reconstruct itinerary : Hard sum.

Goal: Start from JFK and cover the route (for all ticket) in lexical order (if we have multiple choices from point).

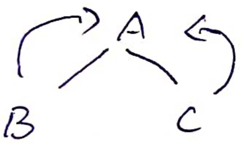
Return the order of travel.

Cases:

1. Ticket are only 1 way. that mean directed graph.

$\boxed{A \to B}$ ✓ $\boxed{A - B}$ ✗

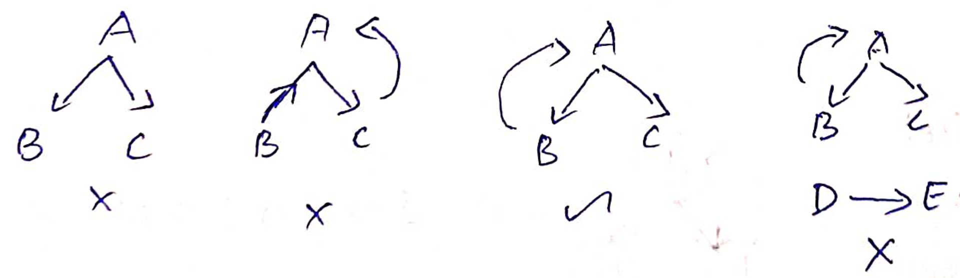
2. Return route with smallest lexical order.



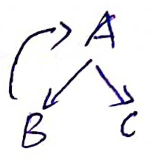
$A \to \overset{\text{small}}{\underline{B}} \to A \to C \to A$ ✓

$A \to \underline{C} \to A \to B \to A$ ✗

3. All ticket form a valid route. (graph must single component)



4. all ticket use only onces.



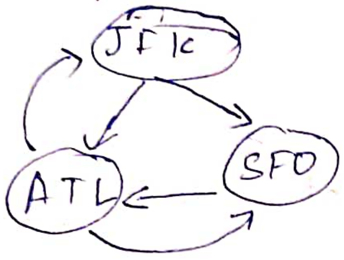
$A \to B \to A \to C$ ✓

$A \to B - \underbrace{A \to B}_{\text{repeat}} \to A \to C$ ✗

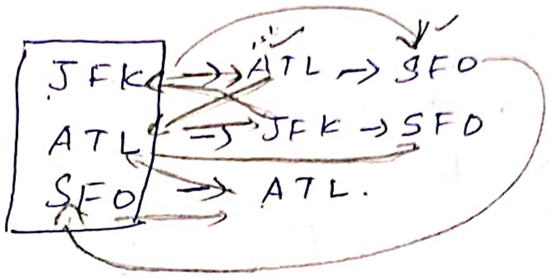
choice of DS

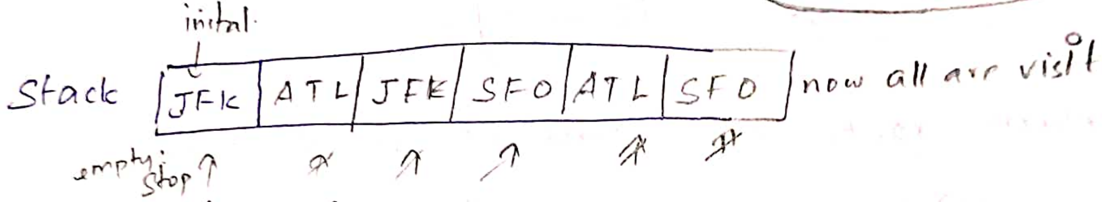
1) Multiset → keep value arranged
2) Map → For O(1) search time for given key.
3) Stack → used to traversal.
4) string vector → store final answer & reversed it

ex: Graph



adjust. list using unordered map <string, multiset>





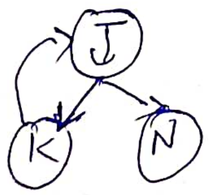
answer vector : SFO → ATL → SFO → JFK → ATL → JFK.

reverse it

∴ JFK → ATL → JFK → SFO → ATL → SFO.

