Stefan's New Year's resolutions (`manymax`)

Stefan has big plans for the year that just begun and among others, one of his plans is to improve his contest ratings. While every competitive programmer dreams of this, he is perfectly aware that not everyone can achieve it. Now he stumbled across an array that he received as a gift many years ago, together with a very old scroll containing the problem statement below.

Figure 1: Stefan discovering a long-lost problem.

You are given an array V of N positive integers, numbered from 0 to $N-1$, inclusive. You have to answer Q queries: given L, R and K, find the **product** of the largest K values of $V_L, V_{L+1}, \ldots, V_{R-1}, V_R$.

Stefan realized that this is not a cakewalk question so now it's your turn to help him overcome this challenge and write a program that answers the queries for him. Since the answers can be very big, you have to print them module $10^9 + 7$ (i.e. the reminder of the division by $10^9 + 7$).

> ☞ Among the attachments of this task you may find a template file `manymax.*` with a sample incomplete implementation.

> ✍ The *modulo* operation ($a \bmod m$) can be written in C/C++/Java/Python as (`a % m`). To avoid the *integer overflow* error, remember to reduce all partial results through the modulus, and not just the final result!
> *Notice that if $x < 10^9 + 7$, then $2x$ fits into a C/C++/Java* `int`.

Input

The input file consists of:

- a line containing integers N, Q.
- a line containing the N integers $V_0, \ldots, V_{N-1}$.
- Q lines, the i-th of which consisting of integers L_i, R_i, K_i.

Output

The output file must contain a single line consisting of the Q integers $\texttt{ans}_0, \ldots, \texttt{ans}_{Q-1}$.

Constraints

- $1 \leq N \leq 200\,000$.
- $1 \leq V_i \leq 200\,000$ for each $i = 0 \ldots N-1$.
- $0 \leq L \leq R < N$ for each query.
- $1 \leq K \leq R - L + 1$ for each query.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

- **Subtask 1** (0 points)   Examples.

- **Subtask 2** (11 points) $N, Q \leq 2000$.

- **Subtask 3** (13 points) $K = R - L + 1$ for all queries.

- **Subtask 4** (19 points) $K = 1$ for all queries.

- **Subtask 5** (25 points) $N, Q \leq 50\,000$.

- **Subtask 6** (32 points)   No additional limitations.

Examples

input	output
9 15 8 1 9 4 7 9 4 9 3 7 7 1 1 4 2 5 6 1 1 7 4 2 8 6 2 3 1 3 6 2 5 7 3 0 7 3 0 5 3 6 8 3 2 5 1 0 7 6 2 3 2 4 7 4	9 63 9 5103 81648 9 63 324 729 648 108 9 163296 36 2268

Explanation

For the **second query** of the sample case, the values between 1 and 4 are $[1, 9, 4, 7]$, the 2 largest values are 9 and 7 and their product is 63.

For the **fourth query** of the sample case, the values between 1 and 7 are $[1, 9, 4, 7, 9, 4, 9]$, the 4 largest values are 9, 9, 9 and 7 and their product is 5103.