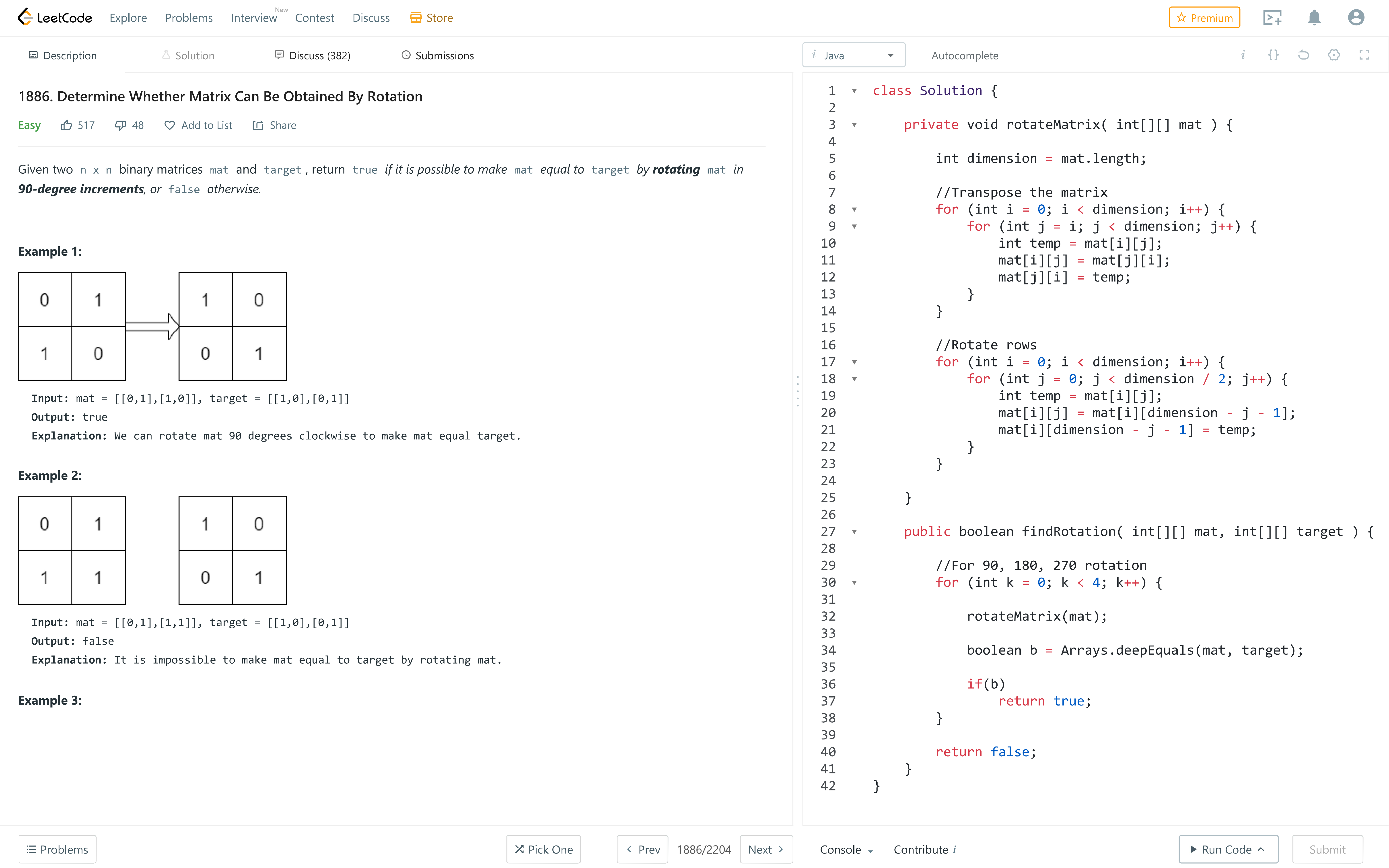

1886. Determine Whether Matrix Can Be Obtained By Rotation

Given two `n x n` binary matrices `mat` and `target`, return `true` *if it is possible to make* `mat` *equal to* `target` *by **rotating*** `mat` *in **90-degree increments**, or* `false` *otherwise.*

Example 1:

0	1
1	0

→

1	0
0	1

```
Input: mat = [[0,1],[1,0]], target = [[1,0],[0,1]]
Output: true
Explanation: We can rotate mat 90 degrees clockwise to make mat equal target.
```

Example 2:

0	1
1	1

1	0
0	1

```
Input: mat = [[0,1],[1,1]], target = [[1,0],[0,1]]
Output: false
Explanation: It is impossible to make mat equal to target by rotating mat.
```

Example 3:

```java
class Solution {

    private void rotateMatrix( int[][] mat ) {

        int dimension = mat.length;

        //Transpose the matrix
        for (int i = 0; i < dimension; i++) {
            for (int j = i; j < dimension; j++) {
                int temp = mat[i][j];
                mat[i][j] = mat[j][i];
                mat[j][i] = temp;
            }
        }

        //Rotate rows
        for (int i = 0; i < dimension; i++) {
            for (int j = 0; j < dimension / 2; j++) {
                int temp = mat[i][j];
                mat[i][j] = mat[i][dimension - j - 1];
                mat[i][dimension - j - 1] = temp;
            }
        }

    }

    public boolean findRotation( int[][] mat, int[][] target ) {

        //For 90, 180, 270 rotation
        for (int k = 0; k < 4; k++) {

            rotateMatrix(mat);

            boolean b = Arrays.deepEquals(mat, target);

            if(b)
                return true;
        }

        return false;
    }
}
```