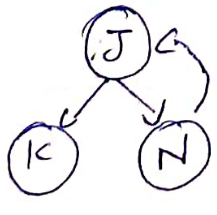
Addition case

Map	Multiset
J	K → N
K	J
N	



→ J → K → J → N ✓

J	K → N
K	
N	J



But here

× $\boxed{J \to K}$ → dead state. u cant read right part.

So that now u conside valid root first and then lexical order.

∴ $\boxed{J \to N \to J \to K}$ ✓

332. Reconstruct Itinerary

Solved ✓

Hard   Topics   Companies

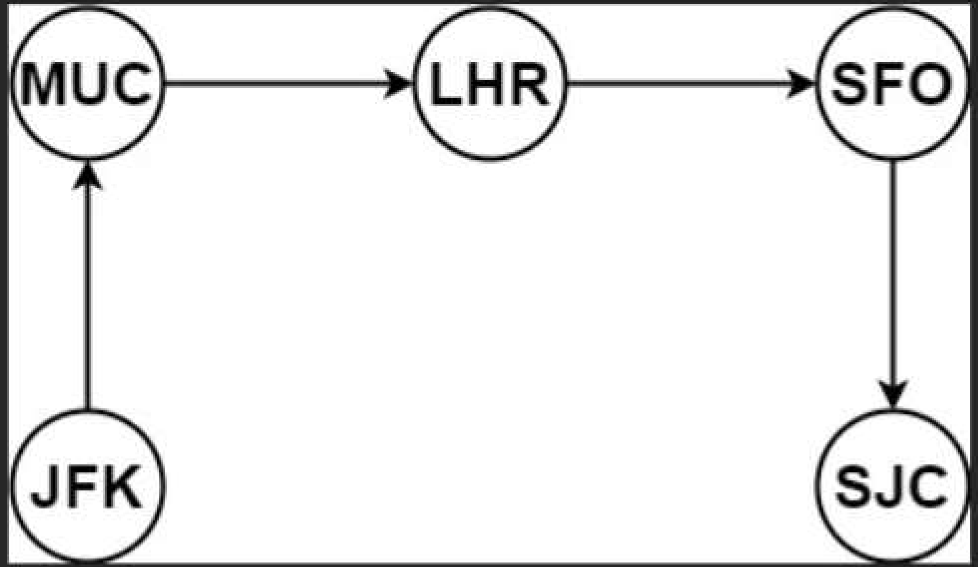
You are given a list of airline `tickets` where `tickets[i] = [from_i, to_i]` represent the departure and the arrival airports of one flight. Reconstruct the itinerary in order and return it.

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:



C++   Auto

```cpp
class Solution {
public:
    vector<string> findItinerary(vector<vector<string>>& ticket) {
        unordered_map<string,multiset<string>>abj;// abjactancy list using map ,here string is key and multiset manage
        //the element in  sorted manner
        for(int i=0;i<ticket.size();i++)
        {
            abj[ticket[i][0]].insert(ticket[i][1]);
        }
        vector<string>ans;
        stack<string>st;
        st.push("JFK");
        while(!st.empty())
        {
            string origin=st.top();
            if(abj[origin].size()==0) // that mean all adjustacy list are visted or multiset is empty
            {
                ans.push_back(origin);
                st.pop();
            }
            else //not visited
            {
                auto desgination=abj[origin].begin() ; //take first value in  mulitset (it have multiple value in  list)
                st.push(*desgination); // get the actual string value from the iterator to push onto the stack.
                abj[origin].erase(desgination); //erase is designed to take an iterator and remove the element at that position
                // that why not use * pointer
            }
        }
        reverse(ans.begin(),ans.end());
        return ans;
    }
};
```