Previous Next

406. Queue Reconstruction by Height

July 27, 2019 | 30.9K views

★★★★★
Average Rating: 4.66 (53 votes)

Suppose you have a random list of people standing in a queue. Each person is described by a pair of integers `(h, k)`, where `h` is the height of the person and `k` is the number of people in front of this person who have a height greater than or equal to `h`. Write an algorithm to reconstruct the queue.

Note:
The number of people is less than 1,100.

Example

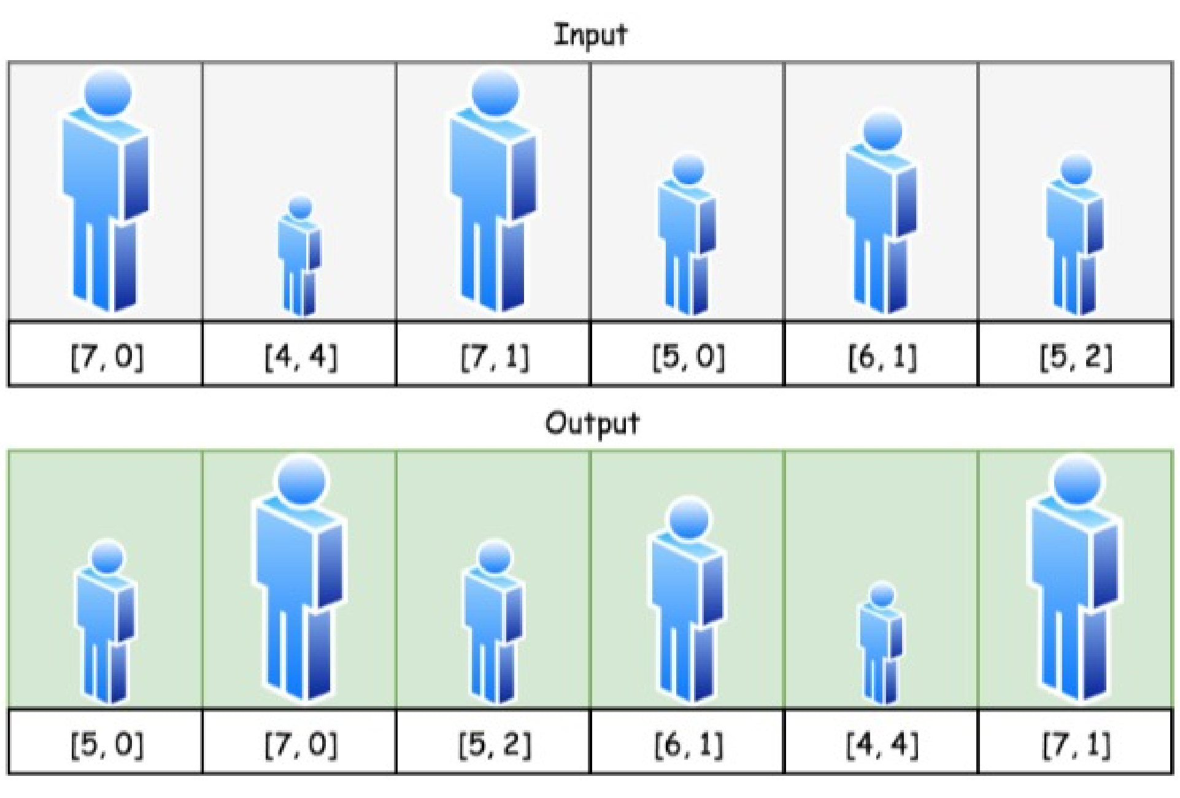
```
Input:
[[7,0], [4,4], [7,1], [5,0], [6,1], [5,2]]

Output:
[[5,0], [7,0], [5,2], [6,1], [4,4], [7,1]]
```

