Problem 332: Reconstruct Itinerary

Problem Information

Difficulty: Hard
Acceptance Rate: 44.01%
Paid Only: No
Tags: Depth-First Search, Graph, Eulerian Circuit

Problem Description

You are given a list of airline `tickets` where `tickets[i] = [fromi, toi]` represent the departure and the arrival airports of one flight. Reconstruct the itinerary in order and return it.

All of the tickets belong to a man who departs from `"JFK"`, thus, the itinerary must begin with `"JFK"`. If there are multiple valid itineraries, you should return the itinerary that has the smallest lexical order when read as a single string.

* For example, the itinerary `["JFK", "LGA"]` has a smaller lexical order than `["JFK", "LGB"]`.

You may assume all tickets form at least one valid itinerary. You must use all the tickets once and only once.

Example 1:

![](https://assets.leetcode.com/uploads/2021/03/14/itinerary1-graph.jpg)

Input: tickets = [["MUC","LHR"],["JFK","MUC"],["SFO","SJC"],["LHR","SFO"]] **Output:** ["JFK","MUC","LHR","SFO","SJC"]

Example 2:

![](https://assets.leetcode.com/uploads/2021/03/14/itinerary2-graph.jpg)

Input: tickets = [["JFK","SFO"],["JFK","ATL"],["SFO","ATL"],["ATL","JFK"],["ATL","SFO"]] **Output:** ["JFK","ATL","JFK","SFO","ATL","SFO"] **Explanation:** Another possible reconstruction is ["JFK","SFO","ATL","JFK","ATL","SFO"] but it is larger in lexical order.

Constraints:

* `1 <= tickets.length <= 300` * `tickets[i].length == 2` * `fromi.length == 3` * `toi.length == 3` * `fromi` and `toi` consist of uppercase English letters. * `fromi != toi`

Code Snippets

C++:

```
class Solution {
public:
vector<string> findItinerary(vector<vector<string>>& tickets) {

}
};
```

Java:

```
class Solution {
public List<String> findItinerary(List<List<String>> tickets) {

}
}
```

Python3:

```
class Solution:
def findItinerary(self, tickets: List[List[str]]) -> List[str]:
```