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

   Assuming the graph has n vertices and m edges, the space complexity of an adjacency list is $O(n + m)$. We need n linked lists to store nodes and their adjacency edges. For all nodes, the total number of adjacency edges is m, so we add the number of linked lists and the the total number of edges to compute the space complexity, which is $O(n + m)$. The worst case is $O(n^2)$ when every node connects to all the other nodes.

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

   Assuming the graph has n vertices, the space complexity of an adjacency matrix is $O(n^2)$. Because we need a 2d array to store whether nodes connect each other, and the rows and columns of this 2d array are n, so the space complexity is apparently $O(n^2)$.

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.**

   Assuming the graph has n vertices and m edges.

   If the graph is represented as adjacency list, the time complexity of searching an entire graph using DFS is $O(n + m)$. For each node, we discover all its neighbors by traversing its adjacency list in linear time. So we add the number of linked lists and the the total number of edges to compute the time complexity, which is $O(n + m)$.

   If the graph is represented as an adjacency matrix, the time complexity of searching an entire graph using DFS is $O(n^2)$. For each node, we discover all its outgoing edges by traversing an entire row of length n in the matrix. Each row stores information of the leading node's edges, and the number of edges is n. Therefore, the time complexity of DFS in this case is $O(n^2)$.

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.**

   Assuming the graph has n vertices and m edges.

   If the graph is represented as adjacency list, the time complexity of searching an entire graph using BFS is $O(n + m)$.

   If the graph is represented as adjacency matrix, the time complexity of searching an entire graph using BFS is $O(n^2)$.

   Since we have to explore every vertex and every edge in our graph whether using BFS or DFS, the time complexity of both BFS or DFS are the same.