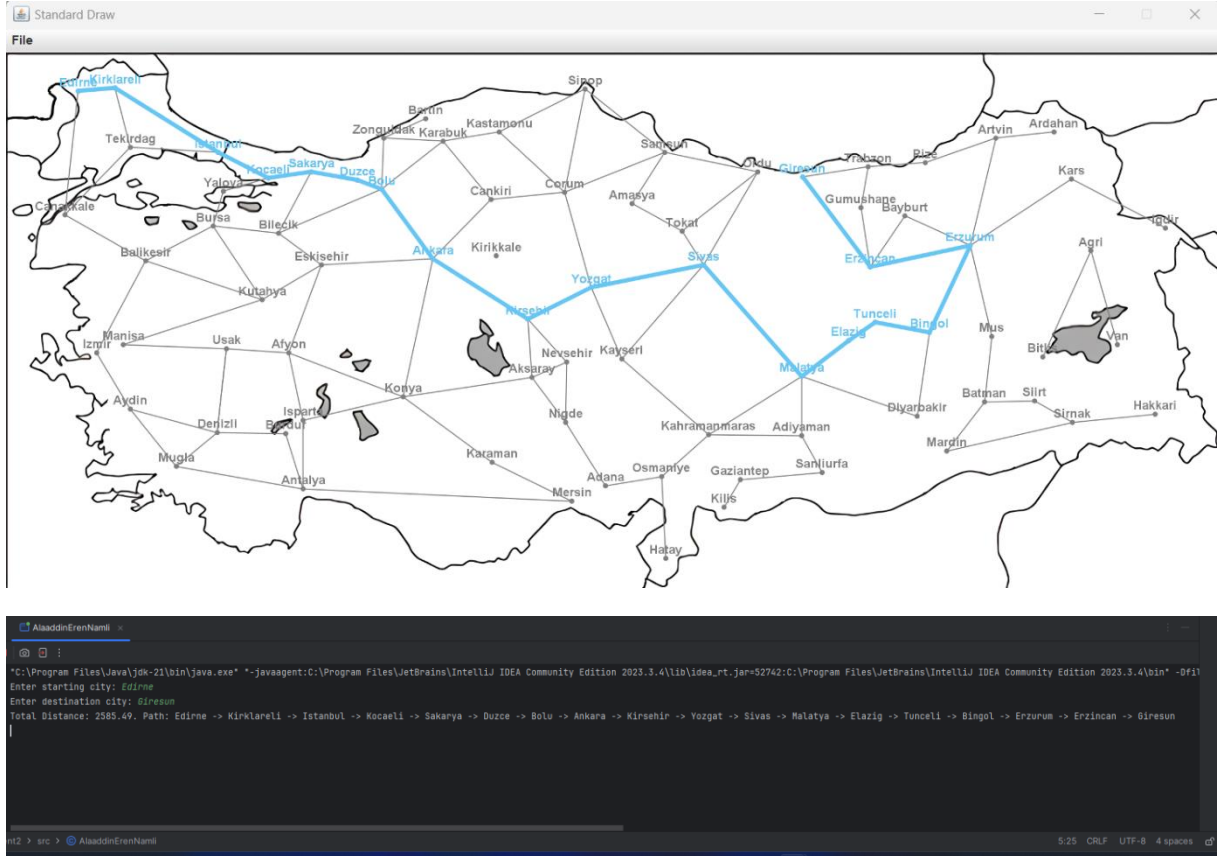


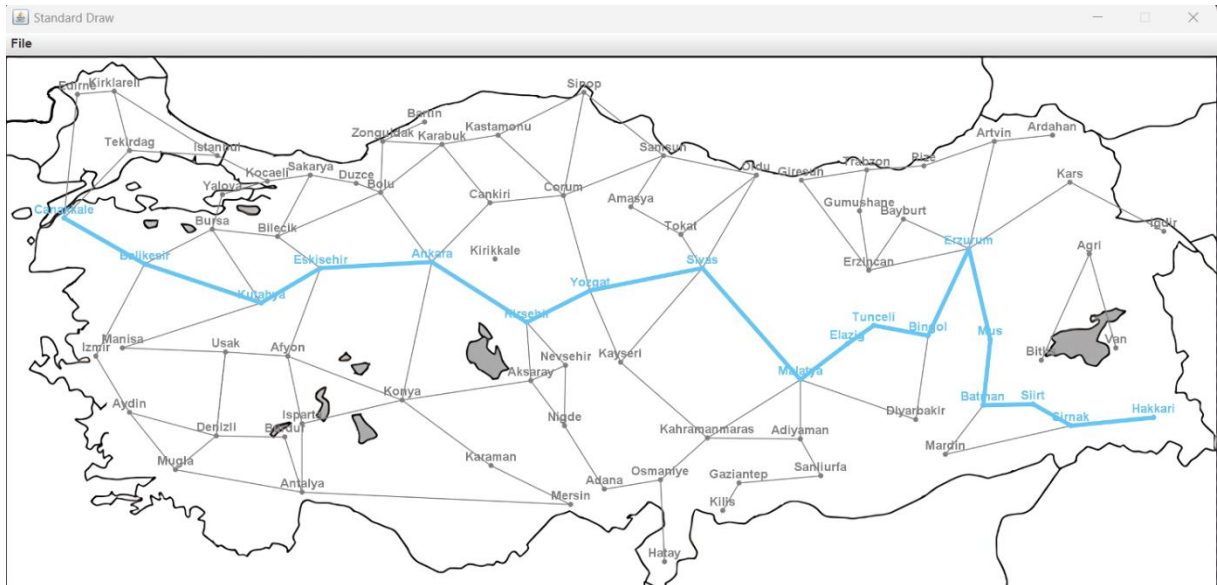
NAVIGATION APP REPORT

ALAADDIN EREN NAMLI

Edirne-Giresun

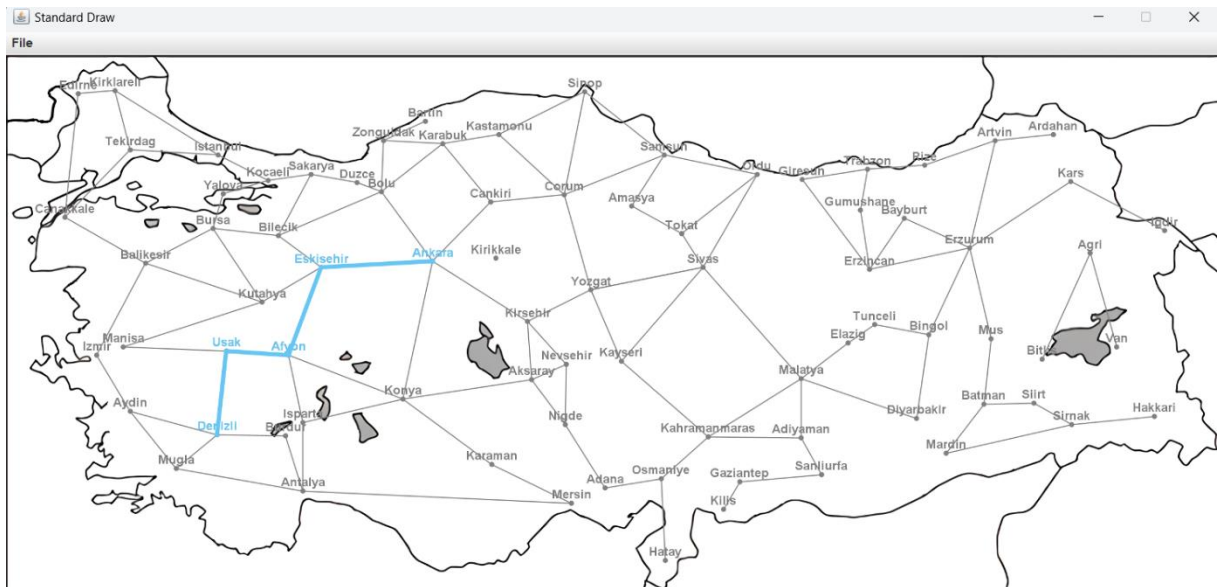


Çanakkale-Hakkari



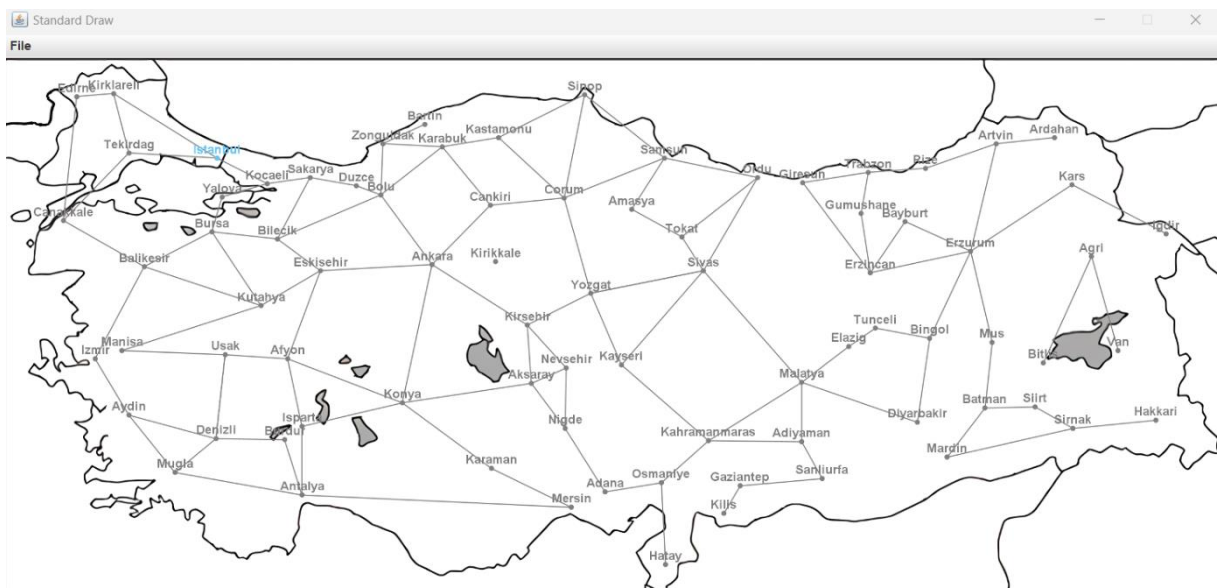
```
AlaaddinErenNami
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52800:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin" -Dfile.encoding=UTF-8
Enter starting city: Canakkale
Enter destination city: Hakkari
Total Distance: 2780.87. Path: Canakkale -> Balikesir -> Kutahya -> Eskişehir -> Ankara -> Kirsehir -> Yozgat -> Sivas -> Malatya -> Elazig -> Tunceli -> Bingol -> Erzurum -> Mus -> Batman -> Silirt -> Sirmak -> Hakkari
```

Ankara-Denizli



```
AlaaddinErenNami
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52785:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin" -Dfile.encoding=UTF-8
Enter starting city: Anka
City named 'Anka' not found. Please enter a valid city name.
Enter starting city: Ankara
Enter destination city: Deni
City named 'Deni' not found. Please enter a valid city name.
Enter destination city: Denizli
Total Distance: 889.10. Path: Ankara -> Eskişehir -> Afyon -> Uşak -> Denizli
```

