# Part2. Pranav Gujjar. Registration: 100443924

**Question 2.1. What would be the changes to make to your IDA\* code of Part 1 to take into account this change to the n-tile problem specification?**

**Ans.2.1**

1. Adapt the **state representation** to handle circular row changes and wrap-around columns.   
2. Make necessary **updates to the heuristic function** to account for the new moves and

provide precise distance estimates to the target state.  
3. To **create legitimate child states** for the new moves, modify the get\_children function.  
4. **Modify the cost computation** to account for circular row shifts with no cost.

**Question 2.2. How would you modify your IDA\* code of part 1 to solve this problem efficiently?**

**Ans.2.2**

1. **Heuristic Optimization**: Modify the heuristic function of the code to effectively calculate distances while taking wrap-around columns and circular shifts into account. For precise distance estimates, use new methods such as pattern databases.   
2. **Data Structures & Algorithms:** To manage states generate child nodes and optimize for new moves, use effective data structures and algorithms. For efficient state tracking, use custom hashing.  
3. **Adjust the search parameters** (such as the maximum threshold) in IDA\* to better fit the particulars of the case. Consider making dynamic changes depending on heuristic correctness and performance measurements.

**Question 2.3. What physical design of an n-tile-like puzzle would correspond to this problem?**

**Ans.2.3**

To solve this challenge, an n-tile puzzle with a rectangular grid with a tile with a number in each cell might be physically designed. Circular row shifts can be represented by physically moving the rows up or down by one position without changing the relative locations of the tiles within each row. The grid features wrap-around columns, which means that the end column connects with the first column. To solve the puzzle and attain the desired state, the player would manipulate this physical grid by moving tiles or rows.