LeetCode 0406 根据身高重建队列

题目

假设有打乱顺序的一群人站成一个队列。 每个人由一个整数对`(h, k)`表示，其中`h`是这个人的身高，`k`是排在这个人前面且身高大于或等于`h`的人数。 编写一个算法来重建这个队列。

注意： 总人数少于1100人。

示例

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

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

思路1(离散化+树状数组 非最优)

离散化+树状数组。

可以通过树状数组统计已经排好序的身高的情况，在新的样本进来的时候，可以快速统计比当前身高高的人的个数。会有很多个符合条件的样本，在其中选择身高最低的那一个加入已排序序列。

为什么选择身高最低的: 插入身高高的样本，后续身高第的样本都不符合条件。当前是`n`，插入后是`n+1`那后续这个样本不能被选择。

由于数组更新时间复杂度较高差点超时。

```python
class BIT:
    def __init__(self, n):
        self.bit = [0] * (n + 1)

    def _low_bit(self, n):
        return n & -n

    def insert(self, n):
```

```
        while n < len(self.bit):
            self.bit[n] += 1
            n += self._low_bit(n)

    def query(self, n):
        ret = 0
        while n > 0:
            ret += self.bit[n]
            n -= self._low_bit(n)
        return ret

class Solution:
    def reconstructQueue(self, people: List[List[int]]) -> List[List[int]]:
        if not people:
            return []
        heights = set()
        for p in people:
            heights.add(p[0])
        heights = list(heights)
        heights.sort(reverse=True)
        d = {}
        for idx, val in enumerate(heights):
            d[val] = idx + 1
        n = len(heights)
        bit = BIT(n)
        ret = []
        indexs = set(range(len(people)))
        while indexs:
            select_idx = -1
            for idx in indexs:
                h, k = d[people[idx][0]], people[idx][1]
                if bit.query(h) == k:
                    if select_idx == -1 or people[select_idx][0] > people[idx][0]:
                        select_idx = idx
            indexs.remove(select_idx)
            ret.append(people[select_idx])
            bit.insert(d[people[select_idx][0]])
        return ret
```

思路二(排序+贪心)

按照身高有高到低排序，身高相同的话`k`由小到大排序

遍历数组，第`i`个人一直身高比他高的有`k`个人，那么他在当先排序数组的`k+1个位置

```python
class Solution:
    def reconstructQueue(self, people: List[List[int]]) -> List[List[int]]:
        if not people:
            return []
        people.sort(key=lambda x:[-x[0], x[1]])
        ret = []
        for p in people:
            ret.insert(p[1], p)
        return ret
```