# A. Closest Point

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

Consider a set of points on a line. The distance between two points $i$ and $j$ is $|i - j|$.

The point $i$ from the set is **the closest** to the point $j$ from the set, if there is no other point $k$ in the set such that the distance from $j$ to $k$ is **strictly less** than the distance from $j$ to $i$. In other words, all other points from the set have distance to $j$ greater or equal to $|i - j|$.

For example, consider a set of points $\{1, 3, 5, 8\}$:

- for the point $1$, the closest point is $3$ (other points have distance greater than $|1 - 3| = 2$);
- for the point $3$, there are two closest points: $1$ and $5$;
- for the point $5$, the closest point is $3$ (but not $8$, since its distance is greater than $|3 - 5|$);
- for the point $8$, the closest point is $5$.

You are given a set of points. You have to add an **integer** point into this set in such a way that **it is different from every existing point in the set**, and **it becomes the closest point to every point in the set**. Is it possible?

## Input

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

Each test case consists of two lines:

- the first line contains one integer $n$ ($2 \le n \le 40$) — the number of points in the set;
- the second line contains $n$ integers $x_1, x_2, \dots, x_n$ ($1 \le x_1 < x_2 < \dots < x_n \le 100$) — the points from the set.

## Output

For each test case, print YES if it is possible to add a new point according to the conditions from the statement. Otherwise, print NO.

| Standard Input | Standard Output |
|---|---|
| 3<br>2<br>3 8<br>2<br>5 6<br>6<br>1 2 3 4 5 10 | YES<br>NO<br>NO |

## Note

In the first example, the point $7$ will be the closest to both $3$ and $8$.

In the second example, it is impossible to add an **integer** point so that it becomes the closest to both $5$ and $6$, and is different from both of them.

,

# B. Game with Doors

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

There are $100$ rooms arranged in a row and $99$ doors between them; the $i$-th door connects rooms $i$ and $i+1$. Each door can be either locked or unlocked. Initially, all doors are unlocked.

We say that room $x$ is reachable from room $y$ if all doors between them are unlocked.

You know that:

- Alice is in some room from the segment $[l, r]$;
- Bob is in some room from the segment $[L, R]$;
- Alice and Bob are in different rooms.

However, you don't know the exact rooms they are in.

You don't want Alice and Bob to be able to reach each other, so you are going to lock some doors to prevent that. What's the smallest number of doors you have to lock so that Alice and Bob cannot meet, regardless of their starting positions inside the given segments?

## Input

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

The first line of each test case contains two integers $l$ and $r$ ($1 \le l < r \le 100$) — the bounds of the segment of rooms where Alice is located.

The second line of each test case contains two integers $L$ and $R$ ($1 \le L < R \le 100$) — the bounds of the segment of rooms where Bob is located.

## Output

For each test case, print a single integer — the smallest number of doors you have to lock so that Alice and Bob cannot meet, regardless of their starting positions inside the given segments.

| Standard Input | Standard Output |
|---|---|
| 4<br>1 2<br>3 4<br>2 5<br>2 5<br>3 7<br>6 7<br>4 5<br>2 8 | 1<br>3<br>2<br>3 |

## Note

In the first test case, it is sufficient to lock the door between rooms $2$ and $3$.

In the second test case, the following doors have to be locked: $(2, 3)$, $(3, 4)$, $(4, 5)$.

In the third test case, the following doors have to be locked: $(5, 6)$ and $(6, 7)$.

,

# C. Splitting Items

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

Alice and Bob have $n$ items they'd like to split between them, so they decided to play a game. All items have a cost, and the $i$-th item costs $a_i$. Players move in turns starting from Alice.

In each turn, the player chooses one of the remaining items and takes it. The game goes on until no items are left.

Let's say that $A$ is the total cost of items taken by Alice and $B$ is the total cost of Bob's items. The resulting *score* of the game then will be equal to $A - B$.

Alice wants to maximize the score, while Bob wants to minimize it. Both Alice and Bob will play optimally.