Solution

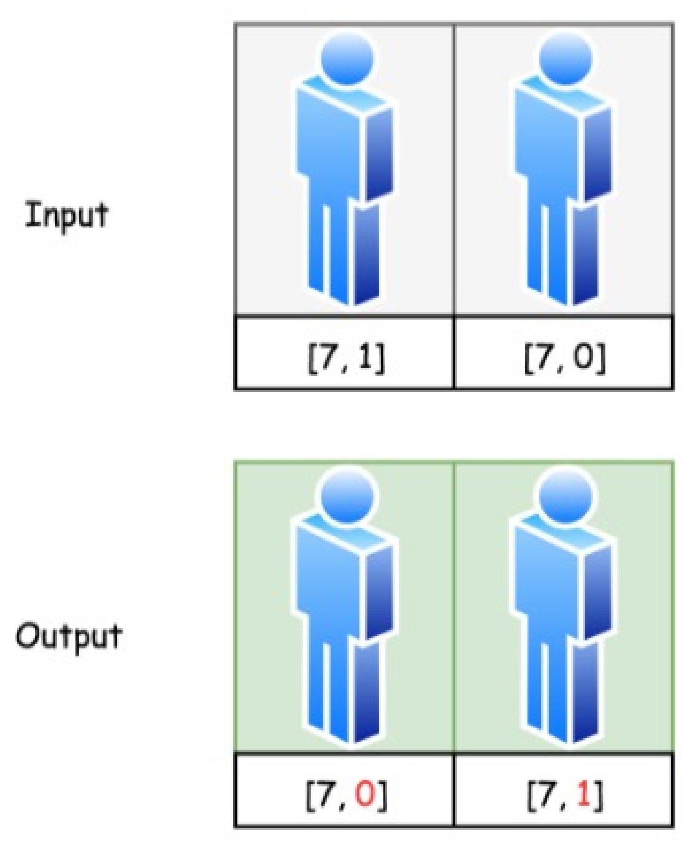
Approach 1: Greedy

Intuition

The problem is to reconstruct the queue.

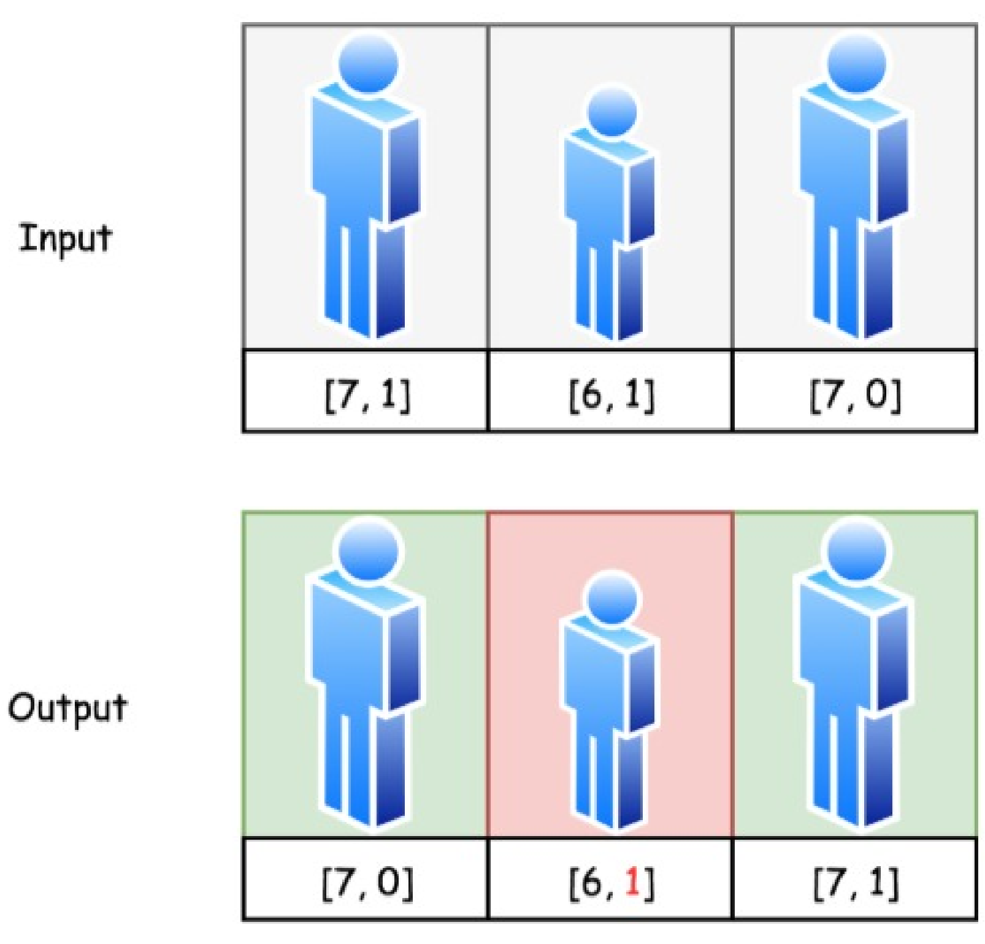


Let's start from the simplest case, when all guys (h, k) in the queue are of the same height h, and differ by their k values only (the number of people in front who have a greater or the same height). Then the solution is simple: each guy's index is equal to his k value. The guy with zero people in front takes the place number 0, the guy with 1 person in front takes the place number 1, and so on and so forth.