Istanbul-Istanbul

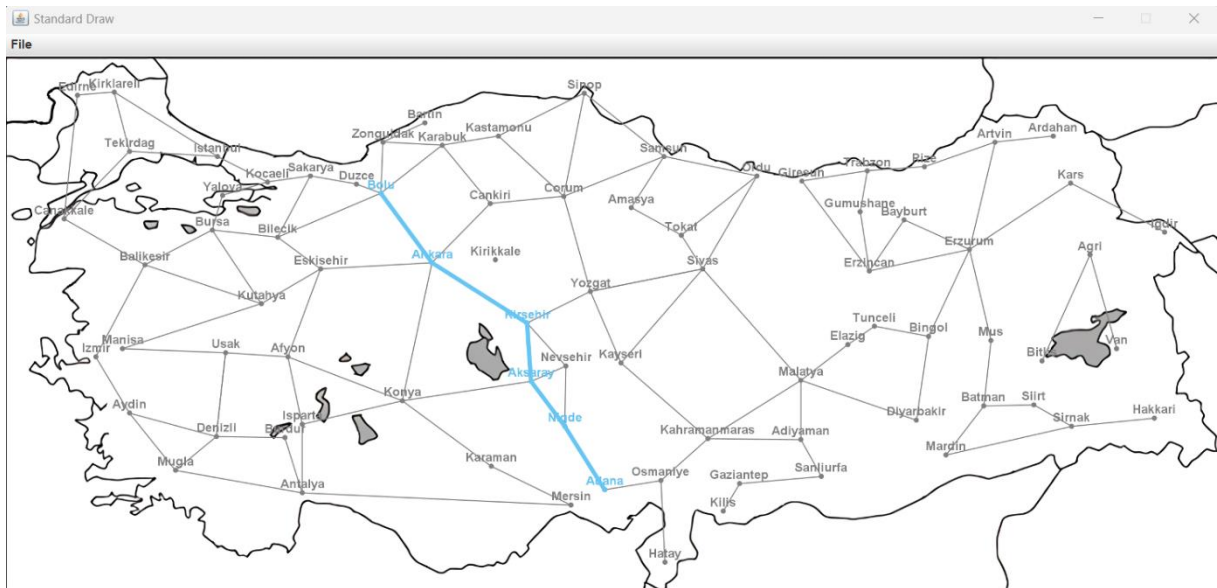


```
AladdinErenNaml  
C:\Program Files\Java\jdk-21\bin\java.exe *--javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52824:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin* -Df1  
Enter starting city: Istanbul  
Enter destination city: Istanbul  
Total Distance: 0.00. Path: Istanbul
```

Izmir-Van

```
AladdinErenNaml  
C:\Program Files\Java\jdk-21\bin\java.exe *--javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52829:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin* -Df1  
Enter starting city: Izmir  
Enter destination city: Van  
No path could be found.  
  
Process finished with exit code 0
```

Adana-Bolu

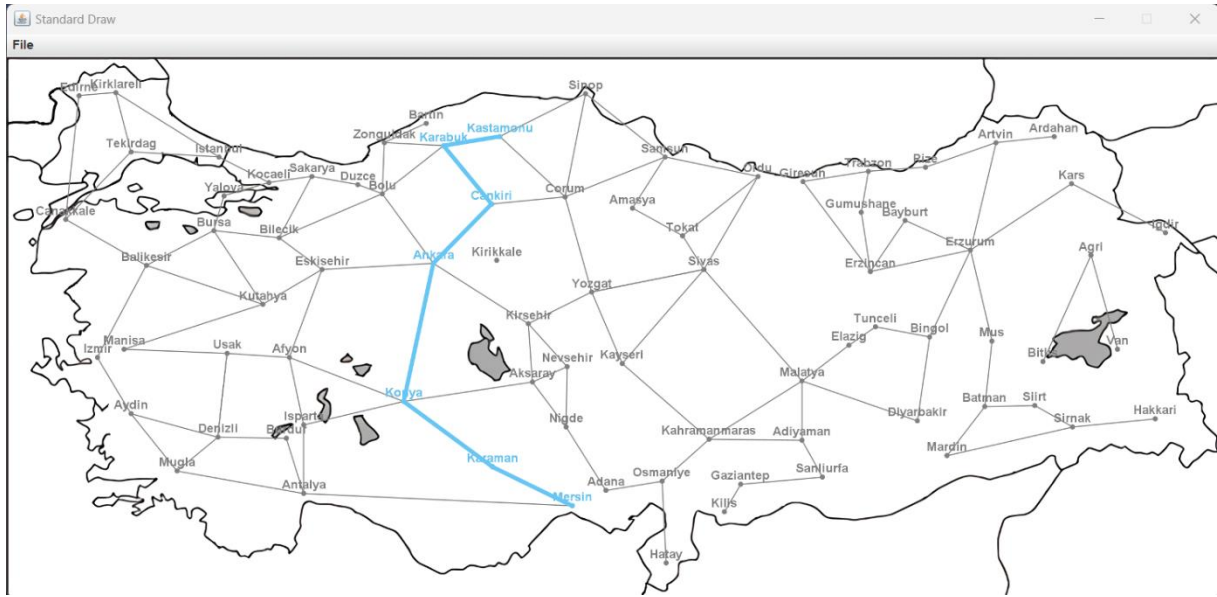


```
AladdinErenNaml  
C:\Program Files\Java\jdk-21\bin\java.exe *--javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52769:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin* -Df1  
Enter starting city: Adana  
Enter destination city: Bolu  
Total Distance: 759.85. Path: Adana -> Nigde -> Aksaray -> Kirsehir -> Ankara -> Bolu
```

