Invalid Transaction

A transaction is possibly invalid if:

- the amount exceeds `$1000`, or;
- if it occurs within (and including) `60` minutes of another transaction with the **same name** in a **different city**.

You are given an array of strings `transaction` where `transactions[i]` consists of comma-separated values representing the name, time (in minutes), amount, and city of the transaction.

Return a list of `transactions` that are possibly invalid. You may return the answer in **any order**.

Example

```
Input: transactions = ["alice,20,800,mtv","alice,50,100,beijing"]
Output: ["alice,20,800,mtv","alice,50,100,beijing"]
```

Brute Force, $O(n^2)$

```
class Transaction:
    def __init__(self, name, time, amount, city):
        self.name = name
        self.time = int(time)
        self.amount = int(amount)
        self.city = city

from collections import defaultdict
class Solution:
    def invalidTransactions(self, transactions):
        transactions = [Transaction(*transaction.split(',')) for transaction in transactions]
        transactions.sort(key=lambda t: t.time) # O(nlogn) time

        trans_indexes = defaultdict(list)
        for i, t in enumerate(transactions): # O(n) time
            trans_indexes[t.name].append(i)

        res = []
        for name, indexes in trans_indexes.items(): # O(n) time
            left = right = 0
            for i in range (len(indexes)):
                # 实际transaction
                # i 是第i个transaction
                t = transactions[indexes[i]]
                if (t.amount > 1000):
                    res.append("{},{},{},{}".format(t.name, t.time, t.amount, t.city))
                    continue
                while left < len(indexes)-1 and transactions[indexes[left]].time < t.time - 60: # O(60) time
                    left += 1
                while right < len(indexes)-1 and transactions[indexes[right+1]].time <= t.time + 60: # O(60) time
                    right += 1
                for j in range(left,right+1): # O(120) time
                    if transactions[indexes[j]].city != t.city:
                        res.append("{},{},{},{}".format(t.name, t.time, t.amount, t.city))
                        break

        return res
```