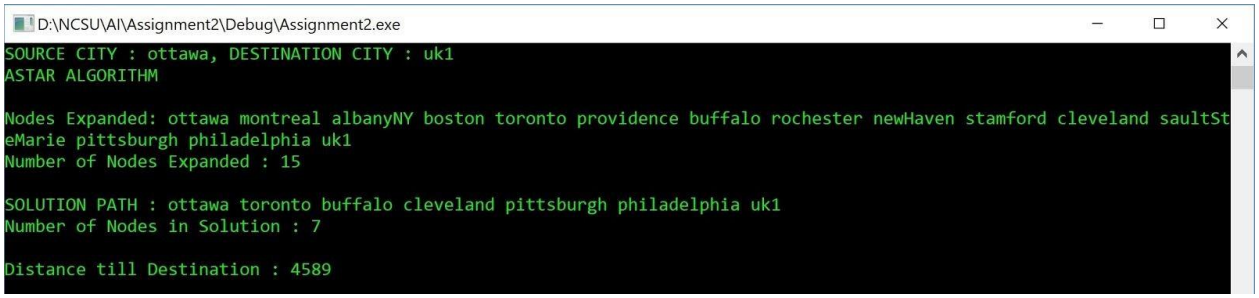
Q1.

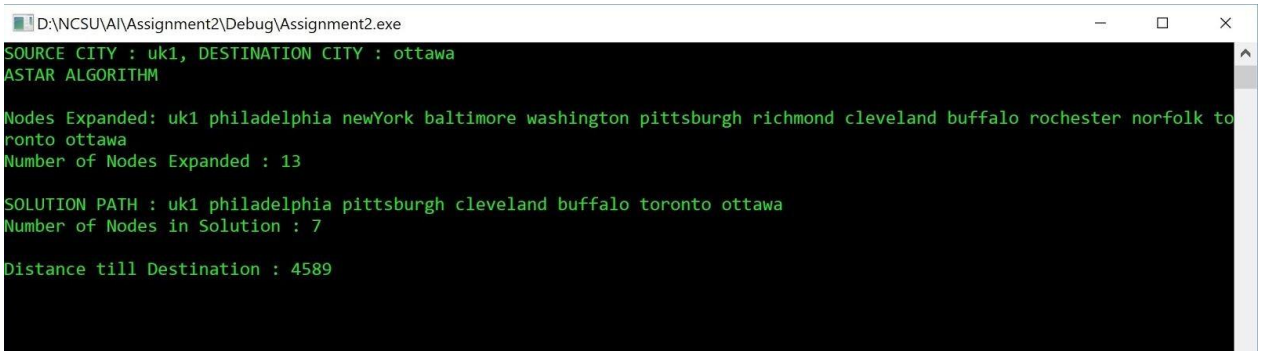
1. Experiment with executing your implementation of A* to find various paths, until you understand the meaning of the output. Are there any pairs of cities (A,B) for which the algorithm finds a different path from B to A than from A to B? Are there any pairs of cities (A,B) for which the algorithm expands a different total number of nodes from B to A than from A to B?

SOLUTION :

No, There aren't any two cities with a path different from A to B and from B to A. A* algorithm(with an underestimating heuristic) always returns the optimal path and there can be only one optimal path in a graph containing only bi-directional edges(undirected graph).

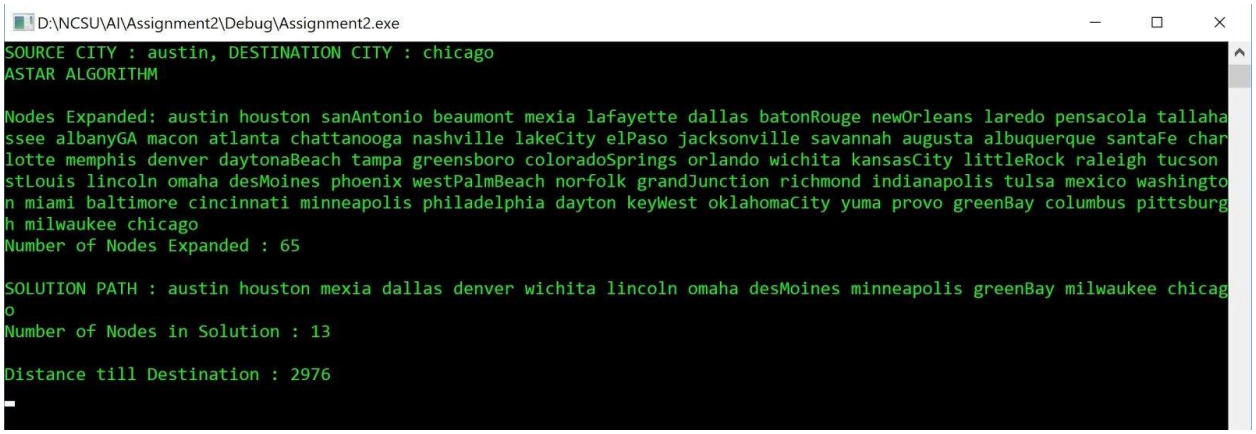
Here is an example of a pair of city (ottawa, uk1) for which the algorithm expands a different number of nodes from ottawa to uk1 and uk1 to ottawa.





