Approach

- A* Search with Manhattan distance heuristic function.
- State of the board is stored using a string of length 16. This is just a flattened version of the provided 2D list. Since string is immutable, we can use this as a key in python dictionary.
- Used a min-heap with each element of the format `[[f(n),g(n)],state]` where `f(n)=g(n)+h(n)`, `g(n)` is the path cost till state `n` and `h(n)` is heuristic cost of state `n`.
- `minimum_cost_list` dictionary contains the minimum value of `f(n)` for all the states that have been added in the min-heap.
- `last_move` dictionary stores the last move performed which lead to this state.
- `direction[]` list is used to generate new nodes.
- In while loop, the state is skipped if the minimum cost stored in the `minimum_cost_list` is lesser than `current_total_cost` popped from min-heap.
- When the goal state is reached, `last_move` dictionary is used to construct the min path.

Why is this correct?

- A* graph search is a complete and optimal informed search algorithm provided that the heuristic is admissible and consistent.
- Here Manhattan distance heuristic function is used.
- Manhattan distance is both admissible and consistent heuristic as it never overestimates the cost of reaching the goal.
- It is consistent as it satisfies the triangle inequality: $h(n) \leq c(n,a,n) + h(n)$
- As h(n) is consistent, then the values of f(n) along any path are nondecreasing.
- In the relaxed problem, since the tile is moved only 1 step at a time and only in one of 4 directions, the best case scenario for each tile is that it has a clear, unobstructed path to its goal state.
- The actual number of steps required will definitely be greater than or equal to the steps required in an unobstructed path.

Comparison of Various Methods Used

- All methods were run 5 times using an a shell script. Average and minimum values are recorded.
- Many admissible heuristics were tested including Manhattan distance, Euclidean distance, misplaced tiles, misplaced rows and columns.
- Only Manhattan distance gave good results. Other heuristics were way too slow to be measured.
- Combination of any heuristic with manhattan distance resulted in large increase in time.
- Table contains top 5 methods based on average and minimum time taken in seconds.
- By “dictionary” in table, I mean the data structure used to store information about a state. See comments in code.
- The last method is submitted as final answer.

Method	Type	initial state			
		1	2	3	4
combined dictionary except zero position, direction stored as integer	Average	0.0007	6.60868	41.6966	67.95684
	Minimum	0.0007	6.0339	37.1608	59.2543
combined dictionary, direction stored as integer	Average	0.0007	6.62482	40.59418	64.82714
	Minimum	0.0007	6.0763	37.1743	58.6978
separate dictionaries, direction stored as integer	Average	0.0007	6.33608	42.48142	66.90344
	Minimum	0.0007	5.7972	42.1009	64.7941
separate dictionaries, direction stored as integer, combined heuristic (Manhattan distance, misplaced tiles)	Average	0.0009	9.2457	52.02322	81.41252
	Minimum	0.0009	7.6266	46.1467	72.4002
separate dictionaries, direction stored as string	Average	0.0007	7.00416	39.79692	62.5412
	Minimum	0.0007	6.6213	35.5181	55.9403