New Year Chaos

It's New Year's Day and everyone's in line for the Wonderland rollercoaster ride! There are a number of people queued up, and each person wears a sticker indicating their *initial* position in the queue. Initial positions increment by 1 from 1 at the front of the line to n at the back.

Any person in the queue can bribe the person *directly in front* of them to swap positions. If two people swap positions, they still wear the same sticker denoting their original places in line. One person can bribe *at most two others*. For example, if $n = 8$ and $Person\ 5$ bribes $Person\ 4$, the queue will look like this: $1, 2, 3, 5, 4, 6, 7, 8$.

Fascinated by this chaotic queue, you decide you must know the minimum number of bribes that took place to get the queue into its current state!

Function Description

Complete the function *minimumBribes* in the editor below. It must print an integer representing the minimum number of bribes necessary, or `Too chaotic` if the line configuration is not possible.

minimumBribes has the following parameter(s):

- *q*: an array of integers

Input Format

The first line contains an integer t, the number of test cases.

Each of the next t pairs of lines are as follows:
- The first line contains an integer t, the number of people in the queue
- The second line has n space-separated integers describing the final state of the queue.

Constraints

- $1 \leq t \leq 10$
- $1 \leq n \leq 10^5$

Subtasks

For 60% score $1 \leq n \leq 10^3$
For 100% score $1 \leq n \leq 10^5$

Output Format

Print an integer denoting the minimum number of bribes needed to get the queue into its final state. Print `Too chaotic` if the state is invalid, i.e. it requires a person to have bribed more than 2 people.

Sample Input

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

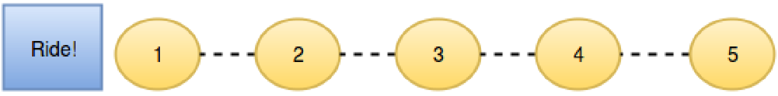
Sample Output

```
3
Too chaotic
```

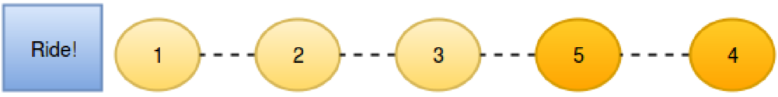
Explanation

Test Case 1

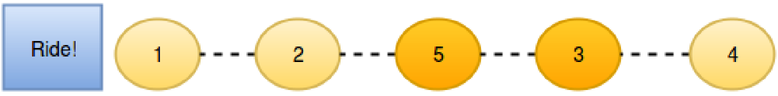
The initial state:



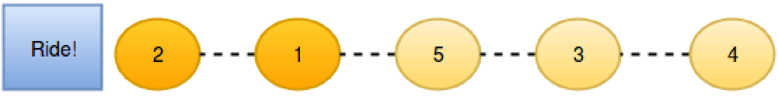
After person 5 moves one position ahead by bribing person 4:



Now person 5 moves another position ahead by bribing person 3:



And person 2 moves one position ahead by bribing person 1:



So the final state is $2, 1, 5, 3, 4$ after three bribing operations.

Test Case 2

No person can bribe more than two people, so its not possible to achieve the input state.

```
def checkChaos(a):

    n=len(a)
    f=True
    c=0
    d={}

    for i in range(n,0,-1):
        p=i-3
        if (p<0):
            p=0
        try:
            temp=a.index(i,p)
        except ValueError:
            f=False
            print("Too chaotic")
            break

        if(temp!=i-1):
            a[temp],a[temp+1]=a[temp+1],a[temp]
            c+=1
            temp+=1

            if(temp!=i-1):
                a[temp],a[temp+1]=a[temp+1],a[temp]
                c+=1
                temp+=1

            if(temp!=i-1):
                f=False
                print("Too chaotic")
                break

    if(f):
        print(c)

t=int(input())
arr=[]
for i in range(t):
    g=int(input())
    arr.append(list(map(int,input().split())))

for j in arr:
    checkChaos(j)
```