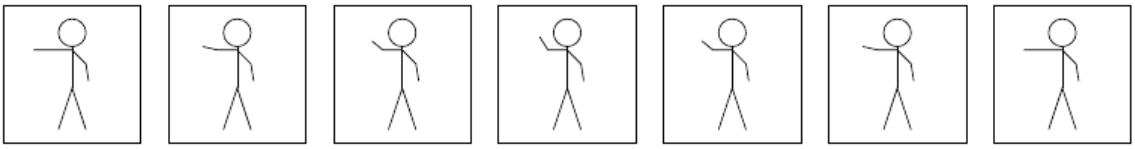
7547 Earl's Extremely Efficient Encryption

Earl has just finished writing an extremely efficient encryption algorithm for decreasing the size of video files. The algorithm has the capability to decrease the file size of a long video by many orders of magnitude. You can think of a video as a series of images.



One simple algorithm is to store each of these images individually. The secret of Earl's algorithm is to only use the differences between successive images in the series. He stores each transition in a 32-bit integer (these values are called the transition integers). Using an unencrypted version of the first image and all of the transition integers, you can fully recreate the video.

The issue is that calculating the transition integers is *lossy* (i.e. the 32-bit integer representation of the transition is only an approximation of the actual differences of the images). This makes it possible for errors to occur and the image to display incorrectly. Each transition on its own has a very high probability of working correctly, but the probability of being able to play the whole video without an error occurring is quite low. Moreover, the errors compound onto one another since the differences between the images in the actual video may not make sense for any images that Earl's algorithm has generated incorrectly.

To account for this issue, the algorithm will not only store the first image of the video, but a subset of the images. At each transition, the algorithm will check if the true (unencrypted) image is stored. If it is, then that image is displayed on the screen. If it is not stored, then the algorithm will use the transition integer to generate the next image.

The *badness* of a video is defined as the largest number of consecutive transition integers used. Earl's algorithm works for all videos consisting of at most L images. Given the number of images in the video collection and the subset of images Earl will include unencrypted, what is the badness value for each video?

Input

The input file contains several test cases, each of them as described below.

The input will begin with six integers: k ($1 \leq k \leq 100$), n ($1 \leq n \leq 10^5$), L ($1 \leq L \leq 10^9$), a ($0 \leq a \leq L$), b ($0 \leq b \leq L$) and g_1 ($0 \leq g_1 \leq L$). We then define $g_0 = 0$ and

$$g_i = (a \cdot g_{i-1} + b) \pmod{L+1} \text{ for all } i \in \{2, \ldots, n\}.$$

This is followed by k lines. The jth of these lines contains a single integer, w_j ($1 \leq w_j \leq L$), which represent the number of images in the j-th video when it is stored uncompressed. The subset of images Earl includes unencrypted is

$$\{g_i : 0 \leq i \leq n\} \cap \{0, 1, \ldots, w_j - 1\}.$$

The videos will be given in increasing order with respect to w_j.

Output

For each test case, output the badness for each video.

Explanation of Sample Input 1:

In the first sample input, the unencrypted images included are 0, 2, 4, 6, 8 and 10. For the video of length 1, you do not need any transition integers, so the badness is 0. For the video of length 3, you only need a transition integer to get from image 0 to image 1. For the video of length 7, you need to use a single transition integer on 4 separate occasions ($0 \to 1, 2 \to 3, 4 \to 5, 6 \to 7$). For a video of length 14, you need to use a transition integer to generate image 11, 12 and 13 (which corresponds to a badness of 3).

Sample Input

```
4 5 20 1 2 2
1
3
7
14
1 4 10 2 0 1
9
```

Sample Output

```
0
1
1
3
3
```