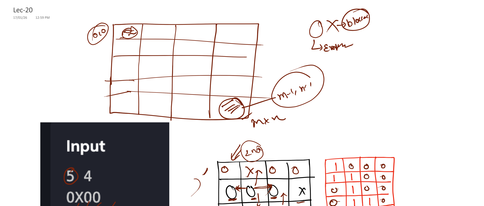
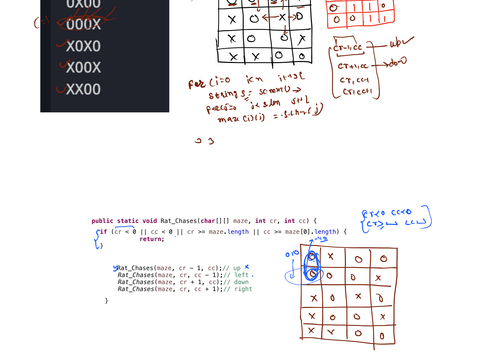
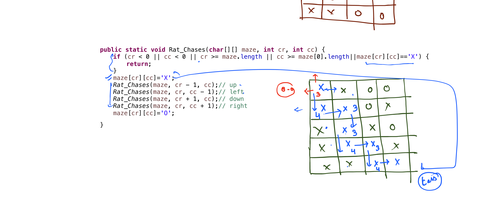
lec-20



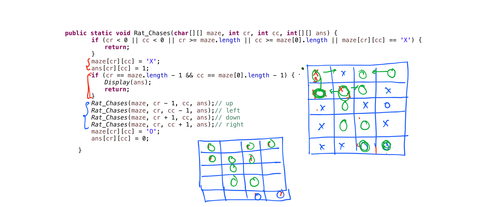
```
Input
5 4
0X00
000X
X0X0
X00X
XX00
```



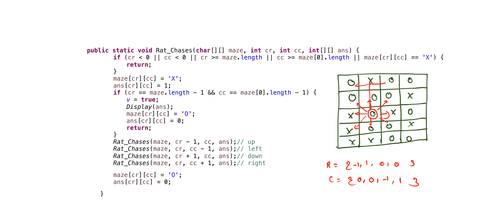
```
public static void Rat_Chases(char[][] maze, int cr, int cc) {
    if (cr < 0 || cc < 0 || cr >= maze.length || cc >= maze[0].length) {
        return;
    }

    Rat_Chases(maze, cr - 1, cc);// up
    Rat_Chases(maze, cr, cc - 1);// left .
    Rat_Chases(maze, cr + 1, cc);// down
    Rat_Chases(maze, cr, cc + 1);// right
}
```



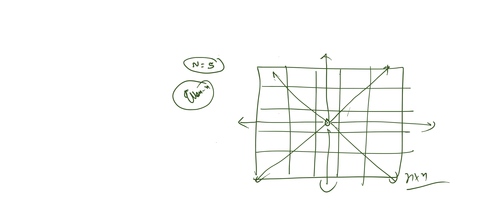
```
public static void Rat_Chases(char[][] maze, int cr, int cc) {
    if (cr < 0 || cc < 0 || cr >= maze.length || cc >= maze[0].length||maze[cr][cc]=='X') {
        return;
    }
    maze[cr][cc]='X';
    Rat_Chases(maze, cr - 1, cc);// up
    Rat_Chases(maze, cr, cc - 1);// left
    Rat_Chases(maze, cr + 1, cc);// down
    Rat_Chases(maze, cr, cc + 1);// right
    maze[cr][cc]='0';
}
```



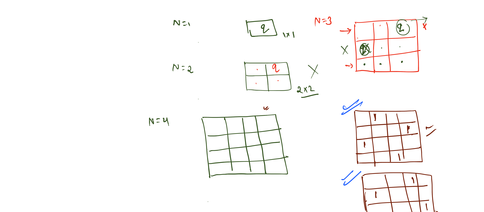
```
public static void Rat_Chases(char[][] maze, int cr, int cc, int[][] ans) {
    if (cr < 0 || cc < 0 || cr >= maze.length || cc >= maze[0].length || maze[cr][cc] == 'X') {
        return;
    }
    maze[cr][cc] = 'X';
    ans[cr][cc] = 1;
    if (cr == maze.length - 1 && cc == maze[0].length - 1) {
        Display(ans);
        return;
    }
    Rat_Chases(maze, cr - 1, cc, ans);// up
    Rat_Chases(maze, cr, cc - 1, ans);// left
    Rat_Chases(maze, cr + 1, cc, ans);// down
    Rat_Chases(maze, cr, cc + 1, ans);// right
    maze[cr][cc] = '0';
    ans[cr][cc] = 0;
}
```



```
public static void Rat_Chases(char[][] maze, int cr, int cc, int[][] ans) {
    if (cr < 0 || cc < 0 || cr >= maze.length || cc >= maze[0].length || maze[cr][cc] == 'X') {
        return;
    }
    maze[cr][cc] = 'X';
    ans[cr][cc] = 1;
    if (cr == maze.length - 1 && cc == maze[0].length - 1) {
        v = true;
        Display(ans);
        maze[cr][cc] = '0';
        ans[cr][cc] = 0;
        return;
    }
    Rat_Chases(maze, cr - 1, cc, ans);// up
    Rat_Chases(maze, cr, cc - 1, ans);// left
    Rat_Chases(maze, cr + 1, cc, ans);// down
    Rat_Chases(maze, cr, cc + 1, ans);// right

    maze[cr][cc] = '0';
    ans[cr][cc] = 0;
}
```



```
public static void Print(boolean[][] board, int row, int tq) {
    if (tq == 0) {
        Display(board);
        return;
    }
    for (int col = 0; col < board[0].length; col++) {
        if (isitsafe(board, row, col) == true) {
            board[row][col] = true;// Queen placed hogya
            Print(board, row + 1, tq - 1);
            board[row][col] = false;
        }
    }
```



```
public static boolean Print(int[][] grid, int row, int col) {

    if(grid[row][col]!=0) {
        return Print(grid, row, col+1);
    }
    else {
        for(int val=1; val<=9; val++) {
            if(isitsafe(grid,row,col,val)) {
                grid[row][col]=val;
                boolean ans =Print(grid, row, col+1);
                grid[row][col]=0;
            }
        }

    }
}
```