Codeforces Round #807 (Div. 2)

A. Mark the Photographer

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

Mark is asked to take a group photo of $2n$ people. The i-th person has height h_i units.

To do so, he ordered these people into two rows, the front row and the back row, each consisting of n people. However, to ensure that everyone is seen properly, the j-th person of the back row must be at least x units taller than the j-th person of the front row for each j between 1 and n, inclusive.

Help Mark determine if this is possible.

Input

The first line contains one integer t ($1 \le t \le 100$) — the number of test cases. Each test case consists of two lines.

The first line of each test case contains two positive integers n and x ($1 \le n \le 100$, $1 \le x \le 10^3$) — the number of people in each row and the minimum difference Mark wants.

The second line of each test case contains $2n$ positive integers $h_1, h_2, \ldots, h_{2n}$ ($1 \le h_i \le 10^3$) — the height of each person in units.

Note that the sum of n over all test cases is not bounded.

Output

For each test case, print a single line containing "YES" if Mark could arrange people satisfying his condition and "NO" otherwise.

You may print each letter in any case (for example, YES, Yes, yes, yEs will all be recognized as positive answers).

Example

input

```
3
3 6
1 3 9 10 12 16
3 1
2 5 2 2 2 5
1 2
8 6
```

output

```
YES
NO
YES
```

Note

In the first test case, one possible order is to have the third, fifth, and sixth person on the back row and the second, first, and fourth on the front row. The heights of the people will look like this.

Back	9	12	16
Front	3	1	10

It works because

- $h_3 - h_2 = 9 - 3 \ge 6$,
- $h_5 - h_1 = 12 - 1 \ge 6$, and
- $h_6 - h_4 = 16 - 10 \ge 6$.

In the second test case, it can be shown there is no way to order people in a way that satisfies the condition.

In the third test case, the only way to arrange people to satisfy the condition is to have the first person on the back row and the second person on the front row.

B. Mark the Dust Sweeper

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

output: standard output

Mark is cleaning a row of n rooms. The i-th room has a nonnegative dust level a_i. He has a magical cleaning machine that can do the following three-step operation.

- Select two indices $i < j$ such that the dust levels a_i, a_{i+1}, $\ldots$, a_{j-1} are all strictly greater than 0.
- Set a_i to $a_i - 1$.
- Set a_j to $a_j + 1$.

Mark's goal is to make $a_1 = a_2 = \ldots = a_{n-1} = 0$ so that he can nicely sweep the n-th room. Determine the minimum number of operations needed to reach his goal.

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 a single integer n ($2 \le n \le 2 \cdot 10^5$) — the number of rooms.

The second line of each test case contains n integers a_1, a_2, ..., a_n ($0 \le a_i \le 10^9$) — the dust level of each room.

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

Output

For each test case, print a line containing a single integer — the minimum number of operations. It can be proven that there is a sequence of operations that meets the goal.

Example

input

```
4
3
2 0 0
5
0 2 0 2 0
6
2 0 3 0 4 6
4
0 0 0 10
```

output

```
3
5
11
0
```

Note

In the first case, one possible sequence of operations is as follows.

- Choose $i = 1$ and $j = 2$, yielding the array $[1, 1, 0]$.
- Choose $i = 1$ and $j = 3$, yielding the array $[0, 1, 1]$.
- Choose $i = 2$ and $j = 3$, yielding the array $[0, 0, 2]$.

At this point, $a_1 = a_2 = 0$, completing the process.
In the second case, one possible sequence of operations is as follows.

- Choose $i = 4$ and $j = 5$, yielding the array $[0, 2, 0, 1, 1]$.
- Choose $i = 2$ and $j = 3$, yielding the array $[0, 1, 1, 1, 1]$.
- Choose $i = 2$ and $j = 5$, yielding the array $[0, 0, 1, 1, 2]$.
- Choose $i = 3$ and $j = 5$, yielding the array $[0, 0, 0, 1, 3]$.
- Choose $i = 4$ and $j = 5$, yielding the array $[0, 0, 0, 0, 4]$.

In the last case, the array already satisfies the condition.

C. Mark and His Unfinished Essay

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

One night, Mark realized that there is an essay due tomorrow. He hasn't written anything yet, so Mark decided to randomly copy-paste substrings from the prompt to make the essay.

More formally, the prompt is a string s of initial length n. Mark will perform the copy-pasting operation c times. Each operation is described by two integers l and r, which means that Mark will append letters $s_l s_{l+1} \ldots s_r$ to the end of string s. Note that the length of s increases after this operation.

