A - Timeout

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 150 points

Problem Statement

The elder of Takahashi Village falls asleep immediately. Specifically, if $S + 0.5$ seconds or more have passed since the elder was last tapped on the shoulder, the elder falls asleep.

Currently, the elder is awake, and an attendant has just tapped the elder on the shoulder.

From now on, the attendant will tap the elder's shoulder exactly N times. The i-th shoulder tap will be performed T_i seconds from now.

Determine whether the elder remains awake continuously from now until T_N seconds later.

Constraints

- $1 \leq N \leq 100$
- $1 \leq S \leq 100$
- $1 \leq T_i \leq 1000$ $(1 \leq i \leq N)$
- $T_i < T_{i+1}$ $(1 \leq i \leq N-1)$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

```
N S
T_1 T_2 ... T_N
```

Output

If the elder remains awake continuously from now until T_N seconds later, output `Yes`; otherwise, output `No`.

Sample Input 1

```
5 10
6 11 21 22 30
```

Sample Output 1

```
Yes
```

The attendant taps the shoulder in chronological order as follows:

- Taps after 6 seconds. At this time, only 6 seconds have passed since the elder was last tapped on the shoulder, so the elder is awake.
- Taps after 11 seconds. At this time, only 5 seconds have passed since the elder was last tapped on the shoulder, so the elder is awake.
- Taps after 21 seconds. At this time, only 10 seconds have passed since the elder was last tapped on the shoulder, so the elder is awake.
- Taps after 22 seconds. At this time, only 1 second has passed since the elder was last tapped on the shoulder, so the elder is awake.
- Taps after 30 seconds. At this time, only 8 seconds have passed since the elder was last tapped on the shoulder, so the elder is awake.

Since the elder is awake from now until 30 seconds later, output `Yes`.

Sample Input 2

```
2 100
1 200
```

Sample Output 2

```
No
```

The elder falls asleep 101.5 seconds from now. Therefore, output `No`.

Sample Input 3

```
10 22
47 81 82 95 117 146 165 209 212 215
```

Sample Output 3

```
No
```

B - Compression

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 150 points

Problem Statement

An integer sequence $A = (A_1, A_2, \ldots, A_N)$ of length N is given.

Output the numbers contained in A in ascending order, removing duplicates.

Constraints

- $1 \leq N \leq 100$
- $1 \leq A_i \leq 100$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$\begin{array}{l} N \\ A_1 \quad A_2 \quad \ldots \quad A_N \end{array}$$

Output

Let $C_1, C_2, \ldots, C_M$ be the numbers contained in A in ascending order. Output in the following format:

$$\begin{array}{l} M \\ C_1 \quad C_2 \quad \ldots \quad C_M \end{array}$$

Sample Input 1

```
4
3 1 4 1
```

Sample Output 1

```
3
1 3 4
```

The numbers contained in $A = (3, 1, 4, 1)$ are $1, 3, 4$ in ascending order, totalling 3 distinct numbers. Therefore, output as shown above.

Sample Input 2

```
3
7 7 7
```

Sample Output 2

```
1
7
```

Sample Input 3

```
8
19 5 5 19 5 19 4 19
```

Sample Output 3

```
3
4 5 19
```

C - Not All Covered

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 300 points

Problem Statement

In the AtCoder Kingdom, there are N castle walls numbered from 1 through N. There are also M turrets.

Turret i guards castle walls numbered from L_i through R_i.

When a turret is destroyed, the castle walls that were guarded by that turret are no longer guarded by that turret.

What is the minimum number of turrets that need to be destroyed so that at least one castle wall is not guarded by any turret?

Constraints

- $1 \leq N \leq 10^6$
- $1 \leq M \leq 2 \times 10^5$
- $1 \leq L_i \leq R_i \leq N \, (1 \leq i \leq M)$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$\begin{array}{l} N \quad M \\ L_1 \quad R_1 \\ L_2 \quad R_2 \\ \vdots \\ L_M \quad R_M \end{array}$$

Output

Output the minimum number of turrets that need to be destroyed so that at least one castle wall is not guarded by any turret.

