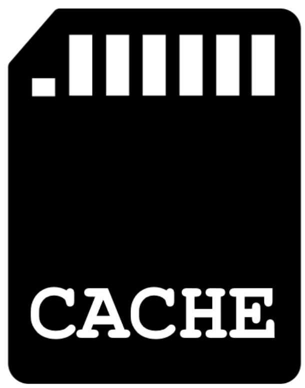
Problem K
Computer Cache
Time limit: 5 seconds



Your computer has a cache consisting of n different addresses, indexed from 1 to n. Each address can contain a single byte. The i^{th} byte is denoted as a_i. Initially all cache bytes start off with the value zero. Formally, the cache can be modeled by a byte array of length n that is initially all zeros.

You have m different pieces of data you want to store. The i^{th} piece of data is a byte array x_i of length s_i.

You are going to do q different operations on your computer. There are three types of operations:

1 i p Load data i starting at position p in the cache. Formally, this means set $a_p = x_{i,1}, a_{p+1} = x_{i,2}, \ldots, a_{p+s_i-1} = x_{i,s_i}$, where $x_{i,k}$ represents the kth byte of the array x_i. This overwrites any previously stored value in the cache. It is guaranteed that this is a valid operation (e.g. $s_i + p - 1 \le n$). It is possible for multiple versions of some data to be loaded in multiple positions at once.

2 p Print the byte that is stored in address p.

3 i l r Increment the l^{th} through r^{th} bytes in the i^{th} piece of data, modulo 256. Formally, this means to set $x_{i,k} = (x_{i,k} + 1) \mod 256$ for $l \le k \le r$. This does not affect values that are already loaded in the cache and only affects future loads.

Input

The first line of input consists of three numbers n, m, and q.

The following m lines consist of descriptions of the data, one per line. The following q lines consist of descriptions of operations, one per line.

It is guaranteed there is at least one type 2 print query operation in the input. Additionally:

$$1 \le n, m, q \le 5 \times 10^5$$
$$\sum_i s_i \le 5 \times 10^5$$
$$s_i \ge 1$$
$$0 \le x_{i,j} \le 255$$

Output

Your program must output the results for each type 2 operation, one integer value per line.

Explanation

`2 1`	Nothing has been put into the cache, so print 0
`1 2 2`	The cache is now [0, 1, 2, 1, 3]
`1 1 1`	The cache is now [255, 0, 15, 1, 3]
`2 1`	Print the first value of the cache which is 255
`2 4`	Print the fourth value of the cache which is 1
`3 1 1 2`	The first piece of data becomes [0, 1, 15]. The cache is still [255, 0, 15, 1, 3]
`2 1`	Print the first value of the cache which is 255.
`1 1 2`	The cache becomes [255, 0, 1, 15, 3].
`2 2`	Print the second value of the cache which is 0.
`2 5`	Print the fifth value of the cache which is 3.

Sample Input 1

```
5 2 10
3 255 0 15
4 1 2 1 3
2 1
1 2 2
1 1 1
2 1
2 4
3 1 1 2
2 1
1 1 2
2 2
2 5
```

Sample Output 1

```
0
255
1
255
0
3
```