Of course, Mark needs to be able to see what has been written. After copying, Mark will ask q queries: given an integer k, determine the k-th letter of the final string s.

Input

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

The first line of each test case contains three integers n, c, and q $(1 \leq n \leq 2 \cdot 10^5$, $1 \leq c \leq 40$, and $1 \leq q \leq 10^4)$ — the length of the initial string s, the number of copy-pasting operations, and the number of queries, respectively.

The second line of each test case contains a single string s of length n. It is guaranteed that s only contains lowercase English letters.

The following c lines describe the copy-pasting operation. Each line contains two integers l and r $(1 \leq l \leq r \leq 10^{18})$. It is also guaranteed that r does not exceed the current length of s.

The last q lines of each test case describe the queries. Each line contains a single integer k $(1 \leq k \leq 10^{18})$. It is also guaranteed that k does not exceed the final length of s.

It is guaranteed that the sum of n and q across all test cases does not exceed $2 \cdot 10^5$ and 10^4, respectively.

Output

For each query, print the k-th letter of the final string s.

Example

input

```
2
4 3 3
mark
1 4
5 7
3 8
1
10
12
7 3 3
creamii
2 3
3 4
2 9
9
11
12
```

output

```
m
a
r
e
a
r
```

Note

In the first test case, the copy-paste process is as follows.

- The first step is pasting string `mark` at the end, yielding the string `markmark`.
- The second step is pasting string `mar` at the end, yielding the string `markmarkmar`.
- The third step is pasting string `rkmark` at the end, yielding the string `markmarkmarrkmark`.

In the second test case, the copy-paste process is as follows.

- The first step is pasting string `re` at the end, yielding the string `creamiire`.
- The second step is pasting string `ea` at the end, yielding the string `creamiireea`.
- The third step is pasting string `reamiire` at the end, yielding the string `creamiireeareamiire`.

D. Mark and Lightbulbs

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

Mark has just purchased a rack of n lightbulbs. The state of the lightbulbs can be described with binary string $s = s_1 s_2 \dots s_n$, where $s_i = 1$ means that the i-th lightbulb is turned on, while $s_i = 0$ means that the i-th lightbulb is turned off.

Unfortunately, the lightbulbs are broken, and the only operation he can perform to change the state of the lightbulbs is the following:

- Select an index i from $2, 3, \dots, n-1$ such that $s_{i-1} \neq s_{i+1}$.
- Toggle s_i. Namely, if s_i is 0, set s_i to 1 or vice versa.

Mark wants the state of the lightbulbs to be another binary string t. Help Mark determine the minimum number of operations to do so.

Input

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

The first line of each test case contains a single integer n ($3 \le n \le 2 \cdot 10^5$) — the number of lightbulbs.

The second line of each test case contains a binary string s of length n — the initial state of the lightbulbs.

The third line of each test case contains a binary string t of length n — the final state of the lightbulbs.

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

Output

For each test case, print a line containing the minimum number of operations Mark needs to perform to transform s to t. If there is no such sequence of operations, print -1.

Example

input

```
4
4
0100
0010
4
1010
0100
5
01001
00011
6
000101
010011
```

output

```
2
-1
-1
5
```

Note

In the first test case, one sequence of operations that achieves the minimum number of operations is the following.

- Select $i = 3$, changing 0100 to 0110.
- Select $i = 2$, changing 0110 to 0010.

In the second test case, there is no sequence of operations because one cannot change the first digit or the last digit of s.
In the third test case, even though the first digits of s and t are the same and the last digits of s and t are the same, it can be shown that there is no sequence of operations that satisfies the condition.

In the fourth test case, one sequence that achieves the minimum number of operations is the following:

- Select $i = 3$, changing 000101 to 001101.
- Select $i = 2$, changing 001101 to 011101.
- Select $i = 4$, changing 011101 to 011001.
- Select $i = 5$, changing 011001 to 011011.
- Select $i = 3$, changing 011011 to 010011.

E. Mark and Professor Koro

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

After watching a certain anime before going to sleep, Mark dreams of standing in an old classroom with a blackboard that has a sequence of n positive integers $a_1, a_2, \ldots, a_n$ on it.

Then, professor Koro comes in. He can perform the following operation:

- select an integer x that appears at least 2 times on the board,
- erase those 2 appearances, and
- write $x + 1$ on the board.

Professor Koro then asks Mark the question, "what is the maximum possible number that could appear on the board after some operations?"

Mark quickly solves this question, but he is still slower than professor Koro. Thus, professor Koro decides to give Mark additional challenges. He will update the initial sequence of integers q times. Each time, he will choose positive integers k and l, then change a_k to l. After each update, he will ask Mark the same question again.