Kastamonu-Kirikkale

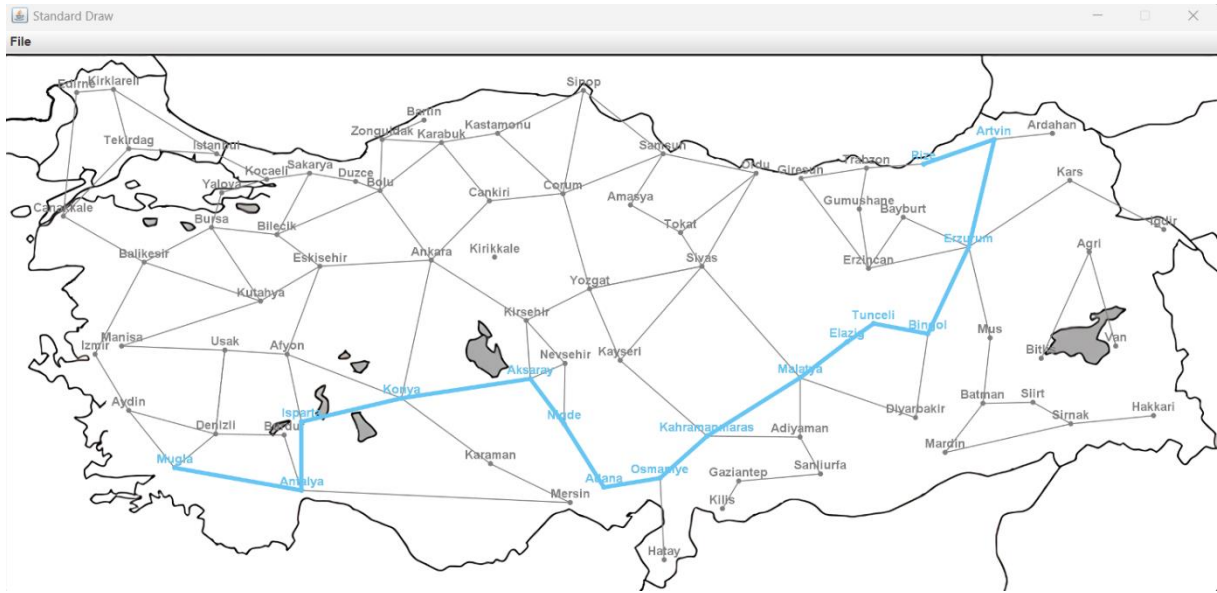
```
AladdinErenNaml  
C:\Program Files\Java\jdk-21\bin\java.exe *--javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52832:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin* -Df1  
Enter starting city: Kastamo  
City named "Kastamo" not found. Please enter a valid city name.  
Enter starting city: Kastamonu  
Enter destination city: Kirikkale  
No path could be found.  
  
Process finished with exit code 0
```

Kastamonu-Mersin



```
AlaaddinErenName
@ :
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52836:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin" -Dfile.encoding=UTF-8
Enter starting city: Kastamonu
Enter destination city: Mer
City named 'Mer' not found. Please enter a valid city name.
Enter destination city: Mersin
Total Distance: 1087.56. Path: Kastamonu -> Karabuk -> Cankiri -> Ankara -> Konya -> Karaman -> Mersin
```

