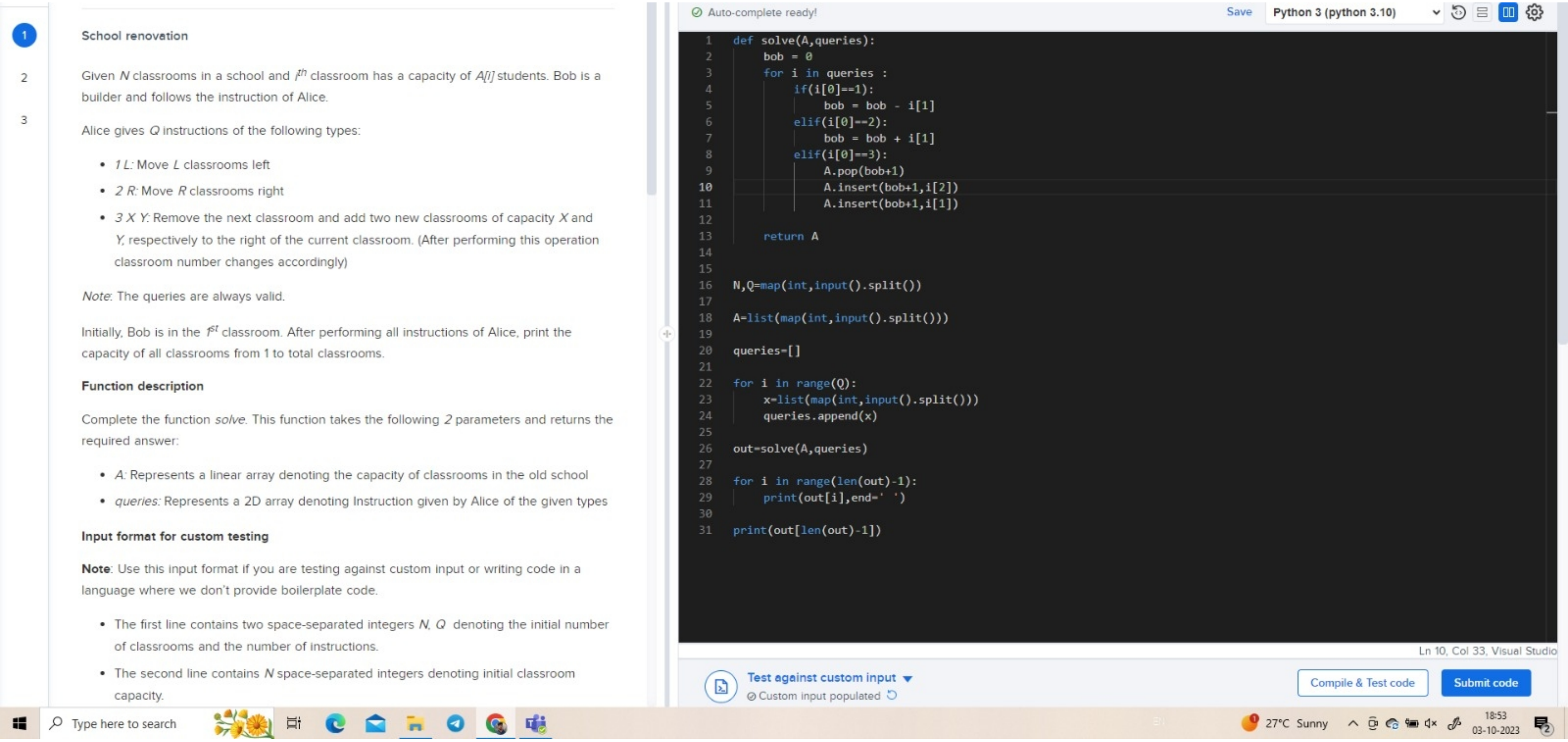

1

2

3

School renovation

Given *N* classrooms in a school and *i*th classroom has a capacity of *A[i]* students. Bob is a builder and follows the instruction of Alice.

Alice gives *Q* instructions of the following types:

- *1 L*: Move *L* classrooms left
- *2 R*: Move *R* classrooms right
- *3 X Y*: Remove the next classroom and add two new classrooms of capacity *X* and *Y*, respectively to the right of the current classroom. (After performing this operation classroom number changes accordingly)

Note: The queries are always valid.

Initially, Bob is in the 1st classroom. After performing all instructions of Alice, print the capacity of all classrooms from 1 to total classrooms.

