1. What is the big-Oh space complexity of an adjacency list? Justify your answer.

   The space complexity of an adjacency list is O(N+E). Because we will have a doubly linked list for N nodes and each node we will store the edges connected with it.

2. What is the big-Oh space complexity of an adjacency matrix? Justify your answer.

   The space complexity for adjacency matrix is $O(N^2)$ because we will need to have n rows and n columns in the adjacency matrix. The matrix size is depending on the number of nodes. We will need to know the relationship(with direction) between each node.

3. What is the big-Oh time complexity for searching an entire graph using *depth-first search* (DFS)? Does the representation of the graph make a difference? Justify your answer.

   DFS will need to go through each node and edges. The representation of graph does make a difference. In the adjacency matrix, we will need to go through the entire matrix with N rows and N columns. Therefore, the time complexity is the time to visit N*N cells, which is $O(N^2)$. In the adjacency list, because the graph is stored in doubly linked list and it would not store the data if two nodes are not connected. Using DFS, we will visit the node's edges to see it has next level , so we will visit the edges associated with the nodes. If there is no next level, we will visit other nodes. Therefore, the time complexity will be the time to visit all nodes in DLL and visit the edges in DLL of node. The big O time complexity is O(V+E).

4. What is the big-Oh time complexity for searching an entire graph using *breadth-first search* (BFS)? Does the representation of the graph make a difference? Justify your answer.

   BFS will have the same time complexity for searching in graph. We also need to visit every node and edge in the graph but the order is different from DFS. The time complexity of BFS is $O(N^2)$ in adjacency matrix because every we need to find the edges for node, we will need to visit the whole rows, and we need to repeat N times to visit everything. The time will be N*N. The time complexity is O(N+E) in adjacency list. We could visit all edges for one node and then visit the other nodes.