Pseudocode description:

Z = 8 #number of nodes

I,j,k used in for loop to visit the each node

x,y source and destination node

MIN_PROBABILITY = 0   #to store the minimum probability of particular node

maps = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H'] # nodes name

Pseudocode for the given problems:

```
# project function to find the maximum  distance between the source and destination node.
def project(graph,direction):
   solve = list(map(lambda i: list(map(lambda j: j, i)), graph))
    path = list(map(lambda i: list(map(lambda j: j, i)), direction))
    for k in range(Z):
        for i in range(Z):
            for j in range(Z):
                if solve[i][j]<solve[i][k] * solve[k][j]:
                    path[i][j] =  path[i][k] + path[k][j]:

 ## to select the maximum distance between the two nodes via using the junction
            solve[i][j] = max(solve[i][j], solve[i][k] * solve[k][j])

## to find the maximum probability of a city
    maxProb = 0       ## initializing the maximum probability to 0
    ans = ' '                  ## to store the final answer and initialized it as NULL
    for i in range(Z):
        tmp = 1  ## tmp variable to store the probability of a city
```

```
        for j in range(Z):
            tmp *= solve[i][j]
        if tmp > maxProb:    ## compare tmp* with maxProb, and print the maximum
            maxProb = tmp       ## select that path which has the maximum value
            ans = i
    print("Probability:", solve[x][y])
    print("Path:", "->".join(list(path[x][y])))
    print("City:",maps[ans])
graph = []
direction = []
for i in range(Z):
    tmp = []
    tmp1 = []
    for j in range(Z):
        tmp.append(MIN_PROBABILITY)
        tmp1.append("")
    graph.append(tmp)
    direction.append(tmp1)


##explicitly describing the edges with values
##ed = [[0, 1], [0, 2], [0, 3], [1, 2], [1, 4], [1, 5], [2, 5], [3, 5], [3, 6], [4, 5], [4, 7], [5, 6], [5, 7],
[6, 7]]
##probs = [0.8, 0.7, 0.9, 0.8, 0.6, 0.6, 0.9, 0.6, 0.8, 0.8, 0.6, 0.7, 0.7, 0.9]
    direction[i[0]][i[1]] = maps[i[x]] + maps[i[y]]
    index += 1
project(graph,direction)
```