8 puzzle dfs.

* Initialize the Node with State, parent, action

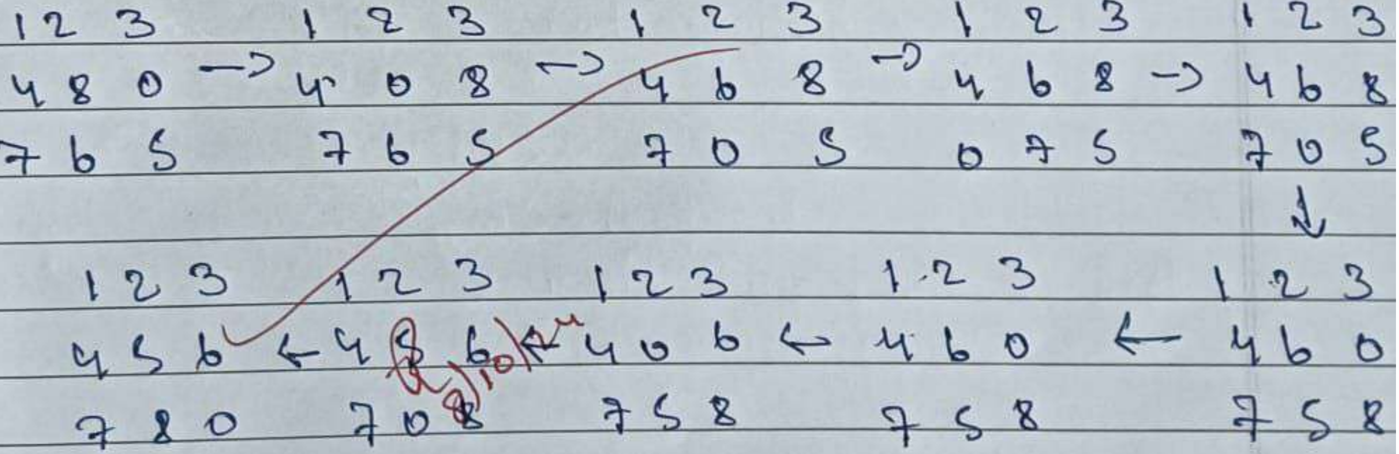
* Class puzzle:

```
Intialize the Self, start, goal
def neighbours (self, State)
    initialize matx, (row, col) = State
    Create Empty list results.

    if row > 0 then do
        copy mat to mat 1
        move the Empty tiles to up
        append State to result

    if col > 0 then do
        copy mat to mat 1
        move the Empty tiles to left
        append state to results

    if row < 2 then do
        copy mat to mat 1
        move the Empty tiles to down
        append State to results

    if col < 2 them do
        copy mat to mat1
        move the Empty tiles to Right
        append State to results
```

```
def dfs (self):
    Create the root node
    Create frontier stack for DFS
    Create a set to store visited states

    While frontier:
        * Get the node from the top of the stack
        *if current state = Goal state:
            return solution path
        * Mark the current state as visited
        * for action, state in neighbors:
            if state is not in visited:
                Create child node
                Add the chid to the frontier stack
    Return None
```

print solution.

State space Tree



moves : 10