good subsequence exists.
- $s[3 \ldots 5]$ = "100". Here $s_3 s_5 s_6$ ("00**1**0**0**0") is a suitable good subsequence.

C. String Equality

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

Ashish has two strings a and b, each of length n, and an integer k. The strings only contain lowercase English letters.

He wants to convert string a into string b by performing some (possibly zero) operations on a.

In one move, he can either

- choose an index i ($1 \leq i \leq n-1$) and swap a_i and a_{i+1}, or
- choose an index i ($1 \leq i \leq n-k+1$) and if $a_i, a_{i+1}, \ldots, a_{i+k-1}$ are **all equal** to some character c ($c \neq$ 'z'), replace each one with the next character $(c+1)$, that is, 'a' is replaced by 'b', 'b' is replaced by 'c' and so on.

Note that he can perform any number of operations, and the operations can only be performed on string a.

Help Ashish determine if it is possible to convert string a into b after performing some (possibly zero) operations on it.

Input
The first line contains a single integer t ($1 \leq t \leq 10^5$) — the number of test cases. The description of each test case is as follows.

The first line of each test case contains two integers n ($2 \leq n \leq 10^6$) and k ($1 \leq k \leq n$).

The second line of each test case contains the string a of length n consisting of lowercase English letters.

The third line of each test case contains the string b of length n consisting of lowercase English letters.

It is guaranteed that the sum of values n among all test cases does not exceed 10^6.

Output
For each test case, print "Yes" if Ashish can convert a into b after some moves, else print "No".

You may print the letters of the answer in any case (upper or lower).

Example

input
4 3 3 abc bcd 4 2 abba azza 2 1 zz aa 6 2 aaabba ddddcc

output
No Yes No Yes

Note
In the first test case it can be shown that it is impossible to convert a into b.

In the second test case,

"a**bb**a" $\xrightarrow{\text{inc}}$ "a**cc**a" $\xrightarrow{\text{inc}}$ ... $\xrightarrow{\text{inc}}$ "azza".

Here "swap" denotes an operation of the first type, and "inc" denotes an operation of the second type.

In the fourth test case,

"aaab**ba**" $\xrightarrow{\text{swap}}$ "aaa**ba**b" $\xrightarrow{\text{swap}}$ "**aa**aabb" $\xrightarrow{\text{inc}}$ ... $\xrightarrow{\text{inc}}$ "dd**aa**bb" $\xrightarrow{\text{inc}}$ ... $\xrightarrow{\text{inc}}$ "dddd**bb**" $\xrightarrow{\text{inc}}$ ... $\xrightarrow{\text{inc}}$ "ddddcc".

D. Circle Game

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

Utkarsh is forced to play yet another one of Ashish's games. The game progresses turn by turn and as usual, Ashish moves **first**.

Consider the 2D plane. There is a token which is initially at $(0, 0)$. In one move a player must increase either the x coordinate or the y coordinate of the token by **exactly** k. In doing so, the player must ensure that the token stays within a (Euclidean) distance d from $(0, 0)$.

In other words, if after a move the coordinates of the token are (p, q), then $p^2 + q^2 \leq d^2$ must hold.

The game ends when a player is unable to make a move. It can be shown that the game will end in a finite number of moves. If both players play optimally, determine who will win.

Input
The first line contains a single integer t ($1 \leq t \leq 100$) — the number of test cases.

The only line of each test case contains two space separated integers d ($1 \leq d \leq 10^5$) and k ($1 \leq k \leq d$).

Output
For each test case, if Ashish wins the game, print "Ashish", otherwise print "Utkarsh" (without the quotes).

Example

input
5 2 1 5 2 10 3 25 4 15441 33

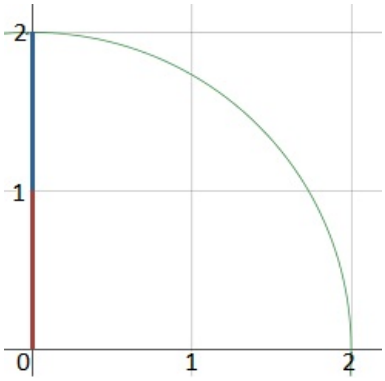
output
Utkarsh Ashish Utkarsh Utkarsh Ashish

Note

In the first test case, one possible sequence of moves can be

$(0,0) \xrightarrow{\text{Ashish}} (0,1) \xrightarrow{\text{Utkarsh}} (0,2).$

Ashish has no moves left, so Utkarsh wins.



E1. Bitwise Queries (Easy Version)

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The only difference between the easy and hard versions is the constraints on the number of queries.

This is an interactive problem.

Ridbit has a hidden array a of n integers which he wants Ashish to guess. Note that n is a **power of two**. Ashish is allowed to ask three different types of queries. They are of the form

- `AND` i j: ask for the bitwise AND of elements a_i and a_j $(1 \le i, j \le n, i \ne j)$
- `OR` i j: ask for the bitwise OR of elements a_i and a_j $(1 \le i, j \le n, i \ne j)$
- `XOR` i j: ask for the bitwise XOR of elements a_i and a_j $(1 \le i, j \le n, i \ne j)$

Can you help Ashish guess the elements of the array?

In this version, each element takes a value in the range $[0, n-1]$ (inclusive) and Ashish can ask no more than $n+2$ queries.

Input

The first line of input contains one integer n $(4 \le n \le 2^{16})$ — the length of the array. It is guaranteed that n is a **power of two**.

