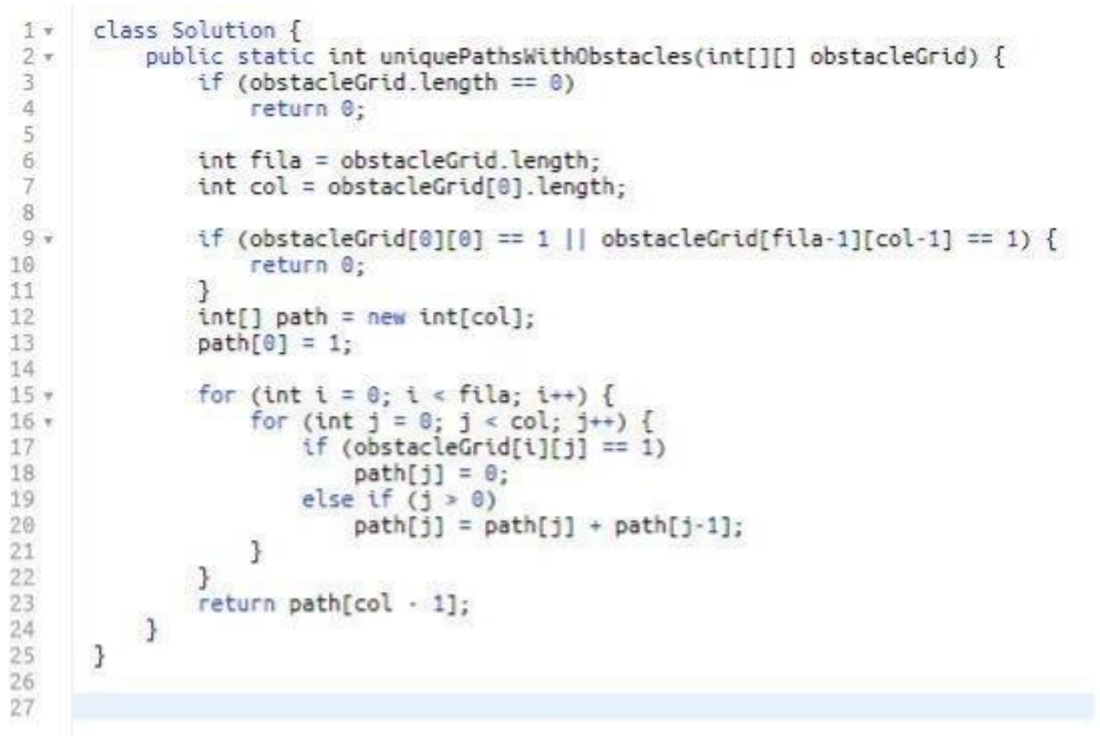
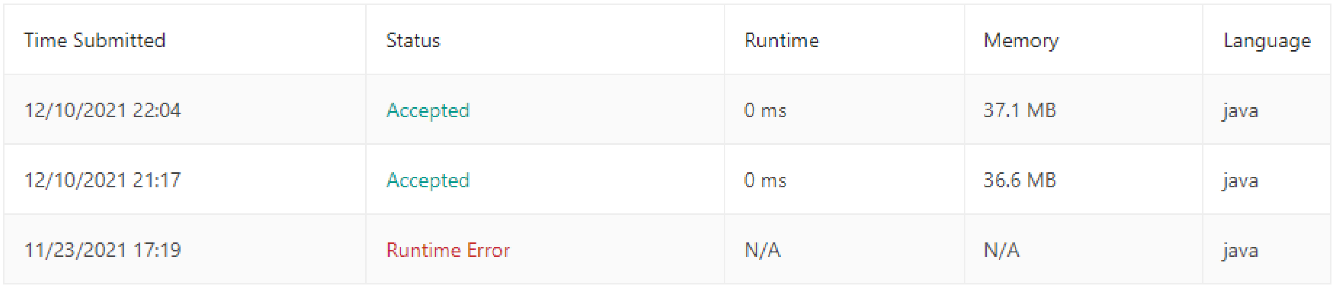
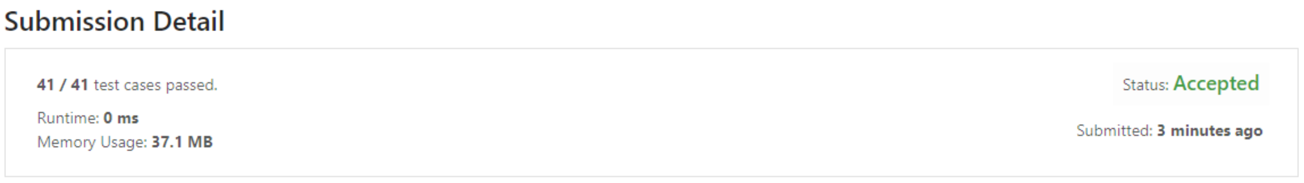


```java
class Solution {
    public static int uniquePathsWithObstacles(int[][] obstacleGrid) {
        if (obstacleGrid.length == 0)
            return 0;

        int fila = obstacleGrid.length;
        int col = obstacleGrid[0].length;

        if (obstacleGrid[0][0] == 1 || obstacleGrid[fila-1][col-1] == 1) {
            return 0;
        }
        int[] path = new int[col];
        path[0] = 1;

        for (int i = 0; i < fila; i++) {
            for (int j = 0; j < col; j++) {
                if (obstacleGrid[i][j] == 1)
                    path[j] = 0;
                else if (j > 0)
                    path[j] = path[j] + path[j-1];
            }
        }
        return path[col - 1];
    }
}
```



