

Climbing the Leaderboard

An arcade game player wants to climb to the top of the leaderboard and track their ranking. The game uses [Dense Ranking](#), so its leaderboard works like this:

- The player with the highest score is ranked number **1** on the leaderboard.
- Players who have equal scores receive the same ranking number, and the next player(s) receive the immediately following ranking number.

Example

ranked = [100, 90, 90, 80]
player = [70, 80, 105]

The ranked players will have ranks **1**, **2**, **2**, and **3**, respectively. If the player's scores are **70**, **80** and **105**, their rankings after each game are **4th**, **3rd** and **1st**. Return [4, 3, 1].

Function Description

Complete the *climbingLeaderboard* function in the editor below.

climbingLeaderboard has the following parameter(s):

- *int ranked[n]*: the leaderboard scores
- *int player[m]*: the player's scores

Returns

- *int[m]*: the player's rank after each new score

Input Format

The first line contains an integer *n*, the number of players on the leaderboard.

The next line contains *n* space-separated integers *ranked[i]*, the leaderboard scores in decreasing order.

The next line contains an integer, *m*, the number games the player plays.

The last line contains *m* space-separated integers *player[j]*, the game scores.

Constraints

- $1 \leq n \leq 2 \times 10^5$
- $1 \leq m \leq 2 \times 10^5$
- $0 \leq ranked[i] \leq 10^9$ for $0 \leq i < n$
- $0 \leq player[j] \leq 10^9$ for $0 \leq j < m$
- The existing leaderboard, *ranked*, is in *descending* order.

- The player's scores, *player*, are in *ascending* order.

Subtask

For 60% of the maximum score:

- $1 \leq n \leq 200$
- $1 \leq m \leq 200$