1. Calculate the shortest paths between any two points in our graph, and create a new matrix.
    a. For the optimization purpose: If there is a direct path between two points, this is the shortest path; no need to calculate.
2. Cluster the TA homes, and for each cluster, choose one location as the center.
    a. We first set a limit (based on the average edge weight).
    b. For every column in the new matrix, we find out the smallest positive number and replace all the other numbers with -1. That is to say, for every location, we only keep its distance from its closest neighbor.
    c. Add a new column to the matrix. For every row in the new matrix, only consider the columns that represent TA homes. Count the number of positive value that is smaller than our limit, and replace numbers greater than the limit with -1. That is to say, we count the number of TA home whose distance from the current location is smaller than our limit. We store this count in the new column.
    d. Look at the newly added column of our matrix. Choose the largest number. The corresponding location is the center of this cluster and the locations corresponding to all positive numbers in this row are within this cluster.
    e. Repeat this process until there is no count greater than C (a parameter to be determined).
3. *Alternately, we can use the k-cluster algorithm to group the TA homes, where k equals 1/10 of the total number of locations. We plan to try out both algorithms and see which one achieves better performance.
4. Reduce the TA home problem to a metric TSP problem.
    a. We construct a new graph. First, add the centers of each cluster to the new graph and the edges between them. Then add all TA homes that are not in any clusters to the new graph with their incident edges. We need to solve a metric TSP problem on this new graph.
5. Use Christofides' Algorithm to solve the metric TSP problem.
    a. https://people.orie.cornell.edu/dpw/orie6300/Recitations/Rec12.pdf
    b. This is an approximation algorithm for the metric TSP problem. We decided to use it because it solves the metric TSP problem in polynomial time and outputs a solution that is at most 3/2*optimal solution.
6. Correction within clusters.
    a. Along the path returned by the TSP problem, for every pair of adjacent vertices (u, v), run the depth-first search algorithm from u in the original graph to find all the possible paths between them. For each path, count the number of vertices on the path that are within these two clusters. Choose the path with the largest number (break tie randomly).
7. Output the final path we choose.
    a. If the TA home is on our path, drop the TA at his/her home. Otherwise, drop the TA at the center of its cluster and let him/her walk home.