2019 LGE Code Jam: Final Round

Problem A. Street light posts

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 sec
Memory limit:	256 MB

Consider N street light posts on a 2D plane. The i-th pole is located at (x_i, y_i), and the coordinates are integers. No two poles are located at the same coordinates.

For any two poles i and j $(i < j)$, if there are street light poles at (x_i, y_j) and (x_j, y_i) as well, then we say the poles i and j are balanced. You need to determine whether the given street light poles are pairwise balanced (i.e., balanced for every pair of poles).

Input

The first line will contain a single integer T, the number of test cases.

For each test case, the first line will contain N (the number of street light poles).

Each of the next N lines will contain two integers (x_i and y_i)separated by a whitespace.

Output

For each test case, if the poles are pairwise balanced, then output "`BALANCED`" and otherwise output "`NOT BALANCED`" in a single line (without quotes).

Limits

- $1 \le T \le 5$
- $2 \le N \le 200,000$
- $-10^9 \le x_i, y_i \le 10^9$

Examples

standard input	standard output
2 6 2 3 2 -3 2 1 -2 3 -2 1 -2 -3 6 2 4 2 -3 2 1 -2 3 -2 1 -2 -3	BALANCED NOT BALANCED

Problem B. Mysterious Number

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.25 sec
Memory limit:	256 MB

Given N non-zero integers, if a non-zero integer M satisfies the following condition, then we say that M is a mysterious number with respect to the N integers.

- The remainder is the same for every one of the given N integers modulo M.

For any set of N integers, there is always at least one mysterious integer. For instance, 1 is always a mysterious number regardless of what the N integers are.

Given N integers, find the largest mysterious number with respect to these integers.

Input

The first line will contain one integer T, the number of test cases.

For each test case, there will be two lines of input.

The first line will contain N and the second line will contain N integers.

Output

For each test case, output the largest mysterious number.

If there is an infinite number of mysterious numbers, output "`INFINITY`" (without quotes).

Limits

- $1 \leq N \leq 2,000$
- $-10^9 \leq N$ integers $\leq 10^9$

Examples

standard input	standard output
4 5 2 4 6 8 10 5 162 72 54 63 57 5 -20 -30 -50 80 75 4 5 5 5 5	2 3 5 INFINITY

Notes

For any integer p and any non-zero integer q, the remainder of p modulo q is defined as $(p - a \times q)$ where a is the unique integer that satisfies $0 \leq p - a \times q < q$.

For instance, the remainder is 1 when $p = 7$ and $q = 3$ and the remainder is 2 when $p = -7$ and $q = 3$.

Problem C. Sorting

Input file:	standard input
Output file:	standard output
Time limit:	0.5 sec
Memory limit:	256 MB

For any permutation A of length N ($A = [A_1, A_2, \ldots, A_N]$), we can sort A with any of the following code.

In this problem, a permutation of length N is a sequence of numbers where each number between 1 and N appears exactly once.

```
input: n, a[1 .. n]
cnt = 0
for j = 2 to n:
  x = a[j]
  i = j - 1
  while i >= 1 and a[i] > x:
    cnt = cnt + 1
    a[i+1] = a[i]
    i = i-1
```

Figure 1: Pseudocode

```
int cnt = 0;
for (int j=2; j<=n; j++) {
    int x = a[j];
    int i = j-1;
    while (i >= 1 && a[i] > x) {
        cnt += 1;
        a[i+1] = a[i];
        i -= 1;
    }
    a[i+1] = x;
}
```

Figure 2: C++, Java

```
cnt = 0
for j in range(2, n+1):
    x = a[j]
    i = j-1
    while i >= 1 and a[i] > x:
        cnt += 1
        a[i+1] = a[i]
        i = i-1
    a[i+1] = x
```

Figure 3: Python

When sort a permutation of length N, cnt will always be between 0 and $N \times (N-1)/2$. Given two integers N and M, find a permutation of length N such that if we apply any of the code above the final value of *cnt* will be equal to M.

Input

The first line will contain one integer T, the number of test cases.

Each of the next T lines will contain two integers N and M, separated by a whitespace.

Output

For each test case, output a permutation of length N so that cnt would be equal to M.

If there exist multiple such permutations, you may output any one of them.

Limits

- $1 \leq N \leq 100,000$
- $0 \leq M \leq N \times (N-1)/2$

Examples

standard input	standard output
4 5 0 5 1 5 5 5 10	1 2 3 4 5 2 1 3 4 5 5 2 1 3 4 5 4 3 2 1

Problem D. Alphabet String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 sec
Memory limit:	256 MB

Consider a string S of length N that consists only of uppercase English alphabets.

