# Codeforces Round #783 (Div. 2)

## A. Direction Change

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

You are given a grid with $n$ rows and $m$ columns. Rows and columns are numbered from $1$ to $n$, and from $1$ to $m$. The intersection of the $a$-th row and $b$-th column is denoted by $(a, b)$.

Initially, you are standing in the top left corner $(1, 1)$. Your goal is to reach the bottom right corner $(n, m)$.

You can move in four directions from $(a, b)$: up to $(a - 1, b)$, down to $(a + 1, b)$, left to $(a, b - 1)$ or right to $(a, b + 1)$.

You cannot move in the same direction in two consecutive moves, and you cannot leave the grid. What is the minimum number of moves to reach $(n, m)$?

### Input

The input consists of multiple test cases. The first line contains a single integer $t$ ($1 \le t \le 10^3$) — the number of the test cases. The description of the test cases follows.

The first line of each test case contains two integers $n$ and $m$ ($1 \le n, m \le 10^9$) — the size of the grid.

### Output

For each test case, print a single integer: $-1$ if it is impossible to reach $(n, m)$ under the given conditions, otherwise the minimum number of moves.

**Example**

| input |
|---|
| 6<br>1 1<br>2 1<br>1 3<br>4 2<br>4 6<br>10 5 |
| **output** |
| 0<br>1<br>-1<br>6<br>10<br>17 |

### Note

Test case $1$: $n = 1$, $m = 1$, and initially you are standing in $(1, 1)$ so $0$ move is required to reach $(n, m) = (1, 1)$.

Test case $2$: you should go down to reach $(2, 1)$.

Test case $3$: it is impossible to reach $(1, 3)$ without moving right two consecutive times, or without leaving the grid.

Test case $4$: an optimal moving sequence could be: $(1, 1) \to (1, 2) \to (2, 2) \to (2, 1) \to (3, 1) \to (3, 2) \to (4, 2)$. It can be proved that this is the optimal solution. So the answer is $6$.

## B. Social Distance

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

$m$ chairs are arranged in a circle sequentially. The chairs are numbered from $0$ to $m - 1$. $n$ people want to sit in these chairs. The $i$-th of them wants at least $a[i]$ empty chairs both on his right and left side.

More formally, if the $i$-th person sits in the $j$-th chair, then no one else should sit in the following chairs: $(j - a[i]) \bmod m$, $(j - a[i] + 1) \bmod m$, ... $(j + a[i] - 1) \bmod m$, $(j + a[i]) \bmod m$.

Decide if it is possible to sit down for all of them, under the given limitations.

### Input
The input consists of multiple test cases. The first line contains a single integer $t$ ($1 \le t \le 5 \cdot 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains two integers $n$ and $m$ ($2 \le n \le 10^5$, $1 \le m \le 10^9$) — the number of people and the number of chairs.

The next line contains $n$ integers, $a_1$, $a_2$, ... $a_n$ ($1 \le a_i \le 10^9$) — the minimum number of empty chairs, on both sides of the $i$-th person.

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

### Output
For each test case print "YES" (without quotes) if it is possible for everyone to sit down and fulfil the restrictions, and "NO" (without quotes) otherwise.

You may print every letter in any case you want (so, for example, the strings "yEs", "yes", "Yes" and "YES" will all be recognized as positive answers).

**Example**

| input |
| --- |
| 6<br>3 2<br>1 1 1<br>2 4<br>1 1<br>2 5<br>2 1<br>3 8<br>1 2 1<br>4 12<br>1 2 1 3<br>4 19<br>1 2 1 3 |
| **output** |
| NO<br>YES<br>NO<br>YES<br>NO<br>YES |

### Note
Test case $1$: $n > m$, so they can not sit down.

Test case $2$: the first person can sit $2$-nd and the second person can sit in the $0$-th chair. Both of them want at least $1$ empty chair on both sides, chairs $1$ and $3$ are free, so this is a good solution.

Test case $3$: if the second person sits down somewhere, he needs $2$ empty chairs, both on his right and on his left side, so it is impossible to find a place for the first person, because there are only $5$ chairs.

Test case $4$: they can sit in the $1$-st, $4$-th, $7$-th chairs respectively.

## C. Make it Increasing

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

You are given an array $a$ consisting of $n$ positive integers, and an array $b$, with length $n$. Initially $b_i = 0$ for each $1 \le i \le n$.

In one move you can choose an integer $i$ ($1 \le i \le n$), and add $a_i$ to $b_i$ or subtract $a_i$ from $b_i$. What is the minimum number of moves needed to make $b$ increasing (that is, every element is strictly greater than every element before it)?

### Input
The first line contains a single integer $n$ ($2 \le n \le 5000$).

The second line contains $n$ integers, $a_1$, $a_2$, ..., $a_n$ ($1 \le a_i \le 10^9$) — the elements of the array $a$.

### Output
Print a single integer, the minimum number of moves to make $b$ increasing.

**Examples**

| input |
| --- |
| 5<br>1 2 3 4 5 |
| **output** |

4

| input |
| --- |
| 7<br>1 2 1 2 1 2 1 |
| output |
| 10 |

| input |
| --- |
| 8<br>1 8 2 7 3 6 4 5 |
| output |
| 16 |

### Note

Example $1$: you can subtract $a_1$ from $b_1$, and add $a_3$, $a_4$, and $a_5$ to $b_3$, $b_4$, and $b_5$ respectively. The final array will be $[-1, 0, 3, 4, 5]$ after $4$ moves.

Example $2$: you can reach $[-3, -2, -1, 0, 1, 2, 3]$ in $10$ moves.