Function description

Complete the function *solve*. This function takes the following *2* parameters and returns the required answer:

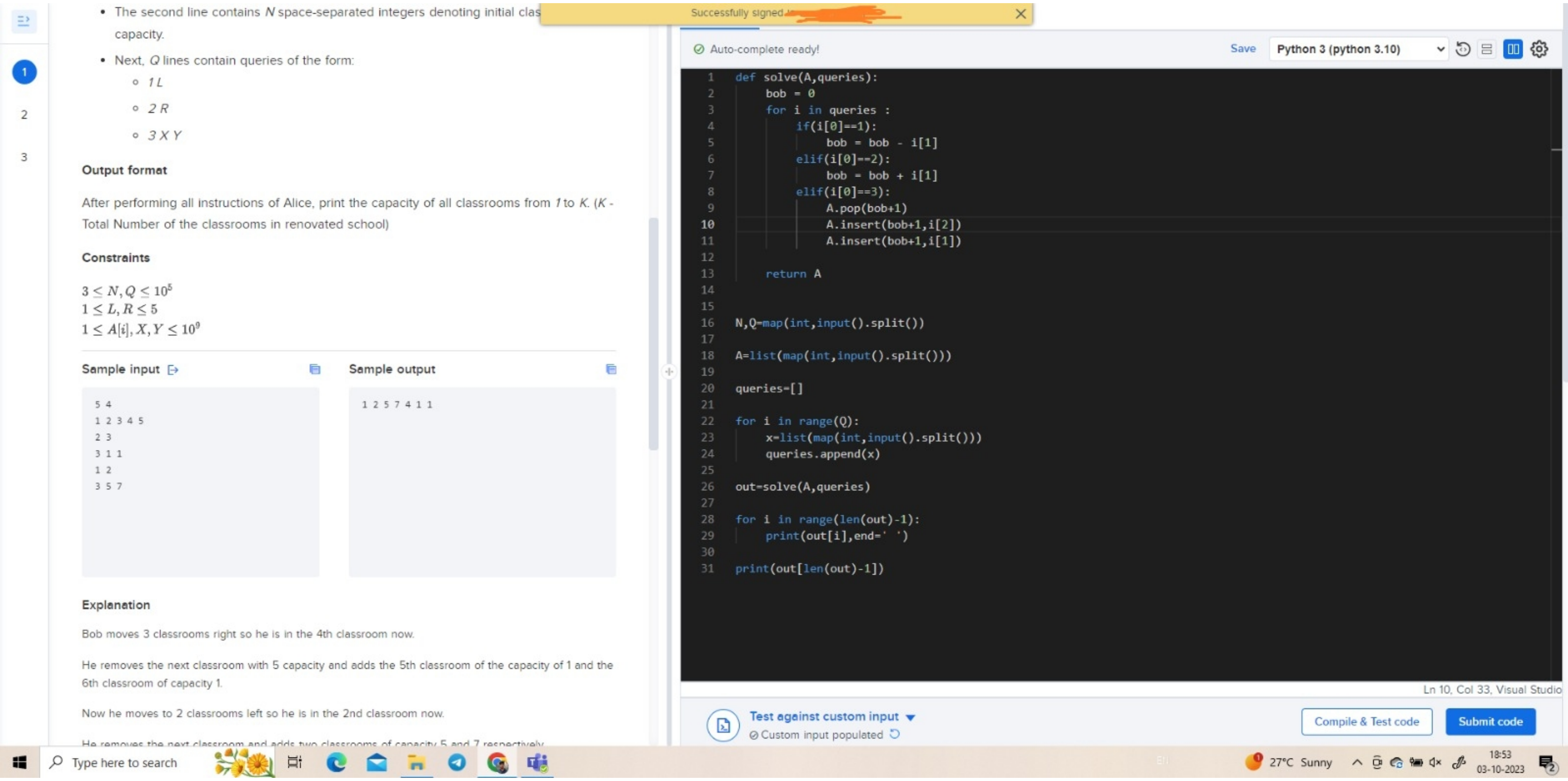
- *A*: Represents a linear array denoting the capacity of classrooms in the old school
- *queries*: Represents a 2D array denoting Instruction given by Alice of the given types

Input format for custom testing

Note: Use this input format if you are testing against custom input or writing code in a language where we don't provide boilerplate code.

