# Part3. Pranav Gujjar. Registration: 100443924

**Question 3.1. How would you modify your IDA\* code to efficiently handle the presence of blocked tiles in the puzzle? Please discuss the necessary changes to the algorithm.**

**Ans. 3.1**

To effectively handle blocked tiles in a puzzle, make the following adjustments:

1. **State Representation:** Update the state representation with details about the tiles that

are blocked. You may do this by showing each tile as either blocked or open.  
2. **Update get\_children Function:** Make necessary adjustments to the get\_children function

to produce child states that take blocked tiles into account. Be careful to prevent

movements into blocked tiles when creating child states.  
3. **Adjusting the Heuristic Function:** To take into consideration the existence of blocked

tiles, modify the heuristic function. The heuristic should steer the search away from

blocked regions, possibly by raising the goal's estimated cost if the path passes through

obstructed tiles.

4**. Cost computation:** Verify that the inclusion of blocked tiles in the cost computation is

appropriately taken into consideration. To penalize movements into blocked tiles, one

possible solution would be to increase the cost of those actions.   
5. **Implementing Validity Checks:** Make sure the generated states do not entail moving into

blocked tiles by implementing checks in the algorithm.

**Question 3.2. Additionally, explain how the algorithm ensures that it finds an optimal solution while navigating around the blocked tiles.**

**Ans. 3.2**

To effectively manage blocked tiles in the IDA\* method, More focus on the following:

1. **Cost Function:** To calculate the expense of getting from the starting state to a specific

state, IDA\* employs a cost function. IDA\* makes sure that it avoids paths that have

greater costs because of blocked tiles by appropriately specifying the cost function to

account for their presence.   
2. **Heuristic Function:** The cost estimate from a state to the objective state is given by the

heuristic function. IDA\* directs the search away from paths that pass through blocked

areas, encouraging the investigation of more viable paths by including information about

blocked tiles in the heuristic.  
3. **Search Methodology:** To make sure it investigates paths with progressively rising costs,

IDA\* uses a depth-first search strategy with iterative deepening. IDA\* can effectively

discover the best solution while avoiding blocked tiles by initially investigating low-cost

paths and then gradually raise the cost threshold.  
4. **Validity Checks:** IDA\* verifies that created states do not include moving into tiles that are

blocked. IDA\* avoids examining blocked routes by ignoring states that do not adhere to

this restriction.