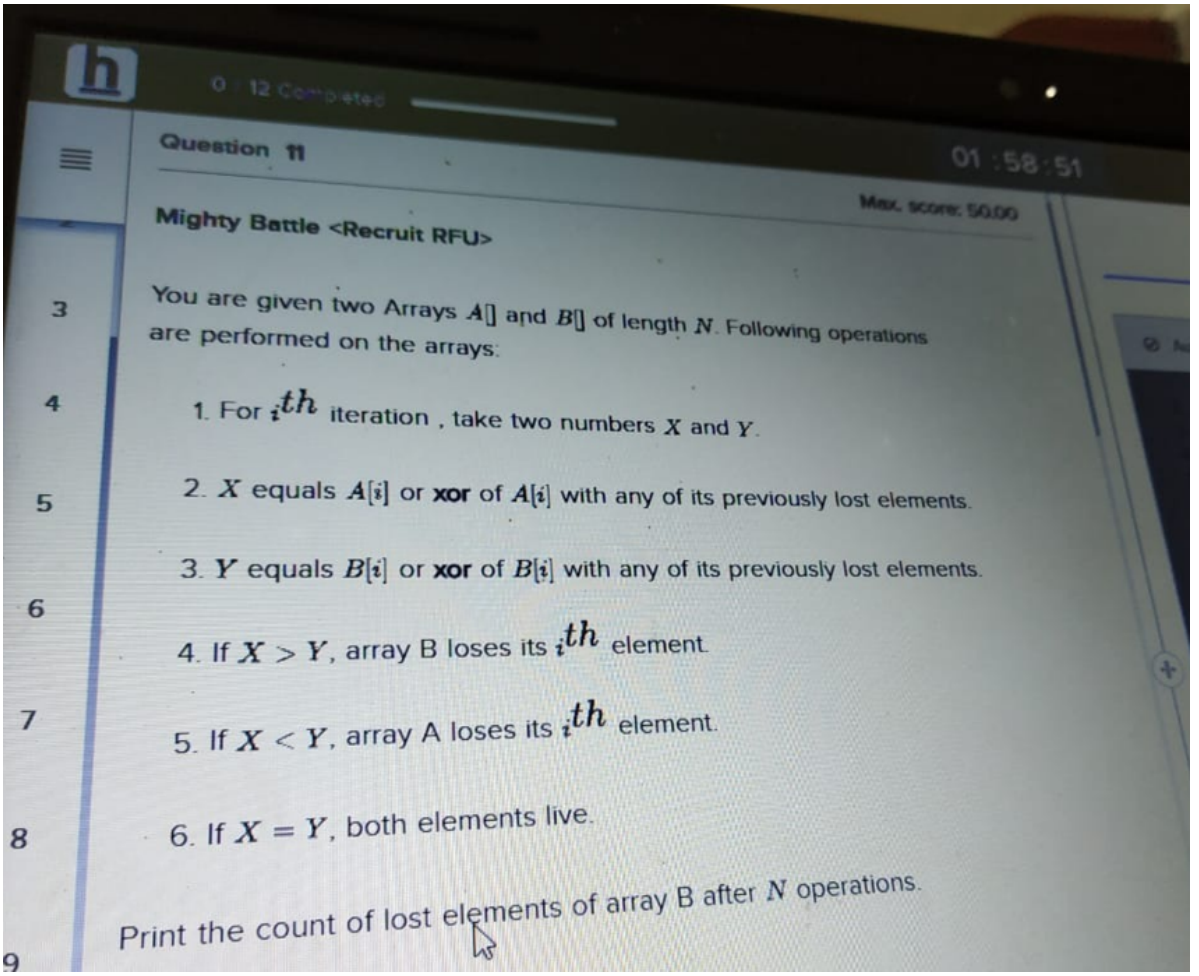
Oppo Codes

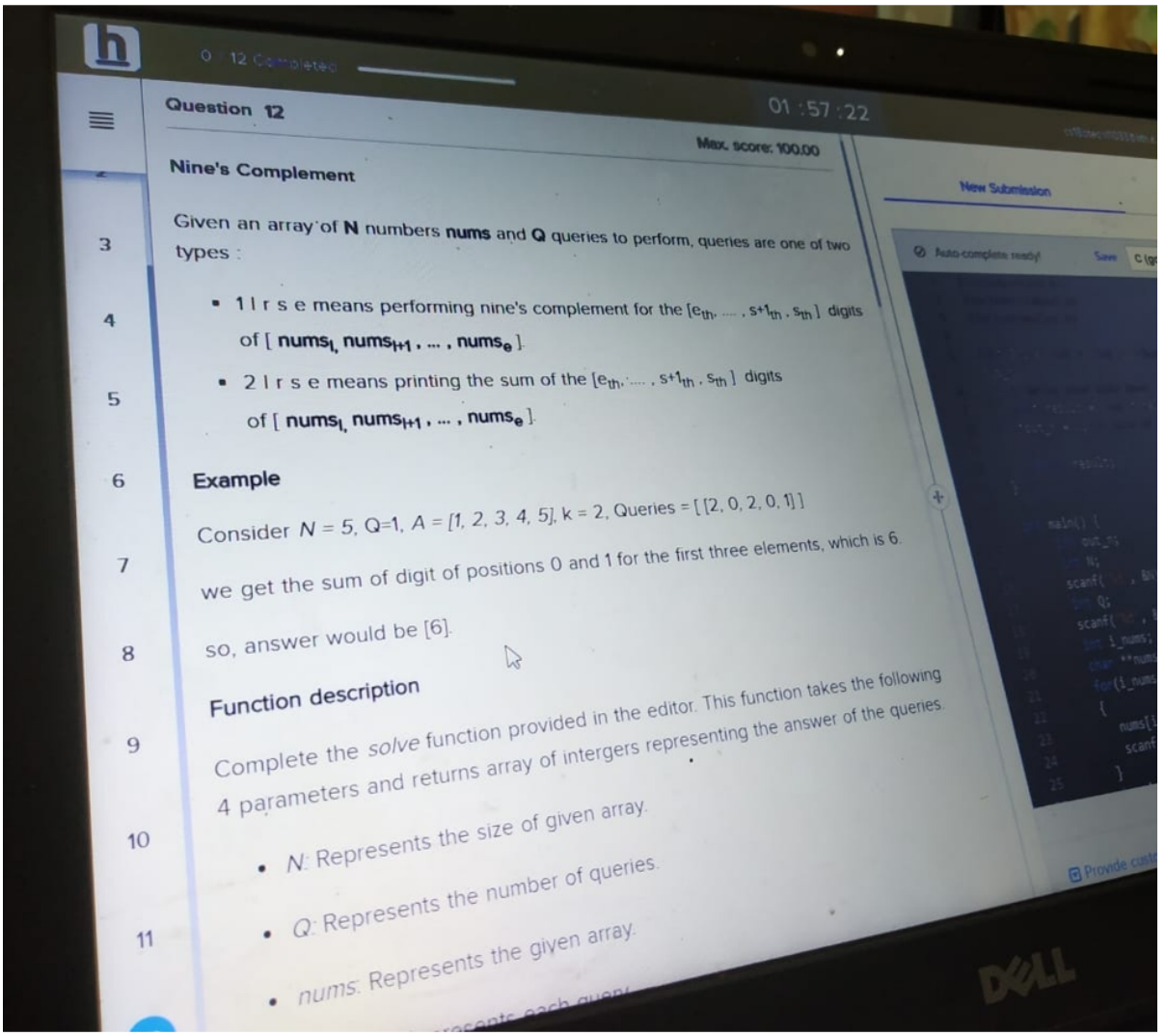
Mighty Battle




```
def max_xor(a,arr):
    if len(arr)==0:
        return a
    else:
        m=arr[0]
        for i in range(len(arr)):
            c=a^arr[i]
            if m<=c:
                m=c
        return m
def solver(N,A,B):
    A_dead=[]
    B_dead=[]
    for i in range(N):
        X=max_xor(A[i],A_dead)
        Y=max_xor(B[i],B_dead)
```

```
        if X>Y:
            # print(B)
            b=B[i]
            B_dead.append(b)
        elif X<Y:
            # print(A)
            a=A[i]
            A_dead.append(a)
    return len(B)-len(B_dead)
```