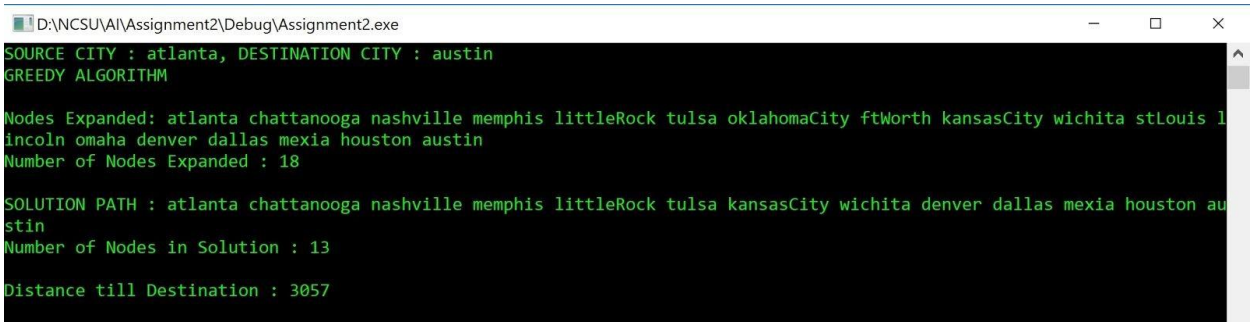
We can see in the given output, that the number of nodes expanded from ottawa to uk1 is 15 and number of nodes expanded from uk1 to ottawa is 13.

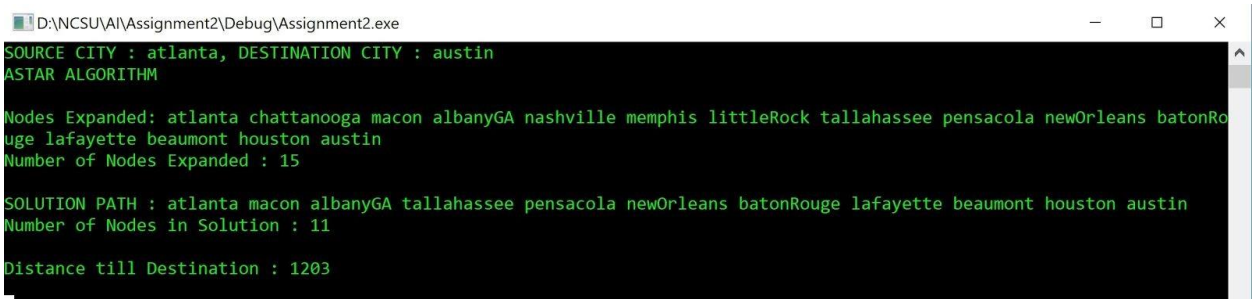
2. Implemented the code for greedy algorithm.
3. Do enough exploration to find at least one path that is longer using greedy search than that found using A*, or to satisfy yourself that there are no such paths. Find at least one path that is found by expanding more nodes than the comparable path using A*, or satisfy yourself that there are no such paths. If there is such a path, list the nodes in the path and the total distance.
SOLUTION:
One path that is longer using greedy search than that found using A* is from AUSTIN TO CHICAGO.



We see that A* algorithm gives a path with 13 nodes while Greedy algorithm gives a path with 25 nodes. Also, the path cost for A* is much less than the cost for greedy algorithm.
One path that is found expanding more nodes in Greedy Algorithm than the comparable path using A* is from ATLANTA to AUSTIN