Sample Input 1

```
10 4
1 6
4 5
5 10
7 10
```

Sample Output 1

```
1
```

If turret 1 is destroyed, no turret guards castle wall 3. Also, if no turrets are destroyed, all castle walls are guarded by some turret. Therefore, output 1.

Sample Input 2

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

Sample Output 2

```
0
```

Since no turret guards castle wall 5, there already exists a castle wall not guarded by any turret without destroying any turrets. Therefore, output 0.

Sample Input 3

```
5 10
2 5
1 5
1 2
2 4
2 2
5 5
2 4
1 2
2 2
2 3
```

Sample Output 3

```
3
```

D - Flip to Gather

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 400 points

Problem Statement

You are given a string S of length N consisting of `0` and `1`.

You can perform the following operation any number of times (possibly zero):

- Choose an integer i satisfying $1 \leq i \leq N$, and change S_i from `0` to `1`, or from `1` to `0`.

Your goal is to make the `1`s in S form **at most one** interval. Find the minimum number of operations required to achieve the goal.

More precisely, the goal is to make S such that there exists a pair of integers (l, r) satisfying both of the following conditions. Find the minimum number of operations required to achieve the goal.

- $1 \leq l \leq r \leq N+1$.
- $S_i =$ `1` and $l \leq i < r$ are equivalent for each integer i satisfying $1 \leq i \leq N$.

It can be proved that the goal can always be achieved with a finite number of operations.

T test cases are given, so solve each.

Constraints

- $1 \leq T \leq 20000$
- $1 \leq N \leq 2 \times 10^5$
- S is a string of length N consisting of `0` and `1`.
- For each input file, the sum of N over all test cases is at most 2×10^5.
- T, N are integers.

Input

The input is given from Standard Input in the following format:

$$
\begin{array}{l}
T \\
\mathrm{case}_1 \\
\mathrm{case}_2 \\
\vdots \\
\mathrm{case}_T
\end{array}
$$

case_i represents the i-th test case and is given in the following format:

$$
\begin{array}{l}
N \\
S
\end{array}
$$

Output

Output T lines. The i-th line $(1 \leq i \leq T)$ should contain the answer for the i-th test case.

Sample Input 1

```
3
5
10011
10
1111111111
7
0000000
```

Sample Output 1

```
1
0
0
```

In the first test case, if we perform the operation to change S_1 to `0`, the `1`s will form one interval. Also, the initial S does not satisfy the condition. Therefore, the answer is 1.

In the second test case, there are no `0`s in S, so no operations need to be performed. Therefore, the answer is 0.

In the third test case, there are no `1`s in S, so no operations need to be performed. Therefore, the answer is 0.

Sample Input 2

```
5
2
01
10
1000010011
12
111100010011
3
111
8
00010101
```

Sample Output 2

```
0
2
3
0
2
```

E - Minimum OR Path

Time Limit: 3 sec / Memory Limit: 1024 MiB

Score : 450 points

Problem Statement

You are given a connected undirected graph with N vertices and M edges without self-loops, where vertices are numbered from 1 to N and edges are numbered from 1 to M. Edge i connects vertices u_i and v_i bidirectionally and has a label w_i.

Among the simple paths (paths that do not visit the same vertex more than once) from vertex 1 to vertex N, find the minimum possible value of the bitwise OR of all labels on edges included in the path.

▶ What is bitwise OR operation?

Constraints

- $2 \le N \le 2 \times 10^5$
- $N-1 \le M \le 2 \times 10^5$
- $1 \le u_i < v_i \le N$
- $0 \le w_i < 2^{30}$
- The given graph is connected.
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$\begin{array}{lll} N & M & \\ u_1 & v_1 & w_1 \\ u_2 & v_2 & w_2 \\ \vdots & & \\ u_M & v_M & w_M \end{array}$$

Output

Output the answer.

