### Algorithm: Depth-First Search (DFS)

Once upon a time, in the kingdom of Algoria, Sir Cedric, a noble knight, embarked on a quest to rescue Princess Elara from the Tower of Mazes. His trusted dragon, Ember, flew them swiftly over the treacherous landscape. The Tower of Mazes, a formidable labyrinth, required a clever strategy to navigate.

#### Initialize Data Structures:

* Sir Cedric used a magical map (stack) to keep track of the paths he needed to explore.
* He carried a magical lantern (set) to remember the paths he had already traveled.

#### Traverse the Labyrinth:

* Sir Cedric started at the entrance, marking it with the lantern's light.
* He delved into the labyrinth, moving from one chamber (node) to the next, always choosing the deepest path first.

#### Repeat:

* Whenever he reached a dead end, he retraced his steps to the last junction and chose a new path.

#### Implementation:

| **def** dfs(maze: Dict[int, List[int]], start: int) -> List[int]:  **def** explore(chamber):  lantern.add(chamber)  **for** passage **in** maze[chamber]:  **if** passage **not** **in** lantern:  explore(passage)  path.append(chamber)  lantern = set()  path = []  **for** chamber **in** maze:  **if** chamber **not** **in** lantern:  explore(chamber)  **return** path[::-1]  *# Example usage:*  maze = {  1: [2, 3],  2: [4, 5],  3: [6, 7],  4: [],  5: [],  6: [],  7: []  }  start = 1  print(dfs(maze, start)) *# Output: Path through the labyrinth* |
| --- |

#### Explanation:

Initialize:

* lantern: A magical set that marks the paths explored.
* path: The sequence of chambers explored, marking Sir Cedric's journey.

Traverse the Labyrinth:

* Sir Cedric delved into the maze, marking chambers and exploring passages.

Repeat:

### He backtracked to the last junction whenever he reached a dead end, ensuring he explored every path.