Assignment 7 - Graphs

3) Comparison

Both A* and Dijkstra will eventually reach H through exploration but in this case, Dijkstra found the shortest path to H in fewer iterations because A* explored B, which had a lower cost, before H. For Dijkstra, it went through 4 iterations to get to H, since it had a lower cost in that algorithm, while A* took 5 iterations to get to H.

7) Algorithm Analysis

The time complexity of DirectedCheck is $O(n^2)$ because it compares every pair of entries in the $n \times n$ adjacency matrix to check if the graph is symmetric. It loops through all rows and columns, checking matrix[i][j] against matrix[j][i] for all i and j. The space complexity is $O(1)$ since it does not use any extra data structures. The space being taken up is the input matrix and variables.

The time complexity of CircularGraphMatrix is $O(n)$ because it loops through each of the n vertices once to get and assign two edges per vertex in the adjacency matrix. The operations inside the loop takes constant time. The space complexity is $O(n^2)$ because it uses a full $n \times n$ matrix for the graph. Most of the entries are zero and only two per row are filled.

The time complexity of SimplePaths is $O(V)$ in the worst case because it uses a recursive backtrack method that explores all simple paths from the start node to the end node. It is checking each one against the target weight. This could lead to checking all permutations of the vertices if the graph is fully connected. The space complexity is $O(V)$ because of the recursive call stack, path, and visited sets, which get larger with the depth of the search, or the number of vertices.