```
(1) A* Algorithm – 8-puzzle game
```

Input-

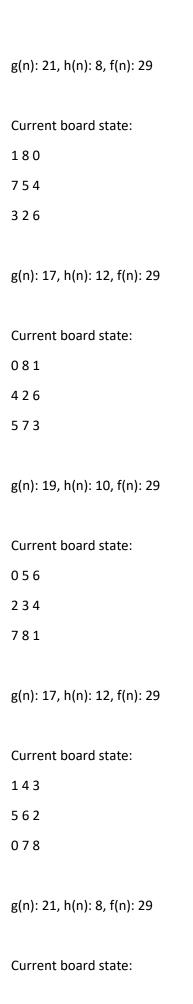
```
import heapq
```

```
class PuzzleState:
  def __init__(self, board, g, h):
    self.board = board # The current state of the board
    self.g = g # Cost to reach this node (depth)
    self.h = h # Heuristic cost (Manhattan distance)
    self.f = g + h \# Total cost (f(n) = g(n) + h(n))
  def __lt__(self, other):
    return self.f < other.f # For priority queue to sort by f(n)
def print board(board):
  """Print the current board state."""
  for row in board:
    print(" ".join(str(num) for num in row))
  print() # Empty line for better readability
def get_blank_position(board):
  for i in range(3):
    for j in range(3):
       if board[i][j] == 0: # Find the blank space (0)
         return (i, j)
def get_successors(state):
  successors = []
  x, y = get_blank_position(state.board) # Get position of blank tile
  directions = [(-1, 0), (1, 0), (0, -1), (0, 1)] # Possible moves
  for dx, dy in directions:
```

```
new_x, new_y = x + dx, y + dy
    if 0 \le \text{new}_x \le 3 and 0 \le \text{new}_y \le 3: # Valid move
      new_board = [row[:] for row in state.board] # Copy the current board
      new_board[x][y], new_board[new_x][new_y] = new_board[new_x][new_y], new_board[x][y]
# Swap
      successors.append(PuzzleState(new_board, state.g + 1, 0)) # Create new state
  return successors
def heuristic_manhattan_distance(board):
  distance = 0
  for i in range(3):
    for j in range(3):
      if board[i][j] != 0:
        target_x = (board[i][j] - 1) // 3
        target_y = (board[i][j] - 1) % 3
         distance += abs(i - target_x) + abs(j - target_y)
  return distance
def is_goal_state(board):
  return board == [[1, 2, 3],
           [8, 0, 4],
           [7, 6, 5]] # Check if the board is in the goal state
def a_star_search_manhattan_distance(start_board):
  start_state = PuzzleState(start_board, 0, heuristic_manhattan_distance(start_board))
  open_set = []
  heapq.heappush(open_set, start_state)
  closed_set = set()
  while open_set:
    current_state = heapq.heappop(open_set)
```

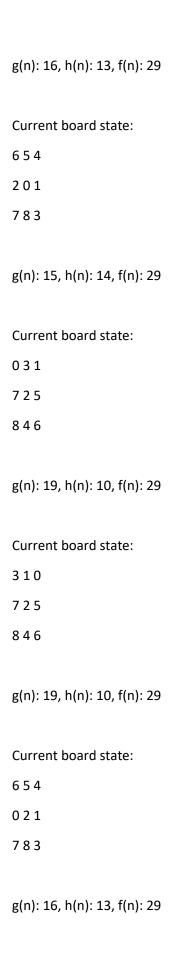
```
# Print current board state and details
    print("Current board state:")
    print_board(current_state.board)
    print(f"g(n): {current_state.g}, h(n): {current_state.h}, f(n): {current_state.f}\n")
    # Check if we've reached the goal
    if is_goal_state(current_state.board):
      print("Goal state reached!")
      return current_state.g # Return the cost to reach the goal
    closed_set.add(tuple(map(tuple, current_state.board)))
    for successor in get_successors(current_state):
      successor.h = heuristic_manhattan_distance(successor.board)
      successor.f = successor.g + successor.h
      if tuple(map(tuple, successor.board)) in closed_set:
        continue
      heapq.heappush(open_set, successor)
  return None # No solution found
def get_user_input():
  board = []
  for i in range(3):
    while True:
      row = input(f"Enter row {i + 1} (3 numbers separated by space): ")
      nums = list(map(int, row.split()))
      if len(nums) == 3 and all(0 <= num <= 8 for num in nums):
```

```
board.append(nums)
        break
      else:
        print("Invalid input. Please enter 3 numbers between 0 and 8.")
  return board
if __name__ == "__main__":
  start_board = get_user_input()
  steps = a_star_search_manhattan_distance(start_board)
  print(f"Steps to solve with Manhattan Distance heuristic: {steps}")
output-
143
580
726
g(n): 22, h(n): 7, f(n): 29
Current board state:
436
175
028
g(n): 19, h(n): 10, f(n): 29
Current board state:
720
413
856
```



| 8 2 1 |
|------------------------------|
| 476 |
| 053 |
| |
| g(n): 19, h(n): 10, f(n): 29 |
| Current board state: |
| 321 |
| 075 |
| 8 4 6 |
| |
| g(n): 18, h(n): 11, f(n): 29 |
| Current board state: |
| 104 |
| 785 |
| 3 2 6 |
| |
| g(n): 16, h(n): 13, f(n): 29 |
| |
| Current board state: |
| 301 |
| 7 2 5 |
| 8 4 6 |
| |
| g(n): 18, h(n): 11, f(n): 29 |
| |
| Current board state: |
| 516 |