$S[i]$ represents the i-th character of S, and $S[i:j]$ is the substring of $S[i:j]$는 $S[i], S[i+1], \ldots, S[j-1], S[j]$ (in this problem we use 1-based index).

$U(i,j)$ is the sorted string of unique characters (alphabets) that appear in $S[i:j]$.

For instance, if S = "ABCBA", we would have $U(1,3)$ = "ABC", $U(2,4)$ = "BC", and $U(1,5)$ = "ABC".

We want to know the number of unique stings across all $U(i,j)$ with $1 \le i \le j \le N$.

Input

The first line will contain one integer T, the number of test cases.

Each of the next T lines will contain one string S that will only contain uppercase English alphabets.

Output

For each test case, output the number of unique strings across all possible $U(i,j)$'s.

Limits

- $1 \le T \le 10$
- $1 \le N \le 100,000$

Examples

standard input	standard output
4 AAA ABCBA ABABAB ABCXYZABC	1 6 3 30

Notes

In the second example, there are six unique strings: `A`, `B`, `C`, `AB`, `BC`, `ABC`.

In the third example, there are three unique string: `A`, `B`, `AB`.

Problem E. Street Cleaning

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 sec
Memory limit:	256 MB

There is a country that has N cities and the cities are numbered from 1 to N. There are M bi-directional streets each of which connects two distinct cities, the streets are numbered from 1 to M. There may exist multiple streets between the same pair of cities.

Due to a recent budget constraint, the government has decided to operate only two street cleaning vehicles. Using these two vehicles, you wish to clean each street exactly once.

Each vehicle can start cleaning from any city you like. The two vehicles may start from the same city or different cities. In order to clean each street exactly once, every street must be traversed by exactly one vehicle because a cleaning vehicle cannot traverse the street without cleaning it. Finally, each vehicle must clean at least one road (otherwise we have no reason to keep both vehicles).

Given the city and street information, determine whether we can clean each street exactly once using two cleaning vehicles. If possible, we should also find a way to do so.

Input

The first line will contain one integer T.

For each test case, the first line will contain two integers N (the number of cities) and M (the number of streets). The next two lines will contain M integers each so that the i-th pair of integers represent the two cities connected by the i-th street.

Each test case will always satisfy the following: From any city you can reach any other city using the given streets.

Output

For each test case, you must output two lines.

The first integer in each of the two lines is the number of streets a vehicle must clean, followed by the street numbers. If the vehicle must traverse the i-th street in the reversed order (compared to the original input), then output -i.

If it is impossible to clean every street exactly once, then you must output 0 in each line (hence, two lines with 0 in each line).

Limits

- $1 \leq N \leq 1,000$
- $1 \leq M \leq 50,000$

Examples

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

Problem F. Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 sec
Memory limit:	256 MB

Albert is playing a game with numbers. To play the game, he needs two sheets of paper, one in blue and another in red. At first, both sheets have number "1" written on them.

Albert plays the game as follows. First he calculates the sum of the two numbers written on the sheets. He then replaces the number on one of the sheets by this sum. One such move is called an "addition". After applying addition(s), Albert ends the game when one of the sheets has x written on it (the number he initially chooses to play this game).

For every integer $x > 1$, if Albert chooses to write only on the same color sheet, then after $(x - 1)$ additions the game always ends. Albert wants to know the minimum number of additions he needs to perform in order to reach x.

Input

The first line will contain T, the number of test cases.

Each of the next T lines will contain a single integer x.

Output

For each test case, you must output a string that describes how Albert can reach x with minimum number of additions.

If M is the minimum number of additions Albert must do in order to get x, then the output must be a string of length M. Each character should be either `B` or `R` depending on which colored sheet Albert must write the summed number on. If there are multiple ways to achieve the minimum number of additions, you may output any one of them.

Limits

- $1 \le T \le 100$

Subtask 1 (10 points)

- $2 \le x \le 500$

Subtask 2 (22 points)

- $2 \le x \le 300,000$

Examples

standard input	standard output
4 2 3 4 5	B RB RRR RBR

Notes

When $x = 5$, there are many ways to reach 5.

`B` represents blue sheet and `R` represents red sheet. The number after `B` or `R` represents the number written on the respective sheet.

- (B1, R1) → Albert chooses B → (B2, R1) → Albert chooses R → (B2, R3) → Albert chooses B → (B5, R3) → Game ends (3 additions).
- (B1, R1) → Albert chooses B → (B2, R1) → Albert chooses B → (B3, R1) → Albert chooses B → (B4, R1) → Albert chooses B → (B5, R1) → Game ends (4 additions).