Codeforces Round 1034 (Div. 3)

A. Blackboard Game

1 second, 256 megabytes

Initially, the integers from 0 to $n-1$ are written on a blackboard.

In one round,

- Alice chooses an integer a on the blackboard and erases it;
- then Bob chooses an integer b on the blackboard such that $a+b \equiv 3 \pmod 4$* and erases it.

Rounds take place in succession until a player is unable to make a move — the first player who is unable to make a move loses. Determine who wins with optimal play.

*We define that $x \equiv y \pmod m$ whenever $x-y$ is an integer multiple of m.

Input

The first line contains an integer t $(1 \le t \le 100)$ — the number of test cases.

The only line of each test case contains an integer n $(1 \le n \le 100)$ — the number of integers written on the blackboard.

Output

For each test case, output on a single line "Alice" if Alice wins with optimal play, and "Bob" if Bob wins with optimal play.

You can output the answer in any case (upper or lower). For example, the strings "aLiCe", "alice", "ALICE", and "alICE" will be recognized as "Alice".

input

```
5
2
4
5
7
100
```

output

```
Alice
Bob
Alice
Alice
Bob
```

In the first sample, suppose Alice chooses 0, then Bob cannot choose any number so Alice wins immediately.

In the second sample, suppose Alice chooses 0, then Bob can choose 3. Then suppose Alice chooses 2, then Bob can choose 1. Then Alice has no numbers remaining, so Bob wins.

B. Tournament

2 seconds, 256 megabytes

You are given an array of integers $a_1, a_2, \ldots, a_n$. A tournament is held with n players. Player i has strength a_i.

While more than k players remain,

- Two remaining players are chosen at random;
- Then the chosen player with the lower strength is eliminated. If the chosen players have the same strength, one is eliminated at random.

Given integers j and k ($1 \le j, k \le n$), determine if there is any way for player j to be one of the last k remaining players.

Input

The first line contains an integer t $(1 \le t \le 10^4)$ — the number of test cases.

The first line of each test case contains three integers n, j, and k ($2 \le n \le 2 \cdot 10^5$, $1 \le j, k \le n$).

The second line of each test case contains n integers, $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$).

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output on a single line "YES" if player j can be one of the last k remaining players, and "NO" otherwise.

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

input

```
3
5 2 3
3 2 4 4 1
5 4 1
5 3 4 5 2
6 1 1
1 2 3 4 5 6
```

output

```
YES
YES
NO
```

In the first sample, suppose that players 2 and 5 are chosen. Then player 2 defeats player 5. Now, the remaining player strengths are

3	2	4	4

Next, suppose that players 3 and 4 are chosen. Then player 3 might defeat player 4. Now, the remaining player strengths are

3	2	4

Player 2 is one of the last three players remaining.

In the third sample, it can be shown that there is no way for player 1 to be the last player remaining.

C. Prefix Min and Suffix Max

2 seconds, 256 megabytes

You are given an array a of **distinct** integers.

In one operation, you may either:

- choose a nonempty **prefix*** of a and replace it with its minimum value, or
- choose a nonempty **suffix**† of a and replace it with its maximum value.

Note that you may choose the entire array a.

For each element a_i, determine if there exists some sequence of operations to transform a into $[a_i]$; that is, make the array a consist of only one element, which is a_i. Output your answer as a binary string of length n, where the i-th character is 1 if there exists a sequence to transform a into $[a_i]$, and 0 otherwise.

* A **prefix** of an array is a subarray consisting of the first k elements of the array, for some integer k.

† A **suffix** of an array is a subarray consisting of the last k elements of the array, for some integer k.

Input

The first line contains an integer t $(1 \le t \le 10^4)$ — the number of test cases.

The first line of each test case contains one integer n ($2 \le n \le 2 \cdot 10^5$) — the size of the array a.