But the game will take place tomorrow, so today Bob can modify the costs a little. He can increase the costs $a_i$ of several (possibly none or all) items by an integer value (possibly, by the same value or by different values for each item). However, the total increase must be less than or equal to $k$. Otherwise, Alice may suspect something. Note that Bob **can't decrease** costs, only increase.

What is the minimum possible score Bob can achieve?

## Input

The first line contains a single integer $t$ ($1 \le t \le 5000$) — the number of test cases. Then $t$ cases follow.

The first line of each test case contains two integers $n$ and $k$ ($2 \le n \le 2 \cdot 10^5$; $0 \le k \le 10^9$) — the number of items and the maximum total increase Bob can make.

The second line of each test case contains $n$ integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the initial costs of the items.

It's guaranteed that the sum of $n$ over all test cases doesn't exceed $2 \cdot 10^5$.

## Output

For each test case, print a single integer — the minimum possible score $A - B$ after Bob increases the costs of several (possibly none or all) items.

| Standard Input | Standard Output |
|---|---|
| 4<br>2 5<br>1 10<br>3 0<br>10 15 12<br>4 6<br>3 1 2 4<br>2 4<br>6 9 | 4<br>13<br>0<br>0 |

## Note

In the first test case, Bob can increase $a_1$ by $5$, making costs equal to $[6, 10]$. Tomorrow, Alice will take $10$ and Bob will take $6$. The total score will be equal to $10 - 6 = 4$, and it's the minimum possible.

In the second test case, Bob can't change costs. So the score will be equal to $(15 + 10) - 12 = 13$, since Alice will take $15$, Bob will take $12$, and Alice — $10$.

In the third test case, Bob, for example, can increase $a_1$ by $1$, $a_2$ by $3$, and $a_3$ by $2$. The total change is equal to $1 + 3 + 2 \le 6$ and costs will be equal to $[4, 4, 4, 4]$. Obviously, the score will be equal to $(4 + 4) - (4 + 4) = 0$.

In the fourth test case, Bob can increase $a_1$ by $3$, making costs equal to $[9, 9]$. The score will be equal to $9 - 9 = 0$.

,

# D. Colored Portals

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

There are $n$ cities located on a straight line. The cities are numbered from $1$ to $n$.

Portals are used to move between cities. There are $4$ colors of portals: blue, green, red, and yellow. Each city has portals of two different colors. You can move from city $i$ to city $j$ if they have portals of the same color (for example, you can move between a "blue-red" city and a "blue-green" city). This movement costs $|i - j|$ coins.

Your task is to answer $q$ independent queries: calculate the minimum cost to move from city $x$ to city $y$.

## Input

The first line contains a single 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 $q$ ($1 \le n, q \le 2 \cdot 10^5$) — the number of cities and the number of queries, respectively.

The second line contains $n$ strings of the following types: BG, BR, BY, GR, GY, or RY; the $i$-th of them describes the portals located in the $i$-th city; the symbol B indicates that there is a blue portal in the city, G — green, R — red, and Y — yellow.

The $j$-th of the next $q$ lines contains two integers $x_j$ and $y_j$ ($1 \le x_j, y_j \le n$) — the description of the $j$-th query.

Additional constraints on the input:

- the sum of $n$ over all test cases does not exceed $2 \cdot 10^5$;
- the sum of $q$ over all test cases does not exceed $2 \cdot 10^5$.

## Output

For each query, print a single integer — the minimum cost to move from city $x$ to city $y$ (or $-1$ if it is impossible).

| Standard Input | Standard Output |
|---|---|
| 2<br>4 5<br>BR BR GY GR<br>1 2<br>3 1<br>4 4<br>1 4<br>4 2<br>2 1<br>BG RY<br>1 2 | 1<br>4<br>0<br>3<br>2<br>-1 |

,

# E. Not a Nim Problem

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

Two players, Alice and Bob, are playing a game. They have $n$ piles of stones, with the $i$-th pile initially containing $a_i$ stones.

On their turn, a player can choose any pile of stones and take any positive number of stones from it, with one condition:

- let the current number of stones in the pile be $x$. It is not allowed to take from the pile a number of stones $y$ such that the greatest common divisor of $x$ and $y$ is not equal to $1$.

