8 PUZZLE ASSIGNMENT

- Includes standard libraries for input/output, memory management, and boolean data types.
- Defines constants for the board size (3x3) and maximum possible states (9!).
- `State` structure represents a puzzle state, including the board configuration, blank tile position, and cost to reach the state.
- `PQNode` structure represents a node in the priority queue, containing a state and its priority.
- `PriorityQueue` structure represents the priority queue, containing an array of `PQNode` pointers and the queue size.
- `initializeState` function creates an initial state from a given 2D array and identifies the blank tile position.
- `isGoalState` function checks if a given state matches the goal configuration.
- `calculateHammingPriority` function counts the number of misplaced tiles.
- `calculateManhattanDistance` function calculates the Manhattan distance for a given tile's position versus its goal position.
- `calculateManhattanPriority` function sums the Manhattan distances for all tiles in the state.
- `printBoard` function prints the current state of the board in a 3x3 grid format.
- `swap` function exchanges the values of two given integers.
- `createPriorityQueue` function initializes an empty priority queue.
- `push` function adds a state to the priority queue with a specified priority.

- `pop` function removes and returns the state with the highest priority (lowest cost) from the priority queue.
- `isEmpty` function checks if the priority queue is empty.
- `aStarSearch` function implements the A* search algorithm, initializing the priority queue with the initial state.
- The `aStarSearch` function expands states, generates successor states, calculates priorities using Manhattan distance, and pushes successors onto the queue.
- Memory for states is dynamically allocated using `malloc` and deallocated using `free`.
- The `main` function initializes the initial state, checks if it is the goal, prints relevant information, and calls `aStarSearch` to find the solution.
- The initial configuration of the board is defined as a 2D array in the `main` function.
- The `main` function prints the initial state configuration using `printBoard`.
- The `main` function calculates and prints the Hamming priority of the initial state.
- The `main` function calculates and prints the Manhattan priority of the initial state.
- The `swap` function is used to generate successor states by moving the blank tile.
- Successor states are generated by moving the blank tile up, down, left, or right within the grid bounds.
- The cost is incremented by 1 for each move made from the current state to a successor state.
- Invalid moves that result in the blank tile moving out of the grid bounds are skipped.