* Develop a recording system for a Jigsaw game that efficiently logs all player movements with minimal space requirement.

• The game has following features:
1. Jigsaw has 26*26 tiles (each tile of size 8*8 pixel)
2. Every tile can be fitted in any one of the 4 orientations.
3. Player is allowed to move square portion of tiles at a time.

(Hint: Use Huffman coding & matrix multiplication algorithms)

* Assumptions:

1. Grid Size: The game board consists of a 26×26 grid of tiles.

2. Unique Tiles: There are 128 unique tiles each with four possible orientations (0, 90, 180, 270).

3. Player movements: Players can move a square portion of tiles at a time, with each movement logged as a change in the board state.

* Logic :-

1. Tile Representation:
- Each unique tile is assigned an ID (0-127).
- The orientation is represented by an additional 2-bit number (0-3)

2. Movement Represented:
- A movement can be represented by the coordinates of the top left corner of the square portion being moved and the size of the square.

- We need to log the state before and after the move to determine the changes.

3. Huffman Coding.
- Huffman coding can be used to encode the unique tiles and their orientations efficiently, based on their frequency of occurrence in the game.

4. Matrix Chain Multiplication:
- This technique helps optimize the sequence of movements ensuring minimal storage usage.

* Notes/Calculations on Encoding and Storage Efficiency.

° Initial storage calculations:
- Each move in Jigsaw game is recorded based on the rotation of a subgrid. The encoded information includes:
  - Top-left corner of the subgrid (x_1, y_1)
  - Bottom-right corner of the subgrid (x_2, y_2)

- Each coordinate (x or y) ranges from 0 to 25. Therefore each coordinate requires 5 bit representation.

- Thus each move requires: $5 \times 4 = 20$ bits.

• storage efficiency :

- Huffman Coding : By assigning shorter codes to frequently used tiles, we reduce the average number of bits required per tile.

- Movement Logging : Only changes in the board state are logged, minimizing storage by avoiding the need to store the entire board state.

——— * * * ———