The player who cannot make a move loses. Both players play optimally (that is, if a player has a strategy that allows them to win, no matter how the opponent responds, they will win). Alice goes first.

Determine who will win.

### Input

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

Each test case consists of two lines:

- the first line contains a single integer $n$ ($1 \le n \le 3 \cdot 10^5$);
- the second line contains $n$ integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^7$).

Additional constraint on the input: the sum of $n$ across all test cases does not exceed $3 \cdot 10^5$.

### Output

For each test case, output `Alice` if Alice wins, or `Bob` if Bob wins.

| Standard Input | Standard Output |
|---|---|
| 3<br>3<br>3 2 9<br>4<br>3 3 6 1<br>5<br>1 2 3 4 5 | Bob<br>Alice<br>Bob |

,

# F. Make a Palindrome

```
Input file:     standard input
Output file:    standard output
Time limit:     5 seconds
Memory limit:   512 megabytes
```

You are given an array $a$ consisting of $n$ integers.

Let the function $f(b)$ return the minimum number of operations needed to make an array $b$ a palindrome. The operations you can make are:

- choose two adjacent elements $b_i$ and $b_{i+1}$, remove them, and replace them with a single element equal to $(b_i + b_{i+1})$;
- or choose an element $b_i > 1$, remove it, and replace it with two positive integers $x$ and $y$ ($x > 0$ and $y > 0$) such that $x + y = b_i$.

For example, from an array $b = [2, 1, 3]$, you can obtain the following arrays in one operation: $[1, 1, 1, 3]$, $[2, 1, 1, 2]$, $[3, 3]$, $[2, 4]$, or $[2, 1, 2, 1]$.

Calculate $\left(\sum_{1 \le l \le r \le n} f(a[l..r])\right)$, where $a[l..r]$ is the subarray of $a$ from index $l$ to index $r$, inclusive. In other words, find the sum of the values of the function $f$ for all subarrays of the array $a$.

## Input

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

The first line of each test case contains a single integer $n$ ($1 \le n \le 2000$).

The second line contains $n$ integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^5$).

Additional constraint on the input: the sum of $n$ over all test cases does not exceed $2000$.

## Output

For each test case, print a single integer — the sum of the values of the function $f$ for all subarrays of the array $a$.

| Standard Input | Standard Output |
|---|---|
| 4<br>3<br>2 1 3<br>4<br>1 1 1 1<br>5<br>4 2 3 1 5<br>4<br>1 2 1 2 | 3<br>0<br>14<br>5 |

,

# G. Substring Compression

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   1024 megabytes
```

Let's define the operation of compressing a string $t$, consisting of at least $2$ digits from $1$ to $9$, as follows:

- split it into an **even** number of non-empty substrings — let these substrings be $t_1, t_2, \ldots, t_m$ (so, $t = t_1 + t_2 + \cdots + t_m$, where $+$ is the concatenation operation);
- write the string $t_2$ $t_1$ times, then the string $t_4$ $t_3$ times, and so on.

For example, for a string "12345", one could do the following: split it into ("1", "23", "4", "5"), and write "235555".

Let the function $f(t)$ for a string $t$ return the minimum length of the string that can be obtained as a result of that process.

You are given a string $s$, consisting of $n$ digits from $1$ to $9$, and an integer $k$. Calculate the value of the function $f$ for all contiguous substrings of $s$ of length exactly $k$.

### Input

The first line contains two integers $n$ and $k$ ($2 \le k \le n \le 2 \cdot 10^5$).

The second line contains the string $s$ ($|s| = n$), consisting only of digits from $1$ to $9$.

### Output

Output $n - k + 1$ integers — $f(s_{1,k}), f(s_{2,k+1}), \ldots, f(s_{n-k+1,n})$.

| Standard Input | Standard Output |
|---|---|
| 4 4<br>5999 | 14 |
| 10 3<br>1111111111 | 2 2 2 2 2 2 2 2 |
| 11 4<br>49998641312 | 12 18 17 15 12 7 7 2 |