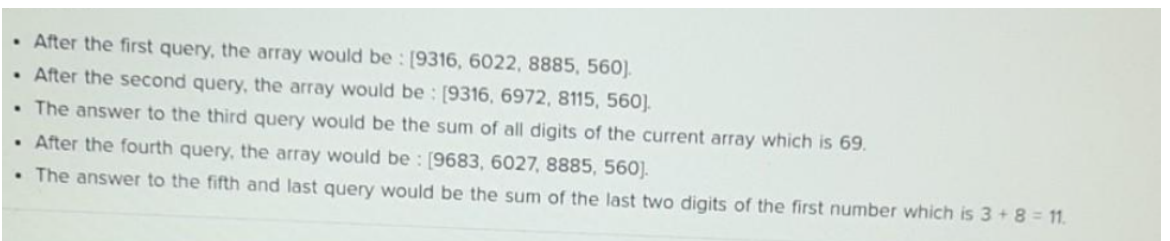
Nine's Complement




```
def nine_complement(N,s,e,number):
    if len(number)<N:
        while True:
            number='0'+number
            if len(number)==N:
                break
    for i in range(N-1-e,N-s):
```

```
        if(number[i] != '.'):
            a = 9 - int(number[i])
            number = (number[:i] +
                      str(a) + number[i + 1:])
    return str(int(number))
def adder(N,s,e,number):
    if len(number)<N:
        while True:
            number='0'+number
            if len(number)==N:
                break
    s=0
    for i in range(N-1-e,N-s):
        if(number[i] != '.'):
            a =  int(number[i])
            s=s+a
    return s
for q in queries:
    type=q[0]
    l=q[1]
    r=q[2]
    s=q[3]
    e=q[4]
    if type==1:
        temp_nums=arr[l:r+1]
        arr[l:r+1]=[nine_complement(N,s,e,t) for t in temp_nums]
    if type==2:
        temp_nums=arr[l:r+1]
        s=0
        for t in temp_nums:
            s=s+adder(N,s,e,t)
        print(s)
##Driver Code Test Case from Chegg
arr=[616,3922,1185,560]
queries=[[1,0,2,2,3],[1,1,2,1,2],[2,0,3,0,3],[1,0,2,0,2],[2,0,0,0,1]]
arr=list(map(str,arr))
N=4
Result Obtained
['9316', '6022', '8885', '560']
['9316', '6972', '8115', '560']
69
['9683', '6027', '8884', '560']
11
```

Result test case from chegg



Numbers and Cards

Question 11 **Max. score: 100.00**

Numbers and cards

You are given *10* types of cards. Every card has a specific number written on it from *0* to *9*. There are *K* cards for each number. You have a total of *10*K* cards.

Task

Select *N* cards from the provided *10*K* cards and determine the number of different sets of *N* cards that can be generated.

The size of these sets must be *N*. If the order of the cards is different in two sets, then these sets are considered equal.

Note: Since the output can be large, print output modulo $(10^9 + 7)$.

Example

Assumptions

- *K = 1*
- *N = 2*

Approach

You have total *10* cards each of *0* to *9*. You have to select a total of *2* cards. So there are *(10*9)/2 = 45* ways.

Hence, the answer is *45*.

