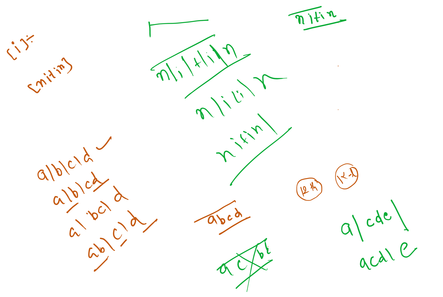
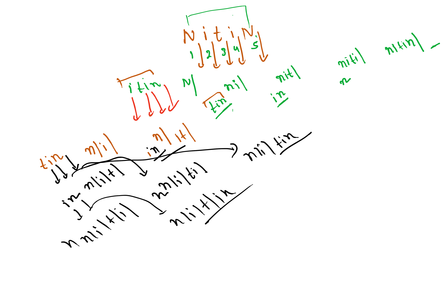
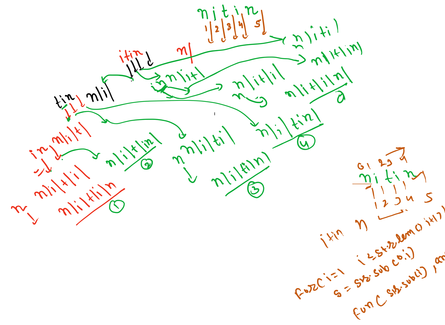
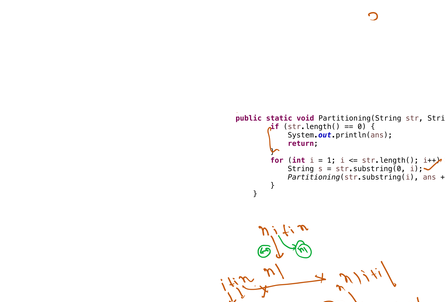
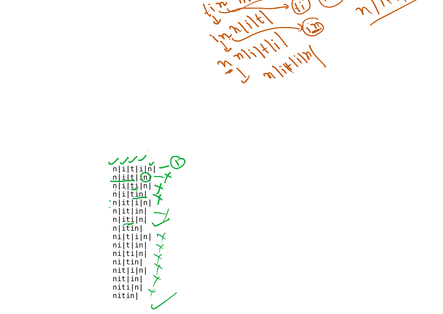
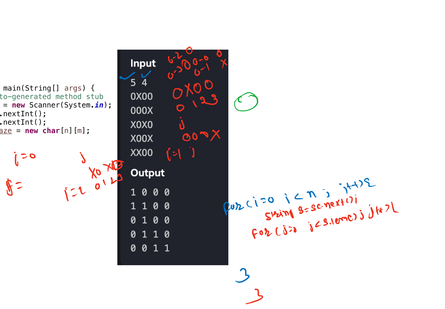






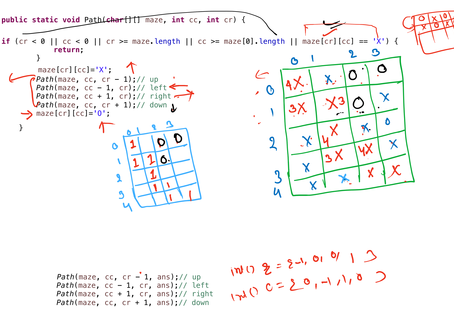






```
public static void Partitioning(String str, String ans) {
        if (str.length() == 0) {
            System.out.println(ans);
            return;
        }

        for (int i = 1; i <= str.length(); i++) {
            String s = str.substring(0, i);
            Partitioning(str.substring(i), ans + s + "|");
        }
    }
```




```
public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int m = sc.nextInt();
        char[][] maze = new char[n][m];
    }
```

Input

```
5 4
OXOO
OOOX
XOXO
XOOX
XXOO
```

Output

```
1 0 0 0
1 1 0 0
0 1 0 0
0 1 1 0
0 0 1 1
```



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



```
Path(maze, cc, cr - 1, ans);// up
Path(maze, cc - 1, cr, ans);// left
Path(maze, cc + 1, cr, ans);// right
Path(maze, cc, cr + 1, ans);// down
```