- The first line contains two space-separated integers *N*, *Q* denoting the initial number of classrooms and the number of instructions.
- The second line contains *N* space-separated integers denoting initial classroom capacity.

```python
def solve(A,queries):
    bob = 0
    for i in queries :
        if(i[0]==1):
            bob = bob - i[1]
        elif(i[0]==2):
            bob = bob + i[1]
        elif(i[0]==3):
            A.pop(bob+1)
            A.insert(bob+1,i[2])
            A.insert(bob+1,i[1])

    return A


N,Q=map(int,input().split())

A=list(map(int,input().split()))

queries=[]

for i in range(Q):
    x=list(map(int,input().split()))
    queries.append(x)

out=solve(A,queries)

for i in range(len(out)-1):
    print(out[i],end=' ')

print(out[len(out)-1])
```



- The second line contains *N* space-separated integers denoting initial clas capacity.
- Next, *Q* lines contain queries of the form:
  - *1 L*
  - *2 R*
  - *3 X Y*

Output format

After performing all instructions of Alice, print the capacity of all classrooms from *1* to *K*. (*K* - Total Number of the classrooms in renovated school)

Constraints

$3 \leq N, Q \leq 10^5$
$1 \leq L, R \leq 5$
$1 \leq A[i], X, Y \leq 10^9$

Sample input

```
5 4
1 2 3 4 5
2 3
3 1 1
1 2
3 5 7
```

Sample output

```
1 2 5 7 4 1 1
```

Explanation

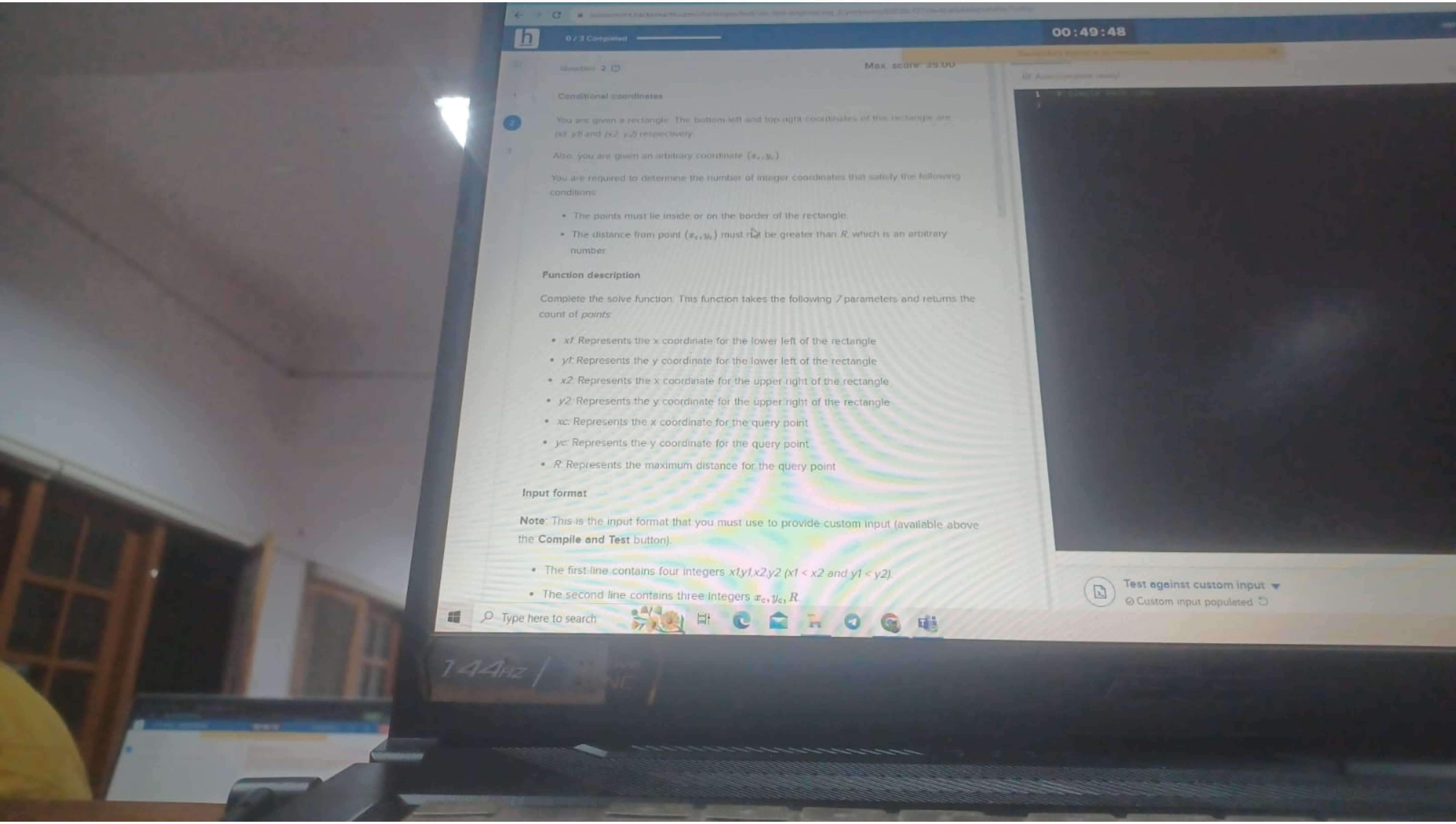
Bob moves 3 classrooms right so he is in the 4th classroom now.

