We have solved this problem by using graph data structure, where each city is represented as a node, and the available train tickets between cities are represented as edges.

Algorithm:

Step1: Create a graph with cities as nodes and available tickets as edges

Step 2: Start from Kiev node and perform a depth-first search to visit all nodes that can be reached from the current node.

Step 3: During Depth First Search, it keep track of the visited cities to avoid visiting them again.

Step 4: If a city has already been visited, mark it as an end and return back to the previous city.

Step 5: When all cities have been visited, the route of the son's journey is obtained by tracing back the path from the last visited city to the first city (Kiev).

This algorithm will give you the correct route which your son took in the form of a list of cities in the order they were visited

Explanation of code:

The algorithm useda graph data structure that represent the cities and available train tickets between them. The cities are represented as nodes in graph and the available train tickets are as edges.

This algorithm starts by creating a graph where cities as nodes and available tickets as edges. Then, it performs a depth-first search (DFS) starting from the city of Kiev. The DFS function takes a city as an input and visits all the cities that can be reached from the current city. During the DFS, the algorithm keeps track of the visited cities in a set visited to avoid visiting a city multiple times. If a city has already been visited, it is marked as a dead end and the DFS returns to the previous city.

When all cities have been visited, the route of the son's journey is obtained by tracing back the path from the last visited city to the first city (Kiev). This is done by maintaining a list route of cities in the order they were visited.

Finally, the output of the algorithm is the route list, which contains the route of the son's journey in the order he visited the cities.

Top of Form

Bottom of Form

Top of Form

Bottom of Form