Tour recommender

Suppose, a friend of yours comes to the US for the first time and he/she wants to visit several places in the US. Now, you know the interesting things and places about the 50 states of the country. You ask you friend about his/her interests and then you suggest him/her to visit some of the states based on the interests.

In this project, you have to act as a tour guide (in fact just a recommender). You are provided with a file ***'states.csv'*** containing the interesting things about each states in the US. You are also provided with a chart file ***'dist.csv'*** which contains state to state distance in a tabular format.

You are given with the **main** function:

```
def main():
    stateInfo = read_state_file( 'states.csv' )
    itemDict = build_itemDict( stateInfo )
    personDatabase = read_person_file( 'person.csv' )
    suggestion = make_suggestion( personDatabase, itemDict )
    distanceMap = build_dist_map( 'dist.csv' )
    (route, driveDist ) = calculate_min_route( distanceMap, suggestion['Someone'] )

    print( route, driveDist )
```

As you can see, you have to use several functions in the main function, they will have to be defined by you. All the functions are described below:

- **read_state_file**
    - parameter: filename - string
    - returns: dictionary containing the interesting things about the states
    - description: reads file and stores the interesting things as values against each state as key in a dictionary
    - example:
        - {'Alaska': ['wildlife', 'fishing', 'mountains', 'wilderness', 'glaciers', 'water sports', 'mountain sports'],
        - 'Georgia': ['wilderness'],
        - 'Louisiana': ['food', 'casinos'],
        - 'Maryland': ['food', 'sports'],
        - 'New York': ['business & industry', 'history', 'cityscape', 'food']}
- **build_itemDict**
    - parameter: stateInfo – dictionary
    - returns: dictionary containing interesting things as keys and states as values
    - description: takes in the stateInfo dictionary and builds an itemDict dictionary which gathers all the states that include specific interesting thing as common
    - example:
        - {'wildlife': ['Alaska'],
        - 'fishing': ['Alaska'],
        - 'mountains': ['Alaska'],
        - 'wilderness': ['Alaska', 'Georgia'],
        - 'glaciers': ['Alaska'],
        - 'water sports': ['Alaska'],

        - 'mountain sports': ['Alaska'],
        - 'food': ['Louisiana', 'Maryland', 'New York'],
        - 'casinos': ['Louisiana'],
        - 'sports': ['Maryland'],
        - 'business & industry': ['New York'],
        - 'history': ['New York'],
        - 'cityscape': ['New York']}
- **read_person_file**
    - parameter: filename - string
    - returns: dictionary containing the interesting things of persons
    - description: reads file and stores the interesting things as values against each person as key in a dictionary
    - example:
        - {'Someone': ['beaches', 'mountains', 'mountain sports', 'fishing', 'sports'],
        - 'Anyone': ['mines', 'glaciers', 'wildlife', 'wilderness'],
        - 'None': ['food', 'cityscape'],
        - 'Somebody': ['casinos', 'beaches', 'water sports', 'springs'],
        - 'Nobody': ['history', 'cityscape', 'parks', 'lakes'],
        - 'Everybody': ['food', 'waterfalls', 'farms', 'wilderness', 'wildlife', 'desert']}
- **make_suggestion**
    - parameter: personDatabase – dictionary, itemDict - dictionary
    - returns: dictionary containing the suggested states for each person in the personDatabase
    - description: looks for person's interest and searches for the item in the itemDict, several interest generates several sets of collection of states, then performs a union operation and sets the ***sorted*** list of state as the value for the person as key in the returned dictionary
    - example:
        - {'Someone': ['Alabama', 'Alaska', 'Arizona', 'California', 'Colorado', 'Delaware', 'Florida', 'Hawaii', 'Maryland', 'Massachusetts', 'Minnesota', 'Nebraska', 'New Hampshire', 'North Carolina', 'Ohio', 'South Carolina', 'Tennessee', 'Texas', 'Utah', 'Vermont', 'Washington'],
        - 'Anyone': ['Alaska', 'Arizona', 'California', 'Colorado', 'Connecticut', 'Florida', 'Georgia', 'Idaho', 'Maine', 'Montana', 'North Dakota', 'Pennsylvania', 'South Dakota', 'Texas', 'Vermont', 'Washington', 'West Virginia', 'Wyoming'],
        - 'None': ['Illinois', 'Kentucky', 'Louisiana', 'Maine', 'Maryland', 'Michigan', 'New Mexico', 'New York', 'North Carolina', 'Pennsylvania', 'Rhode Island', 'Tennessee', 'Texas', 'Vermont', 'Virginia', 'Washington DC', 'Wisconsin'],
        - 'Somebody': ['Alaska', 'Arkansas', 'California', 'Colorado', 'Connecticut', 'Delaware', 'Florida', 'Hawaii', 'Idaho', 'Louisiana', 'Maine', 'Minnesota', 'Mississippi', 'Missouri', 'Nevada', 'New Jersey', 'North Carolina', 'Ohio', 'Oregon', 'Rhode Island', 'South Carolina', 'Texas', 'Washington', 'West Virginia', 'Wyoming'],
        - 'Nobody': ['Alabama', 'Arkansas', 'Florida', 'Hawaii', 'Illinois', 'Massachusetts', 'Michigan', 'Minnesota', 'Missouri', 'New Hampshire', 'New Jersey', 'New Mexico', 'New York', 'Utah', 'Vermont', 'Virginia', 'Washington DC', 'Wisconsin'],
        - 'Everybody': ['Alaska', 'Arizona', 'California', 'Connecticut', 'Florida', 'Georgia', 'Hawaii', 'Idaho', 'Illinois', 'Indiana', 'Iowa', 'Kansas', 'Kentucky', 'Louisiana', 'Maine', 'Maryland', 'Michigan', 'Mississippi', 'Montana', 'Nebraska', 'Nevada', 'New Mexico', 'New York', 'North Carolina', 'Oregon', 'Pennsylvania', 'Rhode Island', 'Tennessee', 'Texas', 'Vermont', 'Virginia', 'Washington', 'Wisconsin', 'Wyoming']}
- **build_dist_map**
    - parameter: filename - string
    - returns: dictionary containing distance from each state to another

    - description: reads the file and stores the distance of a state from all other states in a dictionary, this dictionary is set as the value for the state for which the distance is measured as key for the returned dictionary
    - example:
        - {'New York': {'New York': '', 'Alaska': '4596.65', 'Georgia': '1498.22', 'Maryland': '346.17', 'Louisiana': '2017.01'},
        - 'Alaska': {'New York': '4706.62', 'Alaska': '', 'Georgia': '4546.65', 'Maryland': '4608', 'Louisiana': '4863.3'},
        - 'Georgia': {'New York': '1453.81', 'Alaska': '5363.31', 'Georgia': '', 'Maryland': '775.28', 'Louisiana': '552.38'},
        - 'Maryland': {'New York': '344.92', 'Alaska': '4542.12', 'Georgia': '750.95', 'Maryland': '', 'Louisiana': '1585.65'},
        - 'Louisiana': {'New York': '1979.42', 'Alaska': '5025.53', 'Georgia': '555.13', 'Maryland': '1629.74'}}

- **calculate_min_route**
    - parameter: distanceMap – dictionary, suggestion - list
    - returns: tuple containing a string ***route*** and a number ***driveDist***
    - description: starting from the first state in the list, calculates and selects the closest state and then from that state again the next closest state and so on, builds a string comprising all the states selected sequentially and returns the total driving distance calculated
    - example:
        - for a person having interest in 'food', he/she should visit `Louisiana > Maryland > New York`
        - the driving distance he/she has to cover is `1974.66`