He removes the next classroom with 5 capacity and adds the 5th classroom of the capacity of 1 and the 6th classroom of capacity 1.

Now he moves to 2 classrooms left so he is in the 2nd classroom now.

He removes the next classroom and adds two classrooms of capacity 5 and 7 respectively.

```python
def solve(A,queries):
    bob = 0
    for i in queries :
        if(i[0]==1):
            bob = bob - i[1]
        elif(i[0]==2):
            bob = bob + i[1]
        elif(i[0]==3):
            A.pop(bob+1)
            A.insert(bob+1,i[2])
            A.insert(bob+1,i[1])

    return A


N,Q=map(int,input().split())

A=list(map(int,input().split()))

queries=[]

for i in range(Q):
    x=list(map(int,input().split()))
    queries.append(x)

out=solve(A,queries)

for i in range(len(out)-1):
    print(out[i],end=' ')

print(out[len(out)-1])
```



Conditional coordinates

You are given a rectangle. The bottom-left and top-right coordinates of this rectangle are (x1, y1) and (x2, y2) respectively.

Also, you are given an arbitrary coordinate (x_c, y_c).

You are required to determine the number of integer coordinates that satisfy the following conditions:

- The points must lie inside or on the border of the rectangle.
- The distance from point (x_c, y_c) must not be greater than R, which is an arbitrary number.

Function description

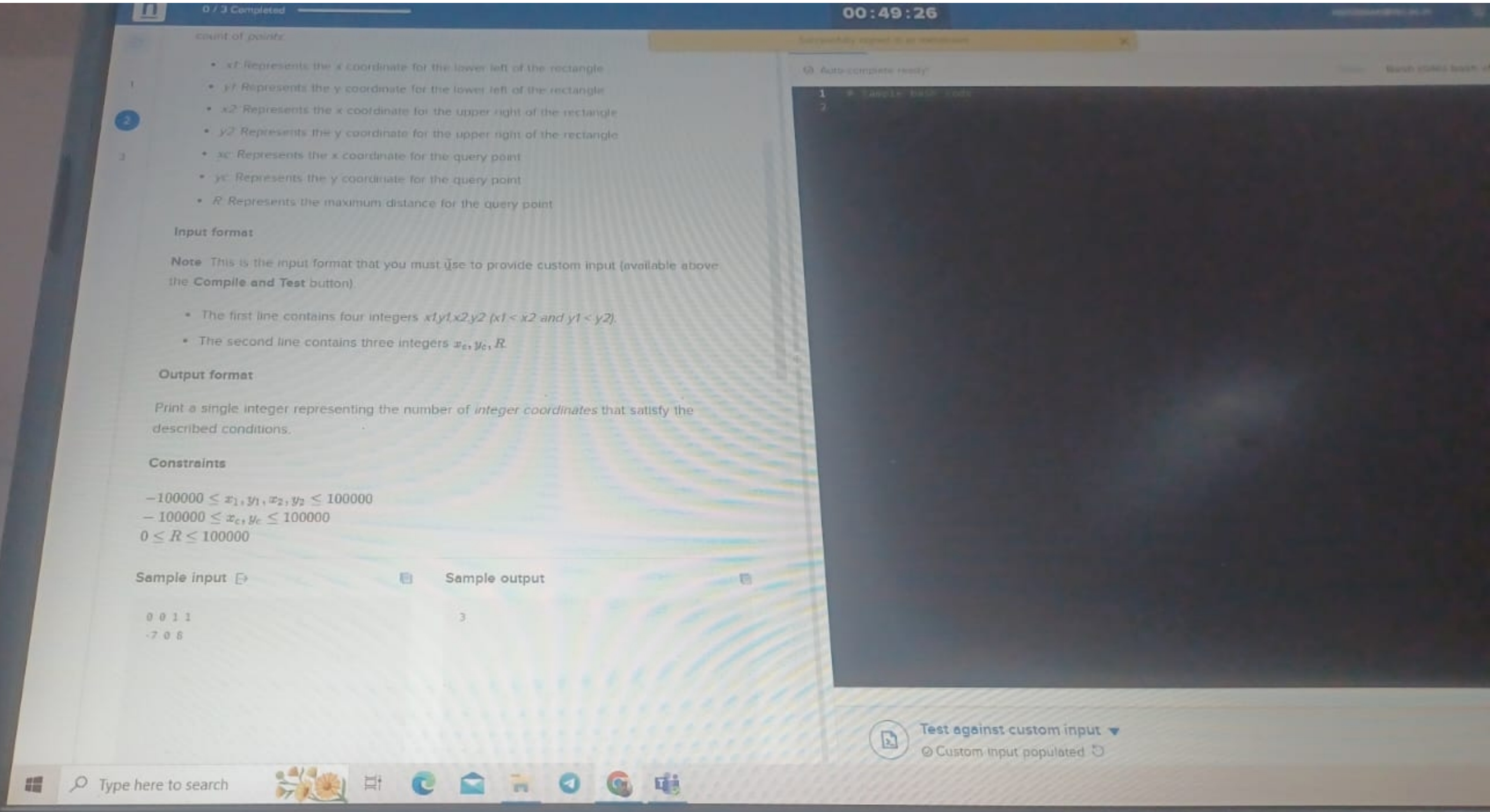
Complete the solve function. This function takes the following 7 parameters and returns the count of *points*:

