Candies and Milestones

Little Pratya loves collecting candies and she also likes playing games :).

Today Pratya is travelling in bus and she has a bag full of **N** candies. She looks outside the window and notices milestones with numbers written on them. She decides to play a game. She wants to select some initial *non-zero* number of candies from the bag and call it her collection of candies. When she sees a milestone on the way, she will add or remove some candies from her collection as following.

Let the previous milestone number be **a1** and the current milestone number be **a2**.

a2 > a1 : Pratya adds (a2 - a1) candies to her collection (thus, her bag of candies loses them). If the bag doesn't contain at least (a2 - a1) candies, she starts crying.

a2 < a1 : Pratya removes (a1 - a2) candies from her collection (thus, her bag of candies gains them). If her collection doesn't contain more than (a1 - a2) candies, she starts crying.

a1 = a2 : Pratya is bored to see the same number again and so she eats one of the candies from her collection.

Moreover, Pratya always wants her candy collection to have at least one candy, otherwise she starts crying. Given the numbers written on all the milestones in order, find the minimum number of candies Pratya should select in the beginning so that she doesn't have to cry at all. If it is not possible , print -1. Note that no change to the collection happens when she sees the first milestone.

Input

The first line contain the number of test cases **T**. Each test case has two lines. The first line specifies **N** and **M**. **N** is the number of candies in the bag and **M** is the total number of milestones. Second line contains **M** integers, the numbers written on the milestones, in the order Pratya sees them.

^ is used for power.

T <= 150

1 <= N <= 10^7

2 <= M <= 10^4

Each milestone number will be between [-10^6,10^6].

Output

For each test case output the minimum number of candies that should be selected from the bag or -1 if its not possible

Example

Input:
3
10 5
1 5 1 -1 -2
2 5
1 5 1 -1 -2
100 3
1 2 3

Output:
4
-1
1

Note : Large input, prefer using scanf / printf to cin / cout
*There are multiple test sets, and the judge shows the **sum** of the time taken over all test sets of your submission, if Accepted.*