1. Data Structures and Strategy

The main difference between these two strategies is in the data structures used to manage the "frontier". Breadth-First Search (BFS) utilizes a Queue (first in, first out) while Depth-First Search (DFS) utilizes a Stack (last in, first out), which will go deep into a single branch before backtracking.

2. Path Optimality and Length

As shown on the results for questions C and D, BFS identified the shortest/optimal path, with 7 nodes. On the other hand, DFS found a longer path with 11 nodes. This is because DFS will first explore deep into the first available branch before backtracking as compared to BFS that will go “layer by layer” trying to find the optimal path.

3. Efficiency and Memory Usage

As shown on the results for questions A and B, DFS visited fewer nodes overall than BFS. This shows that if a goal is located deep within the first branch searched DFS can reach it with very little exploration. On the other hand, BFS explored one more node and typically requires more memory as it explores an entire "frontier" of nodes in the queue. DFS only keeps the current path on its stack, making it less memory intensive for deep searches.

4. Uninformed Search and Selection

Although BFS and DFS are of type uninformed (blind) search strategies, and it would not be realistic to have some clue of where the data could be stored, I believe that depending on how close the data could be we could choose one over the other. BFS can be used for when the goal is suspected to be close to the start point, and DFS can be used there is a memory limit or when the goal is suspected to be deep into the roots of the tree.