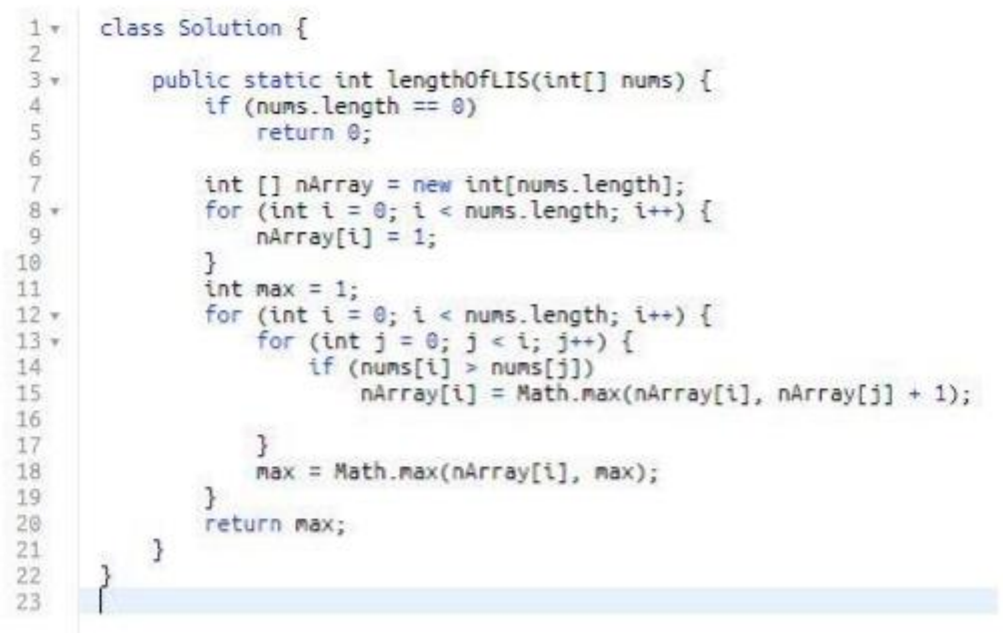
Time Submitted	Status	Runtime	Memory	Language
12/10/2021 22:04	Accepted	0 ms	37.1 MB	java
12/10/2021 21:17	Accepted	0 ms	36.6 MB	java
11/23/2021 17:19	Runtime Error	N/A	N/A	java



Submission Detail

41 / 41 test cases passed.
Runtime: **0 ms**
Memory Usage: **37.1 MB**

Status: Accepted
Submitted: **3 minutes ago**



```java
class Solution {

    public static int lengthOfLIS(int[] nums) {
        if (nums.length == 0)
            return 0;

        int [] nArray = new int[nums.length];
        for (int i = 0; i < nums.length; i++) {
            nArray[i] = 1;
        }
        int max = 1;
        for (int i = 0; i < nums.length; i++) {
            for (int j = 0; j < i; j++) {
                if (nums[i] > nums[j])
                    nArray[i] = Math.max(nArray[i], nArray[j] + 1);

            }
            max = Math.max(nArray[i], max);
        }
        return max;
    }
}

