Assignment 3
Tower of Hanoi using Breadth First Search Algorithm

The solution is structured around several key functions. The **init_state** function generates the initial configuration of the rods and disks, while **get_goal_state** creates the target configuration. Both functions use tuples of tuples to represent the rods, ensuring that each state is immutable and can be efficiently tracked in a set for visited states. The valid_move function enforces the Tower of Hanoi rules by checking if a move is legal before it is considered.

The main logic is handled by the **bfs_hanoi** function, which uses a queue to perform a *Breadth-First Search (BFS)* through all possible states. This function keeps track of explored states to avoid excessive repetitions and records each state transition and move in a structured table. The output is formatted as a CSV file, clearly showing each step, the rods' contents, and the move taken. Additional variables such as **step_count**, **found**, and **solution_path_length** help monitor the search process and ensure that the solution is reported accurately.

Key variables include:

num_discs, **init_rod**, **init_order**, **final_rod**, **final_order**: User inputs defining the problem setup.
output_file: The name of the file where results are saved.
max_steps: An optional limit on the number of states to explore.
queue: Stores states and their corresponding move paths for BFS.
visited: Tracks already explored states.
output_rows: Collects the table of explored states and moves.
step_count, found, **solution_path_length**, **state**, **path**, **move_desc**: Used to manage and describe the search process and results.

By iteratively refining the logic and output formatting, I ensured that the program not only finds the correct solution but also presents it in a clear, step-by-step manner, matching the requirements.

*Note: The **max_steps** input is a practical safeguard and a user-friendly feature that gives the control over the breadth and duration of the search, making the program more robust and flexible.*