Interaction

To ask a query print a single line containing one of the following (without quotes)

- `"AND i j"`
- `"OR i j"`
- `"XOR i j"`

where i and j $(1 \le i, j \le n, i \ne j)$ denote the indices being queried.
For each query, you will receive an integer x whose value depends on the type of query. If the indices queried are invalid or you exceed the number of queries however, you will get $x = -1$. In this case, you should terminate the program immediately.

When you have guessed the elements of the array, print a single line `"! "` (without quotes), followed by n space-separated integers — the elements of the array.

Guessing the array does **not** count towards the number of queries asked.

The interactor is not adaptive. The array a does not change with queries.

After printing a query do not forget to output the end of the line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;

- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see the documentation for other languages.

Hacks

To hack the solution, use the following test format:

On the first line print a single integer n ($4 \leq n \leq 2^{16}$) — the length of the array. It **must** be a power of 2. The next line should contain n space-separated integers in the range $[0, n-1]$ — the array a.

Example

input
4 0 2 3
output
OR 1 2 OR 2 3 XOR 2 4 ! 0 0 2 3

Note
The array a in the example is $[0, 0, 2, 3]$.

E2. Bitwise Queries (Hard Version)

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The only difference between the easy and hard versions is the constraints on the number of queries.

This is an interactive problem.

Ridbit has a hidden array a of n integers which he wants Ashish to guess. Note that n is a **power of two**. Ashish is allowed to ask three different types of queries. They are of the form

- `AND` i j: ask for the bitwise AND of elements a_i and a_j ($1 \leq i, j \leq n, i \neq j$)
- `OR` i j: ask for the bitwise OR of elements a_i and a_j ($1 \leq i, j \leq n, i \neq j$)
- `XOR` i j: ask for the bitwise XOR of elements a_i and a_j ($1 \leq i, j \leq n, i \neq j$)

Can you help Ashish guess the elements of the array?

In this version, each element takes a value in the range $[0, n-1]$ (inclusive) and Ashish can ask no more than $n+1$ queries.

Input
The first line of input contains one integer n ($4 \leq n \leq 2^{16}$) — the length of the array. It is guaranteed that n is a **power of two**.

Interaction
To ask a query print a single line containing one of the following (without quotes)

- `"AND i j"`
- `"OR i j"`
- `"XOR i j"`

where i and j ($1 \leq i, j \leq n, i \neq j$) denote the indices being queried.
For each query, you will receive an integer x whose value depends on the type of query. If the indices queried are invalid or you exceed the number of queries however, you will get $x = -1$. In this case, you should terminate the program immediately.

When you have guessed the elements of the array, print a single line `"! "` (without quotes), followed by n space-separated integers — the elements of the array.

Guessing the array does **not** count towards the number of queries asked.

The interactor is not adaptive. The array a does not change with queries.

After printing a query do not forget to output the end of the line and flush the output. Otherwise, you will get `Idleness limit`

exceeded. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see the documentation for other languages.

Hacks

To hack the solution, use the following test format:

On the first line print a single integer n ($4 \leq n \leq 2^{16}$) — the length of the array. It **must** be a power of 2. The next line should contain n space-separated integers in the range $[0, n-1]$ — the array a.

Example

input
4 0 2 3
output
OR 1 2 OR 2 3 XOR 2 4 ! 0 0 2 3

Note
The array a in the example is $[0, 0, 2, 3]$.

F. Nullify The Matrix

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

Jeel and Ashish play a game on an $n \times m$ matrix. The rows are numbered 1 to n from top to bottom and the columns are numbered 1 to m from left to right. They play turn by turn. Ashish goes **first**.

Initially, each cell of the matrix contains a non-negative integer. Each turn, a player must perform **all** of the following actions in order.

- Choose a starting cell (r_1, c_1) with **non-zero** value.
- Choose a finishing cell (r_2, c_2) such that $r_1 \leq r_2$ and $c_1 \leq c_2$.
- Decrease the value of the starting cell by some positive non-zero integer.
- Pick any of the shortest paths between the two cells and either increase, decrease or leave the values of cells on this path unchanged. Note that:
  - a shortest path is one that passes through the least number of cells;
  - all cells on this path **excluding** the starting cell, but the finishing cell may be modified;
  - the resulting value of each cell must be a non-negative integer;
  - the cells are modified independently and not necessarily by the same value.

If the starting and ending cells are the same, then as per the rules, the value of the cell is decreased. No other operations are performed.

The game ends when all the values become zero. The player who is unable to make a move loses. It can be shown that the game will end in a finite number of moves if both players play optimally.

Given the initial matrix, if both players play optimally, can you predict who will win?

Input
The first line contains a single integer t ($1 \leq t \leq 10$) — the number of test cases. The description of each test case is as follows.

The first line of each test case contains two integers n and m ($1 \leq n, m \leq 100$) — the dimensions of the matrix.

The next n lines contain m space separated integers $a_{i,j}$ ($0 \leq a_{i,j} \leq 10^6$) — the values of each cell of the matrix.

Output

For each test case, if Ashish wins the game, print "Ashish", otherwise print "Jeel" (without the quotes).

Example

input
4 1 1 0 1 3 0 0 5 2 2 0 1 1 0 3 3 1 2 3 4 5 6 7 8 9
output
Jeel Ashish Jeel Ashish

Note

In the first test case, the only cell of the matrix is 0. There are no moves Ashish can make. Jeel is the winner.

In the second test case, Ashish can choose $(r_1, c_1) = (r_2, c_2) = (1, 3)$ and reduce the cell to 0, leaving $[0, 0, 0]$. Jeel cannot perform any moves. Ashish wins.