- *x1*: Represents the x coordinate for the lower left of the rectangle
- *y1*: Represents the y coordinate for the lower left of the rectangle
- *x2*: Represents the x coordinate for the upper right of the rectangle
- *y2*: Represents the y coordinate for the upper right of the rectangle
- *xc*: Represents the x coordinate for the query point
- *yc*: Represents the y coordinate for the query point
- *R*: Represents the maximum distance for the query point

Input format

Note: This is the input format that you must use to provide custom input (available above the **Compile and Test** button).

- The first line contains four integers *x1,y1,x2,y2 (x1 < x2 and y1 < y2)*.
- The second line contains three integers x_c, y_c, R



count of points

- x1 Represents the x coordinate for the lower left of the rectangle
- y1 Represents the y coordinate for the lower left of the rectangle
- x2 Represents the x coordinate for the upper right of the rectangle
- y2 Represents the y coordinate for the upper right of the rectangle
- xc Represents the x coordinate for the query point
- yc Represents the y coordinate for the query point
- R Represents the maximum distance for the query point

Input format

Note This is the input format that you must use to provide custom input (available above the **Compile and Test** button).

- The first line contains four integers *x1,y1,x2,y2 (x1 < x2 and y1 < y2).*
- The second line contains three integers x_c, y_c, R.

Output format

Print a single integer representing the number of *integer coordinates* that satisfy the described conditions.

Constraints

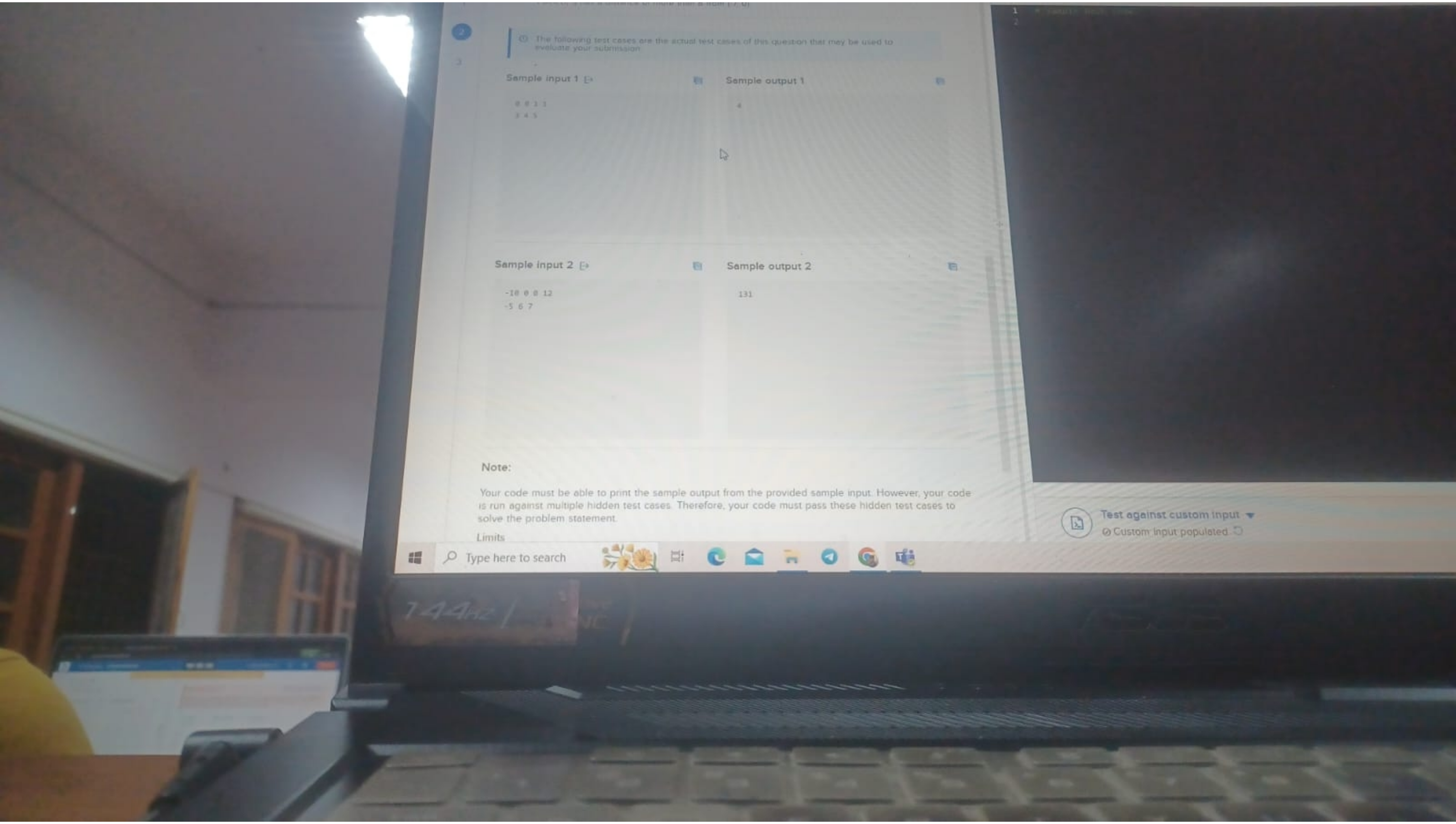
$-100000 \le x_1, y_1, x_2, y_2 \le 100000$
$-100000 \le x_c, y_c \le 100000$
$0 \le R \le 100000$

Sample input

```
0 0 1 1
-7 0 8
```

Sample output

```
3
```



The following test cases are the actual test cases of this question that may be used to evaluate your submission

Sample input 1

```
0 0 1 1
3 4 5
```

Sample output 1

```
4
```

Sample input 2

```
-10 0 0 12
-5 6 7
```

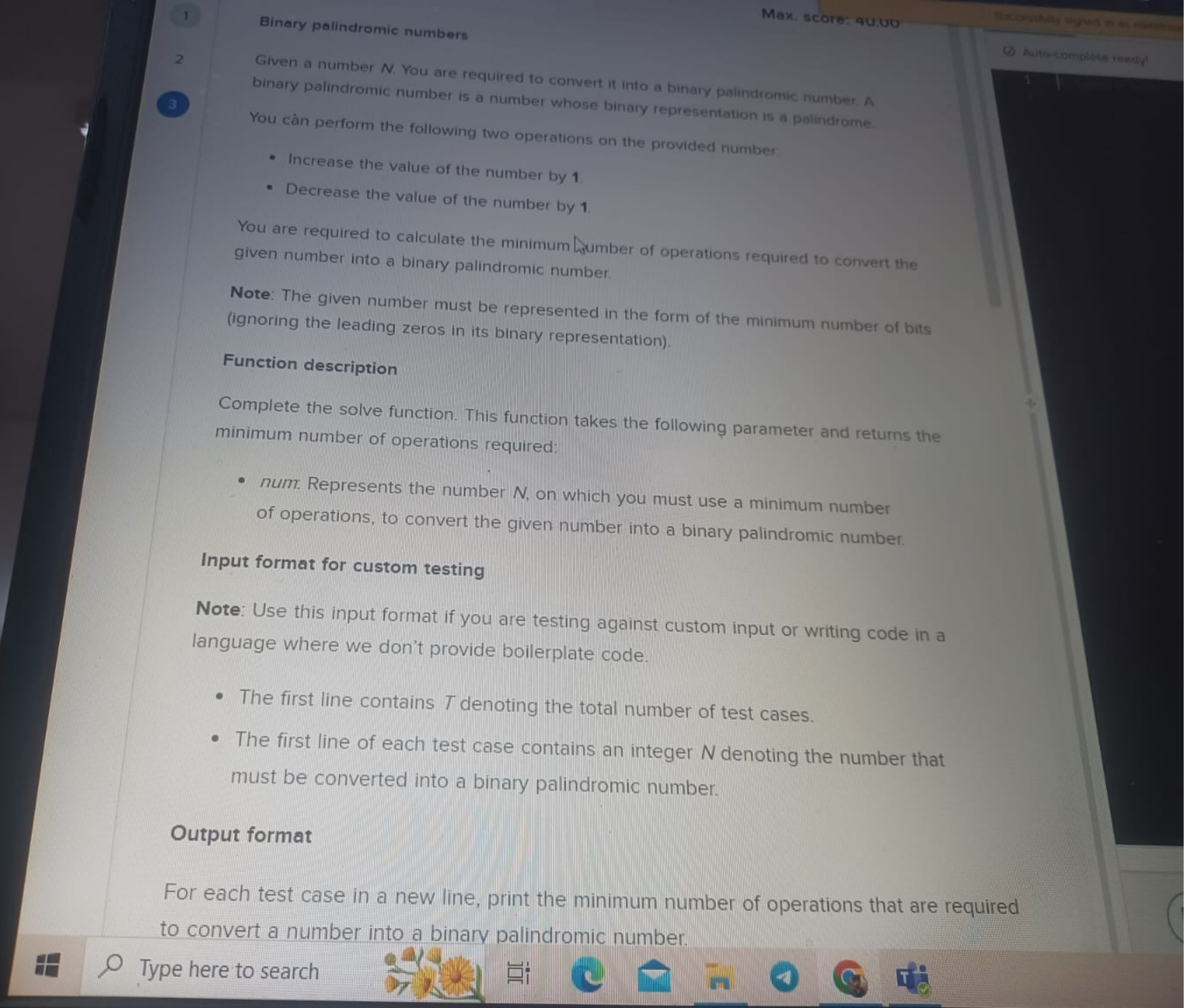
Sample output 2

```
131
```

Note:

Your code must be able to print the sample output from the provided sample input. However, your code is run against multiple hidden test cases. Therefore, your code must pass these hidden test cases to solve the problem statement.

Limits



Max. score: 40.00

Binary palindromic numbers

Given a number *N*. You are required to convert it into a binary palindromic number. A binary palindromic number is a number whose binary representation is a palindrome.

You càn perform the following two operations on the provided number:

- Increase the value of the number by **1**.
- Decrease the value of the number by **1**.

You are required to calculate the minimum number of operations required to convert the given number into a binary palindromic number.

Note: The given number must be represented in the form of the minimum number of bits (ignoring the leading zeros in its binary representation).

Function description

Complete the solve function. This function takes the following parameter and returns the minimum number of operations required:

- *num*: Represents the number *N*, on which you must use a minimum number of operations, to convert the given number into a binary palindromic number.