This strategy could be used even in the case when not all people are of the same height. The smaller persons are "invisible" for the taller ones, and hence one could first arrange the tallest guys as if there was no one else.

Let's now consider a queue with people of two different heights: 7 and 6. For simplicity, let's have just one 6-height guy. First follow the strategy above and arrange guys of height 7. Now it's time to find a place for the guy of height 6. Since he is "invisible" for the 7-height guys, he could take whatever place without disturbing 7-height guys order. However, for him the others are visible, and hence he should take the position equal to his k-value, in order to have his proper place.



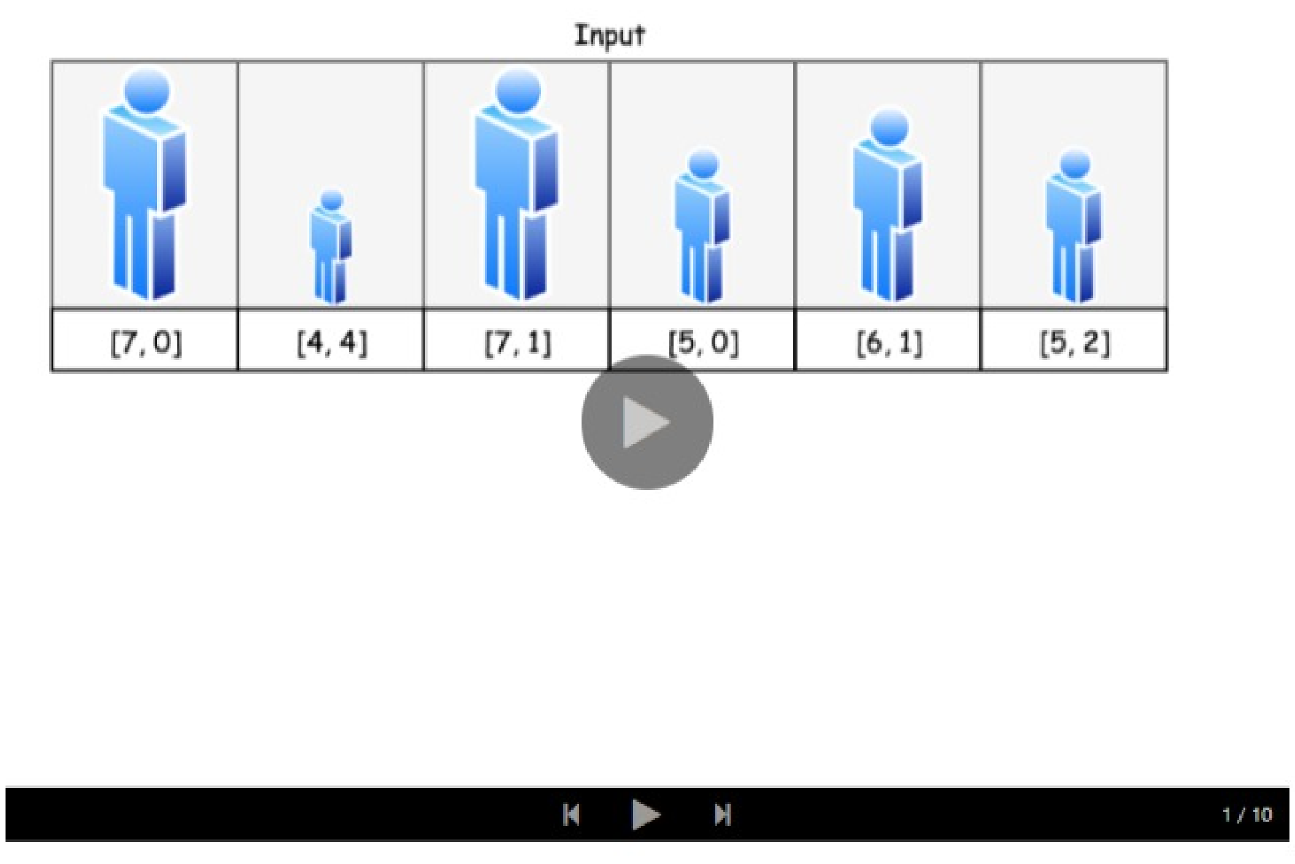
> This idea is easy to extend for the case of numerous guys of height 6. Just sort them by k-values, as it was done before for 7-height guys, and insert them one by one on the positions equal to their k-values.