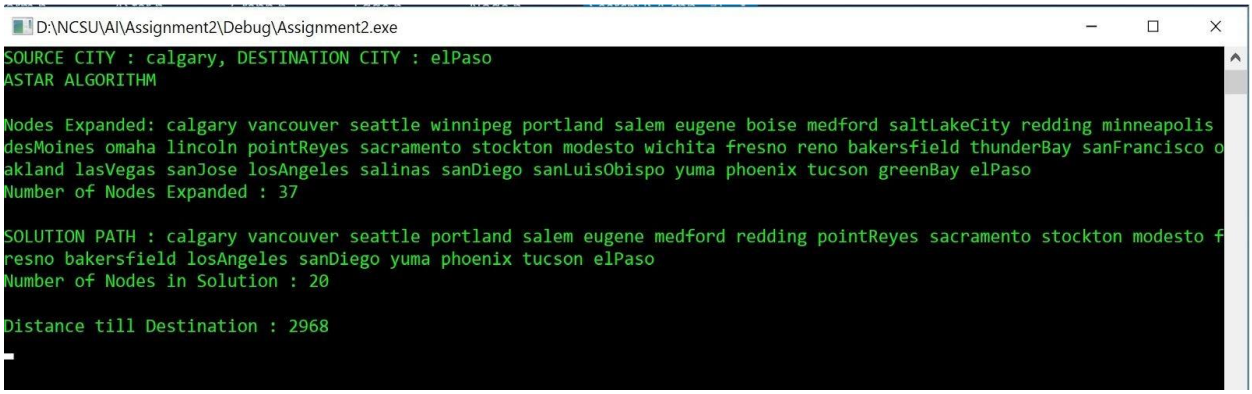
We see that the number of nodes expanded in A* algorithm is 15 while greedy algorithm expands 18 nodes.

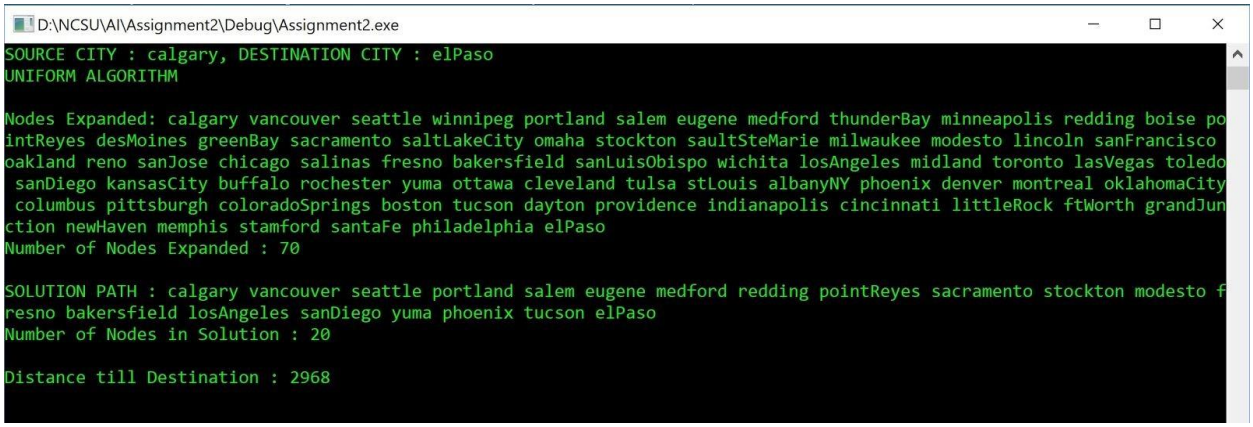




4. [10 points] Change your code so as to implement uniform cost search, as discussed in the web notes.
Implemented the Uniform Cost Search Algorithm.

5. Do enough exploration to find at least one path that is longer using uniform cost than that found using A*, or to satisfy yourself that there are no such paths. Find at least one path that is found by expanding more nodes than the comparable path using A*, or satisfy yourself that there are no such paths. If there is such a path, list the nodes in the path and the total distance.
/*
One path that is longer using uniform cost than that found using A* : There is **no path** for which A* algorithm gives a shorter path than uniform cost algorithm. We know Uniform Cost always gives the least cost path and since, there was no path with a different size than Uniform Cost when run on A* implies that the heuristic used for A* is underestimating and hence, A* also gives the optimal path. This is because both the algorithm guarantee shortest path(given heuristic is underestimating).
***/**
One path that is found by expanding more nodes in Uniform Cost Search than the comparable path using A* is from





We see that the number of nodes expanded in A* algorithm is 37 while Uniform Cost search expands 70 nodes.

6. As part of your answer, compare the solution paths and explain what happened, especially any weird behavior you might detect.

The A* algorithm always gives the optimal solution if the heuristic is underestimating. Hence, it does not matter for a path from A to B and from B to A, as the path remains the same which is optimal i.e. with the least path cost. Initially, we dont know whether the heuristic is underestimating. But when we find all the paths found by Astar equal to the paths found by uniform cost, it was confirmed that the heuristic is underestimating.By comparing Greedy Search algorithm with A* algorithm, we see that Greedy Search algorithm does not guarantee the optimal path. Although, it does expands lesser number of nodes to reach the target city, it would not mind going on a path more costly. Greedy algorithm is thus much faster than A* and also has less memory usage in terms of expanded nodes storage but will not return an optimal path.

By comparing Greedy Search algorithm with A* algorithm, we don’t find any two cities with different solutions confirming the heuristic being underestimating. I literally ran the code for all the possible city combinations and failed to find such a case. Uniform Cost does use a lot more memory in terms of nodes expanded than A* algorithm in most cases.