Help Mark answer these questions faster than Professor Koro!

Note that the updates are persistent. Changes made to the sequence a will apply when processing future updates.

Input

The first line of the input contains two integers n and q ($2 \le n \le 2 \cdot 10^5$, $1 \le q \le 2 \cdot 10^5$) — the length of the sequence a and the number of updates, respectively.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$)

Then, q lines follow, each consisting of two integers k and l ($1 \le k \le n$, $1 \le l \le 2 \cdot 10^5$), telling to update a_k to l.

Output

Print q lines. The i-th line should consist of a single integer — the answer after the i-th update.

Examples

input

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

output

```
6
5
4
5
```

input

```
2 1
200000 1
2 200000
```

output

```
200001
```

Note

In the first example test, the program must proceed through 4 updates.

The sequence after the first update is $[2, 3, 2, 4, 5]$. One sequence of operations that achieves the number 6 the following.

- Initially, the blackboard has numbers $[2, 3, 2, 4, 5]$.
- Erase two copies of 2 and write 3, yielding $[3, 4, 5, 3]$.
- Erase two copies of 3 and write 4, yielding $[4, 5, 4]$.
- Erase two copies of 4 and write 5, yielding $[5, 5]$.
- Erase two copies of 5 and write 6, yielding $[6]$.

Then, in the second update, the array is changed to $[2, 3, 2, 4, 3]$. This time, Mark cannot achieve 6. However, one sequence that Mark can use to achieve 5 is shown below.

- Initially, the blackboard has $[2, 3, 2, 4, 3]$.
- Erase two copies of 2 and write 3, yielding $[3, 4, 3, 3]$.
- Erase two copies of 3 and write 4, yielding $[3, 4, 4]$.
- Erase two copies of 4 and write 5, yielding $[3, 5]$.

In the third update, the array is changed to $[2, 3, 2, 1, 3]$. One way to achieve 4 is shown below.

- Initially, the blackboard has $[2, 3, 2, 1, 3]$.
- Erase two copies of 3 and write 4, yielding $[2, 2, 1, 4]$.

F. Mark and the Online Exam

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

Mark is administering an online exam consisting of n true-false questions. However, he has lost all answer keys. He needs a way to retrieve the answers before his client gets infuriated.

Fortunately, he has access to the grading system. Thus, for each query, you can input the answers to all n questions, and the grading system will output how many of them are correct.

He doesn't have much time, so he can use the grading system at most 675 times. Help Mark determine the answer keys.

Note that answer keys are fixed in advance and will not change depending on your queries.

Input

The first line of the input consists of an integer n ($1 \le n \le 1000$) — the number of questions.

Interaction

After reading n, you can start making queries to the grading system. For each query, print a line containing a string s of length n consisting of only letters 'T' and 'F'.

- $s_i =$'T' means that you answer the i-question `true`.
- $s_i =$'F' means that you answer the i-question `false`.

After a successful query, you should read an integer k ($0 \le k \le n$) — the number of correct answers. **If you read n, then you found the answers, and your program should not make any more queries.**

If your program reads $k = -1$ instead of the number of correct answers, it means that you either made an invalid query or exceeded the query limits. Exit immediately after receiving -1, and you will see `Wrong answer` verdict. Otherwise, you can get an arbitrary verdict because your solution will continue to read from a closed stream.

After printing a query do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

Hacks

To hack, use the following format:

The first line contains an integer n ($1 \le n \le 1000$) — the number of questions.

The second line contains a string s of length n consisting of only 'T' and 'F' — the answer key.

Examples

input

```
3
1
3
```

output

```

FTT
TTF
```

input

```
4
0
3
4
```

output

```

FTFF
TTTT
TFTT
```

Note

The empty lines in the example are just for you to better understand the interaction process. **You're not required to print them.**

In the first example, there are 3 questions, and the answer to each question is '`true`', '`true`', and '`false`', respectively.

- The first query, guessing the answers to be '`false`', '`true`', and '`true`', respectively, guesses only one question — the 2-nd question — correctly.
- Then, in the second query, the program correctly guesses the answer key. The interaction ends here.

In the second example, there are 4 questions, and the answer to each question is '`true`', '`false`', '`true`', and '`true`', respectively.

- The first query guessed none of the questions correctly, resulting in the answer 0.
- The second query guessed the 1-st, 3-rd, and 4-th question correctly, resulting in the answer 3.
- In the third query, the program correctly guesses the answer key. Then, the interaction ends.