The following strategy could be continued recursively:

- Sort the tallest guys in the ascending order by k-values and then insert them one by one into output queue at the indexes equal to their k-values.

- Take the next height in the descending order. Sort the guys of that height in the ascending order by k-values and then insert them one by one into output queue at the indexes equal to their k-values.

- And so on and so forth.



Algorithm

- Sort people:
  - In the descending order by height.
  - Among the guys of the same height, in the ascending order by k-values.
- Take guys one by one, and place them in the output array at the indexes equal to their k-values.

- Return output array.

Implementation

Java Python Copy

```
class Solution:
    def reconstructQueue(self, people: List[List[int]]) -> List[List[int]]:
        people.sort(key = lambda x: (-x[0], x[1]))
        output = []
        for p in people:
            output.insert(p[1], p)
        return output
```

Complexity Analysis

- Time complexity : $\mathcal{O}(N^2)$. To sort people takes $\mathcal{O}(N \log N)$ time. Then one proceeds to n insert operations, and each takes up to $\mathcal{O}(k)$ time, where k is a current number of elements in the list. In total, one needs up to $\mathcal{O}\left(\sum\limits_{k=0}^{N-1} k\right)$ time, i.e. up to $\mathcal{O}(N^2)$ time.

- Space complexity : $\mathcal{O}(N)$ to keep the output.

Rate this article: ★★★★★

Previous Next

Comments: 27 Sort By ▾

Type comment here... (Markdown is supported)

Preview Post

jwzz ★ 185 October 22, 2019 12:32 PM

Hint seems trash

121 Share Reply

SHOW 5 REPLIES

Gabriel-18 ★ 113 August 10, 2019 5:19 PM

After reading, I am still confuse about the algorithm... and more confused about output.add(p[1], p); is there anybody can tell what p[1] means here... what this code means...

17 Share Reply

SHOW 4 REPLIES

pbu ★ 316 June 6, 2020 8:17 PM

one more solution code that is easier to memorize than to understand it.

14 Share Reply

rbpradeep ★ 54 February 3, 2020 11:08 PM

I'm trying to figure out why this algorithm is considered greedy. Whats the optimal substructure here ?

13 Share Reply

SHOW 1 REPLY

droconnel22 ★ 14 April 18, 2020 6:49 AM

Took me 3 hours to basically fail miserably. Tried recursion, then Merge Sort, then bubble sort with linked list insertion.

11 Share Reply

SHOW 2 REPLIES

kchou650 ★ 27 November 7, 2019 2:10 PM

the hints were awful. it implied that the algo had to start with the shortest people first. which i wasn't able to think up of one.

16 Share Reply

SHOW 3 REPLIES

loona ★ 4 June 1, 2020 3:18 AM

Can someone please help me understand why it is accurate to classify this solution as a `Greedy` algorithm?

4 Share Reply

SHOW 1 REPLY

theodesp ★ 13 August 2, 2019 9:12 PM

Javascript:

```
var reconstructQueue = function(people) {
  const result = [];
  const sortedByHeight = people.sort((a, b) => {
```

Read More

3 Share Reply

SHOW 1 REPLY

mahtab_alam ★ 19 June 17, 2020 9:01 PM

This might help, in understanding the solution.

Approach :

1. First sort the input array people in such a way that, persons are sorted in descending order of

Read More

2 Share Reply

Sabunt ★ 2 July 31, 2019 2:02 PM

```
#ruby code
def reconstruct_queue(people)
  people.sort! do |a, b|
    a[0] == b[0] ? a[1] - b[1] : b[0] - a[0]
```

Read More

2 Share Reply

‹ 1 2 3 ›