Big O Analysis

Starting up the program leads to all of the teams being stored in to a list, this happens in O(n) time. As soon as the main window is loaded up all of the information is read in to a list that takes out all duplicate stadiums, this runs in O(n) time. After this all of the edges and vertices are input in to the graph, each of which is input at O(n) time.

for(all teams in database) n

for(all edges in database) n

for all(all vertices in the database) n

O(n) performance

When the Dijkstra's Algorithm is constructed, the program stores in the adjacency matrix in to the dijkstra class to navigate through and find the correct edges to travel on. This storage happens in O(n2) time due to the fact that the matrix is an n x n size. The program then calculates the distances for all of the other vertices in a different method. In this method a loop runs while the program index is under the number of vertices which runs in O(n). The loop also contains a for loop that runs through the number of vertices again, this is executed in O(n2). With the distances calculated in O(n2) time, the method is finished. The next method that outputs everything runs a for loop to go through each of the vertices and outputs their lengths. This happens in O(n). The whole dijkstra's algorithm is performed in O(n2) efficiency.

for(col) n

for(row) n2

while(count less than numofvertices) n

for(i to num of vertices) n2

O(n2)

The next algorithm is the algorithm for the MST. This algorithm first initializes all of the information in an array, which is performed in O(n) time. After this information is initialized a for loop is run for n times, which performs at O(n). In this loop the program updates the matrix and then enters another for loop that runs in n again, causing the program to run in O(n2). In this loop the program stores the parent for the vertices and then when it touches every element of the array the loop is finished. At this point the program outputs everything. The output prints in a for loop as well to get every node in the array. This performs at O(n) time. The MST algorithm runs at O(n2) time.

for(i to size) n

for(count to size) n

for(v to size) n2

for(i to size) n

O(n2)