Mugla-Rize



```
AlaaddinErenName
@ :
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\lib\idea_rt.jar=52848:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.3.4\bin" -Dfile.encoding=UTF-8
Enter starting city: Mugla
Enter destination city: Rize
Total Distance: 2384.58. Path: Mugla -> Antalya -> Isparta -> Konya -> Aksaray -> Nigde -> Adana -> Osmaniye -> Kahramanmaraş -> Malatya -> Elazığ -> Tunceli -> Bingöl -> Erzurum -> Rize
```

Path Finding Algorithm - Dijkstra's Algorithm Using Matrix

The idea is to generate a shortest path tree with a given starting city as a root. Maintain a matrix with two sets,

One set contains cities included in the shortest-path tree,

Other set includes cities not yet included in the shortest-path tree.

At every step of the algorithm, find a city that is in the other set (set not yet included) and has a minimum distance from the starting city.

Algorithm:

Create a set `isInSpt`(shortest path tree set) that keeps track of cities included in the shortest path tree, i.e., whose minimum distance from the starting city is calculated and finalized. Initially, this set is empty.

Assign a distance value to all cities in the input matrix. Initialize all distance values as INFINITE. Assign the distance value as 0 for the starting city so that it is picked first.

While `isInSpt` doesn't include all cities

Pick a city `u` that is not there in `isInSpt` and has a minimum distance value.

Include `u` to `isInSpt`.

Then update the distance value of all adjacent cities of `u`.

To update the distance values, iterate through all adjacent cities.

For every adjacent city `v`, if the sum of the distance value of `u` (from starting city) and weight of road `u-v`, is less than the distance value of `v`, then update the distance value of `v`.

Note: We use a boolean array `isInSpt []` to represent the set of cities included in SPT. If a value `isInSpt [v]` is true, then city `v` is included in SPT, otherwise not. Array `distance[]` is used to store the shortest distance values of all cities.

Note: In pseudocode `u` is represented by `minIndex` variable.

PSEUDOCODE

function dijkstraAlgorithm(matrix,source,target,numberOfCities,cityNames)

 Initialize distance[]

 Initialize isInSpt[]

 Initialize parent[]

 for each city C in matrix:

 distance[C] = INFINITY

 isInSpt[C] = FALSE

 parent[C] = 0

distance[source] = 0

parent[source] = -1

for numberOfCities - 1 times

 minIndex = call getMinDistance(distances, isInSpt,numberOfCities)

 isInSpt[minIndex] = true

 for c=0 to numberOfCities-1

 if (isInSpt[c] = false and matrix[minIndex][c] != 0 and distances[minIndex] !=
INFINITY and distances[minIndex] + matrix[minIndex][c] < distances[v])

 distances[v] = (distances[minIndex] + matrix[minIndex][c]);

 parent.set(v,minIndex);

printDistance(distances,target)

Initialize zeros[]

Initialize pathList[] = call getPath(parent,target,result,zeros)

return call printPath(pathList,target,distances,cityNames)

function getMinDistance(distances, isInSpt,numberOfCities)

 minimumValue = INFINITY

 minIndex = -1

 for c=0 to numberOfCities-1

 if (isInSpt[v] = false and distances[v] <= minimumValue)

 minimumValue = distances[v]

 minIndex = v

 return minIndex

function printDistance(distances,target)

 if (distances[target] != INFINITY)

 print distances[target]

function getPath(parent,target,result,zeros)

if (size of zeros = 2)

return []

if (parent[target] = 0)

add 1 to zeros

if (parent[target] = -1)

add -1 to result

return result

add parent[target] to result

return getPath(parent,parent[target],result,zeros)

function printPath(pathList,target,distances,cityNames)

Initialize citiesOnTheRoute[]

if (distances[target] = INFINITY or pathList = [])

print "No path could be found"

else

print "Path: "

for i=size of pathList-2 to 0

cityName = cityNames[pathList[i]]

add cityName to citiesOnTheRoute

print cityName + "->"

targetCityName = cityNames[target]

add targetCityName to citiesOnTheRoute

print targetCityName

return citiesOnTheRoute

REFERENCES

<https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/>

<https://www.geeksforgeeks.org/printing-paths-dijkstras-shortest-path-algorithm/>

https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm