Question #7:
For #4:
Time complexity: O(n^2) where n is the number of vertices because due to the double for loop, when reading the input it runs n * n times. Also, when I print the matrix in the double for loop, I am printing n^2 elements.
Space complexity: O(n^2) where n is the number of vertices because when i initialized matrix, it allocated space for n * n  integers.
I am processing and storing all n^2 iterations of the matrix exactly once.

For #5:
Time complexity: O(V^7) because it tries all possible paths from the start to the end, without repeating nodes. It uses depth-first search (DFS) to find all the different paths that add up to a total weight of 7.
Space complexity: O(V+E) where V is the number of nodes to keep track of visited nodes and paths and E is the number of edges to store the graph in memory.

For #6:
Time complexity: O(n) where n is the number of vertex-distance pairs in the input. The program runs in linear time with respect to the number of nodes, because every major operation happens once per node.
Space complexity: O(n) where n is the number of vertices because it stores n nodes and graphs with n nodes and 2n edges.