```

Time Submitted	Status	Runtime	Memory	Language
12/10/2021 21:15	Accepted	61 ms	38.4 MB	java
12/10/2021 21:13	Accepted	61 ms	38.5 MB	java

Submission Detail

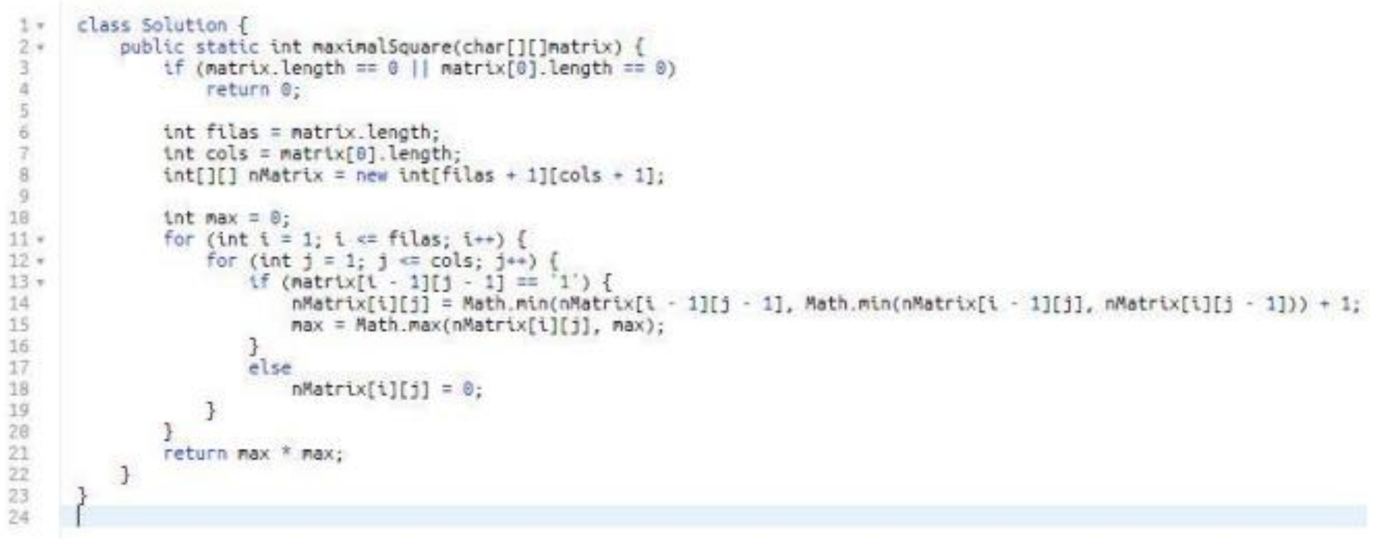
54 / 54 test cases passed.

Runtime: **61 ms**

Memory Usage: **38.4 MB**

Status: Accepted

Submitted: **0 minutes ago**



```java
class Solution {
    public static int maximalSquare(char[][]matrix) {
        if (matrix.length == 0 || matrix[0].length == 0)
            return 0;

        int filas = matrix.length;
        int cols = matrix[0].length;
        int[][] nMatrix = new int[filas + 1][cols + 1];

        int max = 0;
        for (int i = 1; i <= filas; i++) {
            for (int j = 1; j <= cols; j++) {
                if (matrix[i - 1][j - 1] == '1') {
                    nMatrix[i][j] = Math.min(nMatrix[i - 1][j - 1], Math.min(nMatrix[i - 1][j], nMatrix[i][j - 1])) + 1;
                    max = Math.max(nMatrix[i][j], max);
                }
                else
                    nMatrix[i][j] = 0;
            }
        }
        return max * max;
    }
}

```

Time Submitted	Status	Runtime	Memory	Language
12/10/2021 21:18	Accepted	4 ms	41.8 MB	java

Submission Detail

75 / 75 test cases passed.

Runtime: **4 ms**

Memory Usage: **41.8 MB**

Status: **Accepted**

Submitted: **54 minutes ago**