Sample Input 1

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

Sample Output 1

```
3
```

By traversing edges $1, 3, 5$ in order and visiting vertices $1, 2, 3, 4$ in order, the total bitwise OR is $1 \ \mathrm{OR} \ 2 \ \mathrm{OR} \ 3 = 3$.

It is impossible to make the total bitwise OR smaller than 3, so output 3.

Sample Input 2

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

Sample Output 2

```
4
```

The graph may contain multi-edges.

Sample Input 3

```
8 12
4 5 16691344
5 7 129642441
2 7 789275447
3 8 335307651
3 5 530163333
5 6 811293773
3 8 333712701
1 2 2909941
2 3 160265478
5 7 465414272
1 3 903373004
6 7 408299562
```

Sample Output 3

```
468549631
```

F - Athletic

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 500 points

Problem Statement

There are N scaffolds numbered from 1 to N arranged in a line. The height of scaffold $i(1 \leq i \leq N)$ is H_i.

Takahashi decides to play a game of moving on the scaffolds. Initially, he freely chooses an integer $i(1 \leq i \leq N)$ and gets on scaffold i.

When he is on scaffold i at some point, he can choose an integer $j(1 \leq j \leq N)$ satisfying the following condition and move to scaffold j:

- $H_j \leq H_i - D$ and $1 \leq |i-j| \leq R$.

Find the maximum number of moves he can make when he repeats moving until he can no longer move.

Constraints

- $1 \leq N \leq 5 \times 10^5$
- $1 \leq D, R \leq N$
- H is a permutation of $(1, 2, \ldots, N)$.
- All input values are integers.

Input

The input is given from Standard Input in the following format:

```
N D R
H_1 H_2 ... H_N
```

Output

Output the answer.

Sample Input 1

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

Sample Output 1

```
2
```

Takahashi initially gets on scaffold 1 and can move between the scaffolds as follows:

- First move: Since $H_2 \leq H_1 - D$ and $|2-1| \leq R$, he can move to scaffold 2. Move from scaffold 1 to scaffold 2.
- Second move: Since $H_3 \leq H_2 - D$ and $|3-2| \leq R$, he can move to scaffold 3. Move from scaffold 2 to scaffold 3.
- Since the height of scaffold 3 is 1, he can no longer move.

As shown above, he can move 2 times. Also, no matter how he chooses the scaffolds to move to, he cannot move 3 or more times. Therefore, output 2.

Sample Input 2

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

Sample Output 2

```
3
```

G - A/B < p/q < C/D

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 625 points

Problem Statement

You are given positive integers A, B, C, D satisfying $\frac{A}{B} < \frac{C}{D}$.

Find the smallest positive integer q satisfying the following condition:

- There exists a positive integer p such that $\frac{A}{B} < \frac{p}{q} < \frac{C}{D}$.

T test cases are given, so solve each.

Constraints

- $1 \leq T \leq 2 \times 10^5$
- $1 \leq A, B, C, D \leq 10^{18}$
- $\frac{A}{B} < \frac{C}{D}$
- All input values are integers.

Input

The input is given from Standard Input in the following format:

$$\begin{array}{l} T \\ \mathrm{case}_1 \\ \mathrm{case}_2 \\ \vdots \\ \mathrm{case}_T \end{array}$$

Here, case_i represents the i-th test case.

Each test case is given in the following format:

$$A \quad B \quad C \quad D$$

Output

Output T lines. The i-th line should contain the answer for the i-th test case.

Sample Input 1

```
5
3 2 2 1
5 2 8 3
1 2 2 1
60 191 11 35
40 191 71 226
```

Sample Output 1

```
3
5
1
226
4
```

Consider the first test case.

For example, if $p = 5, q = 3$, then $\frac{3}{2} < \frac{5}{3} < \frac{2}{1}$, so $q = 3$ satisfies the condition.

There is no positive integer less than 3 that satisfies the condition for q, so the answer for the first test case is 3.