# Problem J. Bob's Poor Math

| | |
|---|---|
| Input file: | `standard input` |
| Output file: | `standard output` |
| Time limit: | 4 seconds |
| Memory limit: | 256 megabytes |

Bob's math is really poor. He even doesn't know how does 'carry' works in plus operation. In his world, the 'plus' operation performed like this : $9 + 5 = 4$, $99 + 99 = 88$, $5876 + 5576 = 342$, $5555 + 5555 = 0$, $1503 + 2503 = 3006$, $512 + 1314 = 1826$. Oh how poor his math is and he could even be the problem setter. Tragedy. In his world, there is a 'Wish' data structure. Let's see how it works:

*. Initially, there will be a list of numbers to initialize the data structure. After initialization, game begins:

1. You may add a number to his data structure.

   *Add* $x$

2. You may divide all the numbers in the data structure by 10. (Note: it is integer division, for example, $9/10 = 0$, $99/10 = 9$, $5876/10 = 587$)

   $Shift$

3. You may query the largest result by given $x$ 'plus' any number from his data structure.

   *Query* $x$

4. You may query 'sum' ( by his poor math ) of numbers between $L$ and $R$ inclusive. It means, you need to find all numbers $x$ in his data structure that can meet $L \le x \le R$, and conduct the 'plus' one by one to get the 'sum'.

   *Sum* $L$ $R$

Can you help naive Bob to implement data struct 'Wish'?

## Input

At very beginning, there would be a number $\mathbf{T}$ indicates the number of tests. ( $1 \le \mathbf{T} \le 100$ )

For each test case:

The first line contains two integers $N$ and $M$ indicates the number of initial numbers and number of operstions. ($1 \le \mathbf{N} \le 10,000$ and $1 \le \mathbf{M} \le 10,000$)

The second line contains $N$ integers for the initial numbers $a_i$. ( $0 \le a_i \le 2 \times 10^9$ )

The following $M$ lines would follow the operation format #1 to #4.

For the operations, $0 \le L \le R \le 2 \times 10^9$ , $0 \le x \le 2 \times 10^9$

## Output

For each test case, output one line containing "Case #x:", where x is the test case number (starting from 1). For each operation *Query* and *Sum*, output the answer

## Example

| standard input | standard output |
|---|---|
| 1<br>4 5<br>1 2 88 29<br>Add 44<br>Query 71<br>Shift<br>Sum 0 10<br>Query 91 | Case #1:<br>90<br>4<br>99 |

## Note

For Sample Case #1:

- Initially: [1 , 2 , 29 , 88]
- Add 44: [1 , 2 , 29 , 88 , 44]
- Query 71: $29 + 71 = 90$; $88 + 71 = 59$; $44 + 71 = 15$
- Shift: [0 , 0 , 2 , 8 , 4]
- Sum 0 10: $0 + 0 + 2 + 8 + 4 = 4$
- Query 91: $91 + 8 = 99$