Mike Dumas

May 1 2013

Project 4 Dijkstra Outline

1. Read input file
2. Parse Origin, Destination and triplength from input file
   1. Ex. Origin;Destination;Trip-length
   2. assume origin and destination are strings of max length 30 and are delmited by a semicolon not a space
   3. trip length is an integer representing the weight of edges
3. Use the graph class to add vertex and to add edges to build graph
   1. Determine the numb of vertices that have been input. Store it in an integer bucket
4. After inputting data, building graph, and storing veticies into some containor such as a vector or list implement dijstra's algorithm
   1. Sort Cities
   2. Print out the Graph
   3. User inputs the start vertex at the keyboard. If incorrect entry force retry
   4. use a data structure that can do the following:
      1. has strings that represent different verticies
      2. booleans to mark a vertex as to whether or not it was visited (meaning if it has been chosen as the smallest among the current unmarked)
      3. integers representing the total distance used in reaching this vertex. (total distance)
      4. strings representing the previous vertex that was visited before reaching a vertex
   5. Determine the index within local container of vertices where the starting vertex is located. Mark it as visited and make its distance 0. as previous it should have N/A
   6. Determine the vertices that ar adjacent to current vertex using get to vertices operation.
      1. use get to vertices and weight is within the dijstra's algorithm
   7. For each vertex adjacent to current vertex, do the following
      1. determine the weight value that exists between each vertex and current vertex
      2. if an adjacent vertex is currently unmarked and its distance value in the table is greater than the sum of the weights value plus the distance of your current vertex → reset its distance value to this smaller sum.
      3. Store the name of the current vertex as the previous vertex of each adjancent vertex
   8. find the minimum distance value among all unmarked vertices and set the vertex that corresponds to this minimum distance as your new current vertex. Mark this vertex
   9. Print out the data corresponding to your current marked vertex
   10. Repeat the previous four steps above until all of your vertices have been marked.
5. Extra credit, indicating whether or not a graph contains a cycle