Example

Assumptions

- *K = 1*
- *N = 2*

Approach

You have total *10* cards each of *0* to *9*. You have to select a total of *2* cards. So there are *(10*9)/2 = 45* ways.

Hence, the answer is *45*.

Function description

Complete the *solve* function provided in the editor. This function takes the following *2* parameters and returns the count of *ways*:

- *N*: Represents the number of cards to be selected
- *K*: Represents the count of each type of card

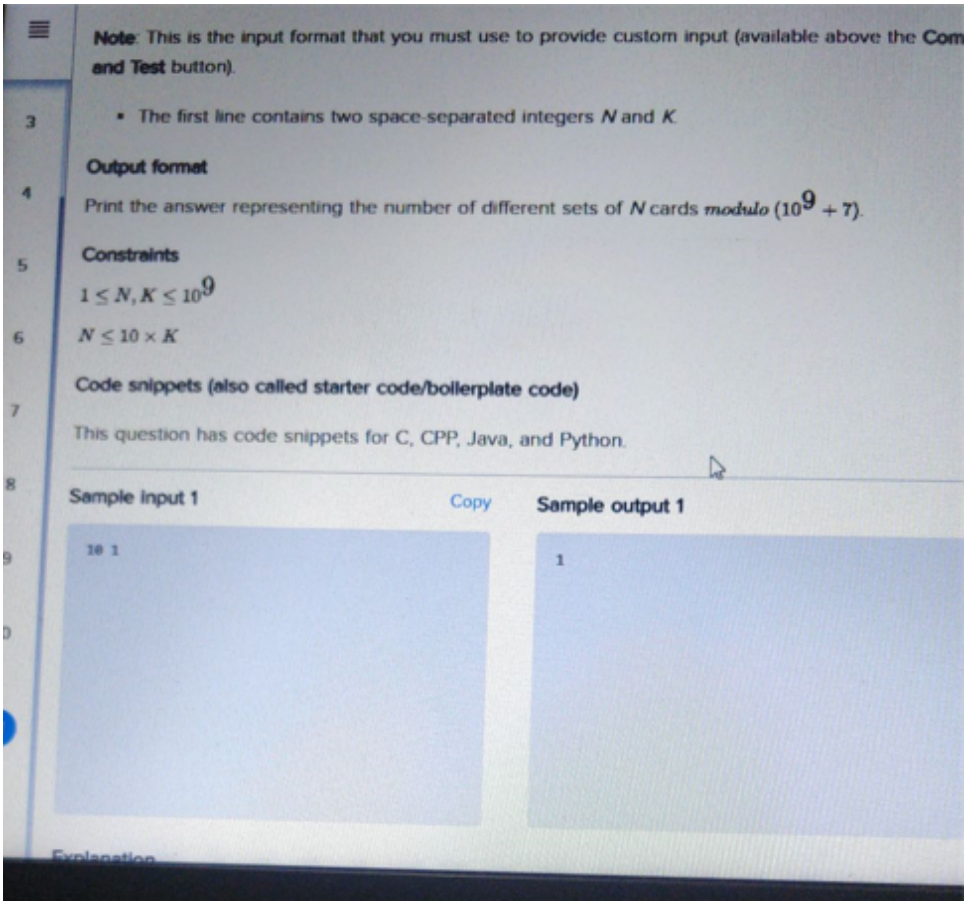
Input format

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 two space-separated integers *N* and *K*.

Output format

Print the answer representing the number of different sets of *N* cards *modulo* $(10^9 + 7)$.



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

- The first line contains two space-separated integers N and K.

Output format

Print the answer representing the number of different sets of N cards *modulo* $(10^9 + 7)$.

Constraints

$1 \leq N, K \leq 10^9$

$N \leq 10 \times K$

Code snippets (also called starter code/boilerplate code)

This question has code snippets for C, CPP, Java, and Python.

Sample input 1 Copy

```
10 1
```

Sample output 1

```
1
```

Explanation