3 4 0



| 6 5 4 |
|------------------------------|
| 210 |
| 783 |
| |
| g(n): 16, h(n): 13, f(n): 29 |
| |
| Current board state: |
| 163 |
| 705 |
| 284 |
| |
| g(n): 19, h(n): 10, f(n): 29 |
| |
| Current board state: |
| 054 |
| 621 |
| 783 |
| |
| g(n): 17, h(n): 12, f(n): 29 |
| |
| Current board state: |
| 650 |
| 2 1 4 |
| 783 |
| |
| g(n): 17, h(n): 12, f(n): 29 |
| |
| Current board state: |
| 6 5 4 |
| 213 |

g(n): 17, h(n): 12, f(n): 29

Current board state:

103

765

284

g(n): 20, h(n): 9, f(n): 29

Current board state:

506

281

473

g(n): 16, h(n): 13, f(n): 29

Current board state:

534

162

078

g(n): 17, h(n): 12, f(n): 29

Current board state:

183

045

726

g(n): 22, h(n): 7, f(n): 29

| 4 3 5 |
|----------------------------------|
| 106 |
| 287 |
| g(n): 19, h(n): 10, f(n): 29 |
| 8(11)1 13) 11(11)1 13) 1(11)1 13 |
| Current board state: |
| 5 3 4 |
| 180 |
| 7 2 6 |
| |
| g(n): 18, h(n): 11, f(n): 29 |
| |
| Current board state: |
| 013 |
| 278 |
| 456 |
| |
| g(n): 19, h(n): 10, f(n): 29 |
| |
| Current board state: |
| 301 |
| 752 |
| 486 |
| |
| g(n): 20, h(n): 9, f(n): 29 |
| |
| Current board state: |

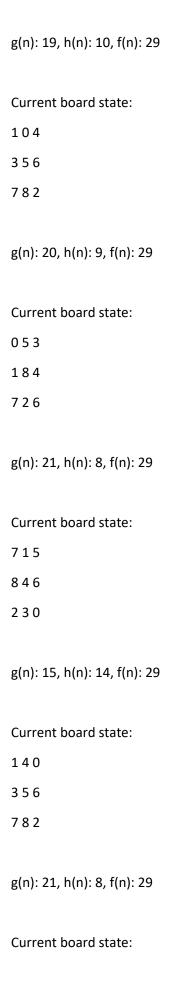
```
184
```

Current board state:

Current board state:

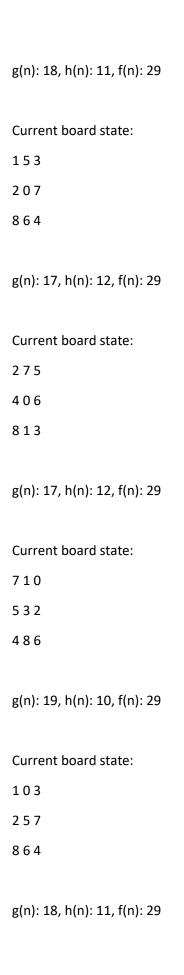
Current board state:

Current board state:



| 315 |
|------------------------------|
| 2 4 8 |
| 076 |
| |
| g(n): 17, h(n): 12, f(n): 29 |
| |
| Current board state: |
| 482 |
| 576 |
| 103 |
| |
| g(n): 18, h(n): 11, f(n): 29 |
| |
| Current board state: |
| 731 |
| 582 |
| 406 |
| |
| g(n): 18, h(n): 11, f(n): 29 |
| |
| Current board state: |
| 213 |
| 076 |
| 8 5 4 |
| |
| g(n): 20, h(n): 9, f(n): 29 |
| |
| Current board state: |
| 7.0.1 |

5 3 2



| 153 |
|------------------------------|
| 027 |
| 8 6 4 |
| |
| g(n): 18, h(n): 11, f(n): 29 |
| |
| Current board state: |
| 153 |
| 8 2 6 |
| 0 4 7 |
| |
| g(n): 21, h(n): 8, f(n): 29 |
| |
| Current board state: |
| 630 |
| 184 |
| 275 |
| |
| g(n): 15, h(n): 14, f(n): 29 |
| |
| Current board state: |
| 150 |
| 467 |
| 823 |
| |
| g(n): 19, h(n): 10, f(n): 29 |
| |
| Current board state: |
| 521 |

g(n): 17, h(n): 12, f(n): 29

Current board state:

186

427

503

g(n): 18, h(n): 11, f(n): 29

Current board state:

721

406

385

g(n): 19, h(n): 10, f(n): 29

Current board state:

256

384

701

g(n): 16, h(n): 13, f(n): 29

Current board state:

186

427

053

g(n): 19, h(n): 10, f(n): 29

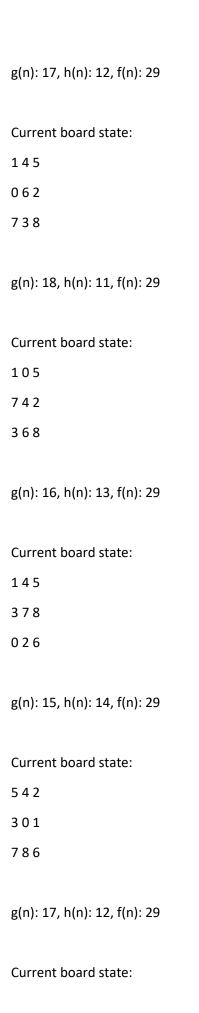
```
108
```

Current board state:

Current board state:

Current board state:

Current board state:



742

368

g(n): 17, h(n): 12, f(n): 29

Current board state:

123

804

765

g(n): 21, h(n): 8, f(n): 29

Goal state reached!

Steps to solve with Manhattan Distance heuristic: 21