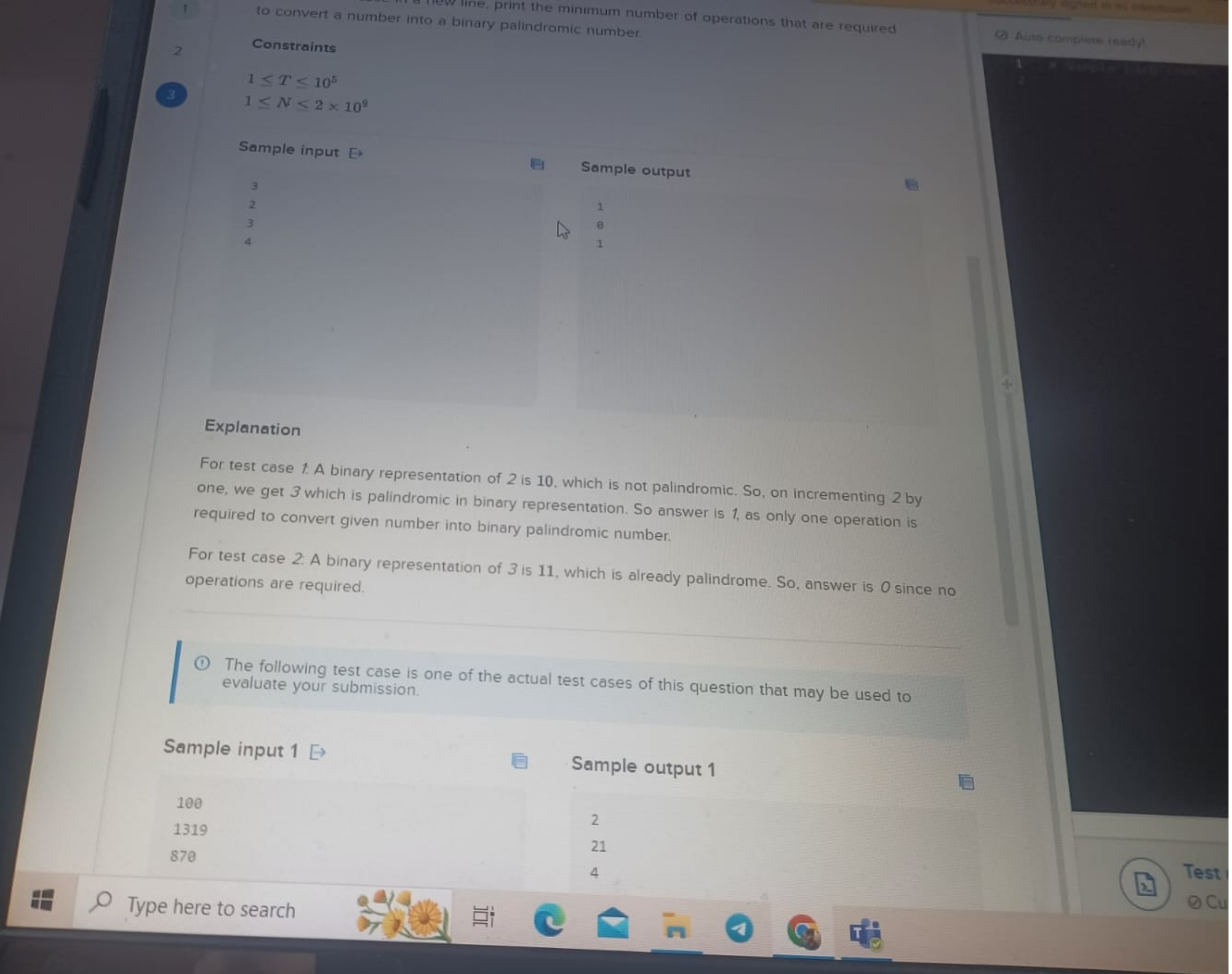
Input format for custom testing

Note: Use this input format if you are testing against custom input or writing code in a language where we don't provide boilerplate code.

- The first line contains *T* denoting the total number of test cases.
- The first line of each test case contains an integer *N* denoting the number that must be converted into a binary palindromic number.

Output format

For each test case in a new line, print the minimum number of operations that are required to convert a number into a binary palindromic number.



new line, print the minimum number of operations that are required to convert a number into a binary palindromic number.

Constraints

$1 \le T \le 10^5$
$1 \le N \le 2 \times 10^9$

Sample input

```
3
2
3
4
```

Sample output

```
1
0
1
```

Explanation

For test case *1*: A binary representation of *2* is 10, which is not palindromic. So, on incrementing *2* by one, we get *3* which is palindromic in binary representation. So answer is *1*, as only one operation is required to convert given number into binary palindromic number.

For test case *2*: A binary representation of *3* is 11, which is already palindrome. So, answer is *0* since no operations are required.

The following test case is one of the actual test cases of this question that may be used to evaluate your submission.

Sample input 1

```
100
1319
870
```

Sample output 1

```
2
21
4
```