What is the central difference between depth- and breadth-first searches as you have noticed from your output?

The depth first search output requires more steps than the breadth first search, meaning that the breadth first search algorithm searches through the maze in the shortest pass possible. In the output, the breadth first search finds the cheese in fewer steps than the depth first search.

Under what circumstances would one be better than the other?

If the purpose of an algorithm is to find the shortest path, then a breadth first search would be more efficient. If an algorithm requires a wide search where the target node is located quite deep, a depth first search algorithm would be more efficient.

Compare the complexity of both search strategies?

Both algorithms have the same time complexity of O(n) . The complexity for both searches depend on the size of the input n. The steps in the while loop are O(1) but because the entire algorithm depends on n, the complexity is denoted as O(n).

Part I – Depth-First Search
Loop Invariant: The values in the maze, for each iteration of the while loop are either already stored on the stack or visited.

Part II – Breadth-First Search
Loop Invariant: The values in the maze, for each iteration of the while loop are either already stored on the queue or visited.