The second line of each test case contains n integers, $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$). It is guaranteed that all a_i are distinct.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a binary string of length n — the i-th character should be 1 if there exists a sequence of operations as described above, and 0 otherwise.

input

```
3
6
1 3 5 4 7 2
4
13 10 12 20
7
1 2 3 4 5 6 7
```

output

```
100011
1101
1000001
```

In the first sample, you can first choose the prefix of size 3. Then the array is transformed into

1	4	7	2

Next, you can choose the suffix of size 2. Then the array is transformed into

1	4	7

Finally, you can choose the prefix of size 3. Then the array is transformed into

1

So we see that it is possible to transform a into $[1]$.

It can be shown that it is impossible to transform a into $[3]$.

D. Binary String Battle

2 seconds, 256 megabytes

Alice and Bob are given a binary string s of length n, and an integer k ($1 \le k < n$).

Alice wins if she is able to transform all characters of s into zeroes. If Alice is unable to win in a finite number of moves, then Bob wins.

Alice and Bob take turns, with Alice going first.

- On Alice's turn, she may choose any **subsequence*** of length k in s, then set all characters in that subsequence to zero.
- On Bob's turn, he may choose any **substring**† of length k in s, then set all characters in that substring to one.

Note that Alice wins if the string consists of all zeros at any point during the game, including in between Alice's and Bob's turns.

Determine who wins with optimal play.

* A **subsequence** of a string s is a set of characters in s. Note that these characters do not have to be adjacent.

† A **substring** of a string s is a contiguous group of characters in s. Note that these characters must be adjacent.

Input

The first line contains an integer t $(1 \le t \le 10^4)$ — the number of test cases.

The first line of each test case contains two integers n and k ($2 \le n \le 2 \cdot 10^5, 1 \le k < n$).

The second line of each test case contains a binary string s of length n.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output on a single line "`Alice`" if Alice wins with optimal play, and "`Bob`" if Bob wins with optimal play.

You can output the answer in any case (upper or lower). For example, the strings "`aLiCe`", "`alice`", "`ALICE`", and "`alICE`" will be recognized as "`Alice`".

input

```
6
5 2
11011
7 4
1011011
6 1
010000
4 1
1111
8 3
10110110
6 4
111111
```

output

```
Bob
Alice
Alice
Bob
Bob
Alice
```

In the third sample, Alice can choose the subsequence consisting of s_2, turning s into 000000. Then she wins immediately.

In the fourth sample, it can be shown that there is no way for Alice to guarantee that she can turn s into 0000 within a finite number of moves.

E. MEX Count

3 seconds, 256 megabytes

Define the MEX (minimum excluded value) of an array to be the smallest nonnegative integer not present in the array. For example,

- $\mathrm{MEX}([2,2,1]) = 0$ because 0 is not in the array.
- $\mathrm{MEX}([3,1,0,1]) = 2$ because 0 and 1 are in the array but 2 is not.
- $\mathrm{MEX}([0,3,1,2]) = 4$ because 0, 1, 2, and 3 are in the array but 4 is not.

You are given an array a of size n of nonnegative integers.

For all k ($0 \le k \le n$), count the number of possible values of $\mathrm{MEX}(a)$ after removing exactly k values from a.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the size of the array a.

The second line of each test case contains n integers, $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le n$).

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single line containing $n+1$ integers — the number of possible values of $\mathrm{MEX}(a)$ after removing exactly k values, for $k = 0, 1, \ldots, n$.

input

```
5
5
1 0 0 1 2
6
3 2 0 4 5 1
6
1 2 0 1 3 2
4
0 3 4 1
5
0 0 0 0 0
```

output

```
1 2 4 3 2 1
1 6 5 4 3 2 1
1 3 5 4 3 2 1
1 3 3 2 1
1 1 1 1 1 1
```

In the first sample, consider $k=1$. If you remove a 0, then you get the following array:

1	0	1	2

So we get $\mathrm{MEX}(a) = 3$. Alternatively, if you remove the 2, then you get the following array:

1	0	0	1

So we get $\mathrm{MEX}(a) = 2$. It can be shown that these are the only possible values of $\mathrm{MEX}(a)$ after removing exactly one value. So the output for $k = 1$ is 2.