## D. Optimal Partition

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

You are given an array $a$ consisting of $n$ integers. You should divide $a$ into continuous non-empty subarrays (there are $2^{n-1}$ ways to do that).

Let $s = a_l + a_{l+1} + \ldots + a_r$. The value of a subarray $a_l, a_{l+1}, \ldots, a_r$ is:

- $(r - l + 1)$ if $s > 0$,
- $0$ if $s = 0$,
- $-(r - l + 1)$ if $s < 0$.

What is the maximum sum of values you can get with a partition?

### Input

The input consists of multiple test cases. The first line contains a single integer $t$ ($1 \le t \le 5 \cdot 10^5$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer $n$ ($1 \le n \le 5 \cdot 10^5$).

The second line of each test case contains $n$ integers $a_1$, $a_2$, ..., $a_n$ ($-10^9 \le a_i \le 10^9$).

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

### Output

For each test case print a single integer — the maximum sum of values you can get with an optimal parition.

#### Example

| input |
| --- |
| 5<br>3<br>1 2 -3<br>4<br>0 -2 3 -4<br>5<br>-1 -2 3 -1 -1<br>6<br>-1 2 -3 4 -5 6<br>7<br>1 -1 -1 1 -1 -1 1 |
| output |
| 1<br>2<br>1<br>6<br>-1 |

### Note

Test case $1$: one optimal partition is $[1, 2]$, $[-3]$. $1 + 2 > 0$ so the value of $[1, 2]$ is $2$. $-3 < 0$, so the value of $[-3]$ is $-1$. $2 + (-1) = 1$.

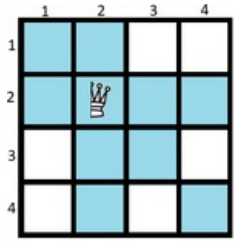
Test case $2$: the optimal partition is $[0, -2, 3]$, $[-4]$, and the sum of values is $3 + (-1) = 2$.

## E. Half Queen Cover

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

You are given a board with $n$ rows and $n$ columns, numbered from $1$ to $n$. The intersection of the $a$-th row and $b$-th column is denoted by $(a, b)$.

A half-queen attacks cells in the same row, same column, and on one diagonal. More formally, a half-queen on $(a, b)$ attacks the cell $(c, d)$ if $a = c$ or $b = d$ or $a - b = c - d$.



The blue cells are under attack.

What is the minimum number of half-queens that can be placed on that board so as to ensure that each square is attacked by at least one half-queen?
Construct an optimal solution.

### Input

The first line contains a single integer $n$ ($1 \le n \le 10^5$) — the size of the board.

### Output

In the first line print a single integer $k$ — the minimum number of half-queens.

In each of the next $k$ lines print two integers $a_i$, $b_i$ ($1 \le a_i, b_i \le n$) — the position of the $i$-th half-queen.

If there are multiple solutions, print any.

**Examples**

| input |
| --- |
| 1 |
| **output** |
| 1<br>1 1 |

| input |
| --- |
| 2 |
| **output** |
| 1<br>1 1 |

| input |
| --- |
| 3 |
| **output** |
| 2<br>1 1<br>1 2 |

### Note

Example $1$: one half-queen is enough. Note: a half-queen on $(1, 1)$ attacks $(1, 1)$.

Example $2$: one half-queen is enough too. $(1, 2)$ or $(2, 1)$ would be wrong solutions, because a half-queen on $(1, 2)$ does not attack the cell $(2, 1)$ and vice versa. $(2, 2)$ is also a valid solution.

Example $3$: it is impossible to cover the board with one half queen. There are multiple solutions for $2$ half-queens; you can print any of them.

## F. Edge Elimination

time limit per test: 2 seconds
memory limit per test: 256 megabytes

input: standard input
output: standard output

You are given a tree (connected, undirected, acyclic graph) with $n$ vertices. Two edges are adjacent if they share exactly one endpoint. In one move you can remove an arbitrary edge, if that edge is adjacent to an even number of remaining edges.

Remove all of the edges, or determine that it is impossible. If there are multiple solutions, print any.

## Input

The input consists of multiple test cases. The first line contains a single integer $t$ ($1 \le t \le 10^5$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer $n$ ($2 \le n \le 2 \cdot 10^5$) — the number of vertices in the tree.

Then $n - 1$ lines follow. The $i$-th of them contains two integers $u_i$, $v_i$ ($1 \le u_i, v_i \le n$) the endpoints of the $i$-th edge. It is guaranteed that the given graph is a tree.

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

## Output

For each test case print "NO" if it is impossible to remove all the edges.

Otherwise print "YES", and in the next $n - 1$ lines print a possible order of the removed edges. For each edge, print its endpoints in any order.

**Example**

| input |
| --- |
| 5<br>2<br>1 2<br>3<br>1 2<br>2 3<br>4<br>1 2<br>2 3<br>3 4<br>5<br>1 2<br>2 3<br>3 4<br>3 5<br>7<br>1 2<br>1 3<br>2 4<br>2 5<br>3 6<br>3 7 |
| **output** |
| YES<br>2 1<br>NO<br>YES<br>2 3<br>3 4<br>2 1<br>YES<br>3 5<br>2 3<br>2 1<br>4 3<br>NO |

## Note

Test case $1$: it is possible to remove the edge, because it is not adjacent to any other edge.

Test case $2$: both edges are adjacent to exactly one edge, so it is impossible to remove any of them. So the answer is "NO".

Test case $3$: the edge $2 - 3$ is adjacent to two other edges. So it is possible to remove it. After that removal it is possible to remove the remaining edges too.