### Algorithm: Breadth-First Search (BFS)

In the mystical lands of Algoria, Sir Cedric and Ember faced the Enchanted Forest, where Princess Elara was held captive by the Dark Sorcerer. The forest was dense with enchantments, requiring Sir Cedric to explore it methodically.

#### Initialize Data Structures:

* Sir Cedric used a magical compass (queue) to keep track of the clearings (nodes) he needed to visit.
* He left behind magical runes (set) at each clearing to mark his path.

#### Traverse the Enchanted Forest:

* Sir Cedric entered the forest, marking the entrance clearing with a rune.
* He explored each clearing level by level, ensuring no path was left unseen.

#### Repeat:

* He continued exploring until he found Princess Elara or had visited all clearings.

#### Implementation:

| **from** **collections** **import** deque  **def** bfs(forest: Dict[int, List[int]], start: int) -> List[int]:  runes = set()  compass = deque([start])  path = []  **while** compass:  clearing = compass.popleft()  **if** clearing **not** **in** runes:  runes.add(clearing)  path.append(clearing)  **for** trail **in** forest[clearing]:  **if** trail **not** **in** runes:  compass.append(trail)  **return** path  *# Example usage:*  forest = {  1: [2, 3],  2: [4, 5],  3: [6, 7],  4: [],  5: [],  6: [],  7: []  }  start = 1  print(bfs(forest, start)) *# Output: Path through the Enchanted Forest* |
| --- |

#### Explanation:

Initialize:

* runes: Magical markers to track explored clearings.
* compass: A queue to explore the forest level by level.

Traverse the Enchanted Forest:

* Sir Cedric explored each clearing, ensuring no trail was left unseen.

Repeat:

* He continued exploring until Princess Elara was found or all clearings were visited.