

题目描述：

让我们来模拟一个消息队列的运作，有一个发布者和若干消费者，发布者会在给定的时刻向消息队列发送消息，若此时消息队列有消费者订阅，这个消息会被发送到订阅的消费者中优先级最高（输入中消费者按优先级升序排列）的一个；若此时没有订阅的消费者，该消息被消息队列丢弃。消费者则会在给定的时刻订阅消息队列或取消订阅。

当消息发送和订阅发生在同一时刻时，先处理订阅操作，即同一时刻订阅的消费者成为消息发送的候选。

当消息发送和取消订阅发生在同一时刻时，先处理取消订阅操作，即消息不会被发送到同一时刻取消订阅的消费者。

输入描述：

输入为两行。

第一行为  $2N$  个正整数，代表发布者发送的  $N$  个消息的时刻和内容（为方便解析，消息内容也用正整数表示）。第一个数字是第一个消息的发送时刻，第二个数字是第一个消息的内容，以此类推。用例保证发送时刻不会重复，但注意消息并没有按照发送时刻排列。

第二行为  $2M$  个正整数，代表  $M$  个消费者订阅和取消订阅的时刻。第一个数字是第一个消费者订阅的时刻，第二个数字是第一个消费者取消订阅的时刻，以此类推。用例保证每个消费者的取消订阅时刻大于订阅时刻，消费者按优先级升序排列。

两行的数字都由空格分隔。 $N$  不超过 100， $M$  不超过 10，每行的长度不超过 1000 字符。

输出描述：

输出为  $M$  行，依次为  $M$  个消费者收到的消息内容，消息内容按收到的顺序排列，且由空格分隔；若某个消费者没有收到任何消息，则对应的行输出-1。

补充说明：

```
示例1
输入: 2 22 1 11 4 44 5 55 3 33
      1 7 2 3
输出: 11 22 44 55
      22
说明: 消息11在1时刻到达，此时只有第一个消费者订阅，消息发送给它；消息22在2时刻到达，此时两个消费者都订阅了，消息发送给优先级最高的第二个消费者；消息33在3时刻到达，此时只有第一个消费者订阅，消息发送给它；余下的消息按规则也是发送给第一个消费者。

示例2
输入: 5 64 11 64 9 97
      9 11 4 9
输出: 97
      64
说明: 消息64在5时刻到达，此时只有第二个消费者订阅，消息发送给它；消息97在9时刻到达，此时只有第一个消费者订阅（因为第二个消费者刚好在9时刻取消订阅），消息发送给它；11时刻也到达了一个内容为64的消息，不过因为没有消费者订阅，消息被丢弃。
```

```

1 arr1=list(map(int,input().split()))
2 arr2=list(map(int,input().split()))
3
4
5 producers=[] # 记录每条消息发送时间和发送内容
6 for i in range(0,len(arr1),2):
7     producers.append([arr1[i],arr1[i+1]])
8 producers.sort(key= lambda x:x[0])
9
10 customers=[] # 记录每个顾客订阅时间和取消订阅时间
11 for i in range(0,len(arr2),2):
12     customers.append([arr2[i],arr2[i+1]])
13
14
15 res=[[ ] for i in range(len(customers))]
16
17 for i in producers:
18     time,context=i
19
20     for j in range(len(customers)-1,-1,-1): # 消息只会发送给优先级最高的消费者，倒序遍历
21         subtime, unsubtime=customers[j]
22
23         if subtime<=time<unsubtime:
24             res[j].append(context)
25             break
26
27 for i in res:
28     if len(i)==0:
29         print('-1')
30
31     else:
32         print(' '.join([str(x) for x in i]))

```