5 Maximum in Sliding Window

Problem Introduction

Given a sequence $a_1, \ldots, a_n$ of integers and an integer $m \leq n$, find the maximum among $\{a_i, \ldots, a_{i+m-1}\}$ for every $1 \leq i \leq n - m + 1$. A naive $O(nm)$ algorithm for solving this problem scans each window separately. Your goal is to design an $O(n)$ algorithm.

Problem Description

Input Format. The first line contains an integer n, the second line contains n integers $a_1, \ldots, a_n$ separated by spaces, the third line contains an integer m.

Constraints. $1 \leq n \leq 10^5$, $1 \leq m \leq n$, $0 \leq a_i \leq 10^5$ for all $1 \leq i \leq n$.

Output Format. Output $\max\{a_i, \ldots, a_{i+m-1}\}$ for every $1 \leq i \leq n - m + 1$.

Time Limits.

language	`C`	`C++`	`Java`	`Python`	`C#`	`Haskell`	`JavaScript`	`Ruby`	`Scala`
time (sec)	1	1	1.5	5	1.5	2	5	5	3

Memory Limit. 512MB.

Sample 1.

Input:

```
8
2 7 3 1 5 2 6 2
4
```

Output:

```
7 7 5 6 6
```

What to Do

We give hints for three different solutions.

1. *Implement a queue using two stacks.* Use a queue data structure for sliding a window through a sequence: for shifting a window one position to the right, pop the leftmost element of the queue and push a new element from the new window. A queue can be implemented using two stacks such that each queue operation takes constant time *on average*. Then, use your implementation of the stack with maximum.

2. *Preprocess block suffixes and prefixes.* Partition the input sequence into blocks of length m and precompute the maximum for every suffix and every prefix of each block. Afterwards, the maximum in each sliding window can be found by considering a suffix and a prefix of two consecutive blocks.

3. *Store relevant items in a dequeue.* Use a double-ended queue (dequeue) to store elements of the current window. At the same time, store only relevant elements: before adding a new element drop all smaller elements.

Need Help?

Ask a question or see the questions asked by other learners at this forum thread.