F. Minimize Fixed Points

3 seconds, 256 megabytes

Call a permutation* p of length n *good* if $\gcd(p_i, i)^\dagger > 1$ for all $2 \le i \le n$. Find a good permutation with the *minimum* number of **fixed points**‡ across all good permutations of length n. If there are multiple such permutations, print any of them.

* A permutation of length n is an array that contains every integer from 1 to n exactly once, in any order.

† $\gcd(x, y)$ denotes the greatest common divisor (GCD) of x and y.

‡ A **fixed point** of a permutation p is an index j ($1 \le j \le n$) such that $p_j = j$.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The only line of each test case contains an integer n ($2 \le n \le 10^5$) — the length of the permutation.

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

Output

For each test case, output on a single line an example of a good permutation of length n with the minimum number of fixed points.

input

```
4
2
3
6
13
```

output

```
1 2
1 2 3
1 4 6 2 5 3
1 12 9 6 10 8 7 4 3 5 11 2 13
```

In the third sample, we construct the permutation

i	p_i	$\gcd(p_i, i)$
1	1	1
2	4	2
3	6	3
4	2	2
5	5	5
6	3	3

Then we see that $\gcd(p_i, i) > 1$ for all $2 \le i \le 6$. Furthermore, we see that there are only two fixed points, namely, 1 and 5. It can be shown that it is impossible to build a good permutation of length 6 with fewer fixed points.

G. Modular Sorting

5 seconds, 256 megabytes

You are given an integer m ($2 \le m \le 5 \cdot 10^5$) and an array a consisting of nonnegative integers smaller than m.

Answer queries of the following form:

- $1\ i\ x$: assign $a_i := x$
- $2\ k$: in one operation, you may choose an element a_i and assign $a_i := (a_i + k) \pmod m$* — determine if there exists some sequence of (possibly zero) operations to make a nondecreasing†.

Note that instances of query 2 are independent; that is, no actual operations are taking place. Instances of query 1 are persistent.

* $a \pmod m$ is defined as the unique integer b such that $0 \le b < m$ and $a - b$ is an integer multiple of m.

† An array a of size n is called nondecreasing if and only if $a_i \le a_{i+1}$ for all $1 \le i < n$.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains three integers, n, m, and q ($2 \le n \le 10^5$, $2 \le m \le 5 \cdot 10^5$, $1 \le q \le 10^5$) — the size of the array a, the integer m, and the number of queries.

The second line of each test case contains n integers, $a_1, a_2, \ldots, a_n$ ($0 \le a_i < m$).

Then follows q lines. Each line is of one of the following forms:

- $1\ i\ x$ $(1 \le i \le n, 0 \le x < m)$
- $2\ k$ $(1 \le k < m)$

It is guaranteed that the sum of n and the sum of q over all test cases each do not exceed 10^5.

Output

For each instance of query 2, output on a single line "YES" if there exists some sequence of (possibly zero) operations to make a nondecreasing, and "NO" otherwise.

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

input

```
2
7 6 6
4 5 2 2 4 1 0
2 4
1 4 5
2 4
2 3
1 7 2
2 3
8 8 3
0 1 2 3 4 5 6 7
2 4
1 3 4
2 4
```

output

```
YES
NO
NO
YES
YES
NO
```

In the first sample, the array is initially:

4	5	2	2	4	1	0

By applying the operation twice on a_1, twice on a_2, once on a_5, twice on a_6, and once on a_7, the array becomes:

0	1	2	2	2	3	4

which is in nondecreasing order.

After the second query, the array becomes:

4	5	2	5	4	1	0

and it can be shown that it is impossible to sort this with operations of the form $a_i := (a_i + 4) \pmod 6$, and it is also impossible to sort this with operations of the form $a_i := (a_i + 3) \pmod 6$.