

Asignatura: Análisis y Diseño de Algoritmo - Laboratorio 'B'

Alumno: Hincho Jove, Angel Eduardo

Semestre: 2021 - B

Este breve informe complementa lo desarrollado en el Laboratorio 08. Aquí se encontrarán las capturas de pantalla para los casos de pruebas de los ejercicios mientras que el código y relación de recurrencia se encuentran en GitHub.

- **Comentario:** Al momento de entregar el código para los diferentes problemas en las páginas web se obviaron ciertos detalles como el método principal o los comentarios de la documentación.

1. Unique Path - Casos de Prueba

The screenshot shows a LeetCode interface with the 'Unique Paths' problem solved in Java. The code is as follows:

```
1 class Solution {
2     public int uniquePathsWithObstacles(int[][] grid) {
3         // Cols debe ser igual a grid[0]
4         int rows = grid.length, cols = grid[0].length;
5         // Aqui es int[rows][cols], anteriormente puse [rows][rows]
6         int[][] path = new int[rows][cols];
7         boolean isObstacle = false;
8         // Debemos llenar la primera fila y columna
9         for(int i = 0; i < cols; i++) {
10             if(isObstacle || grid[0][i] == 1) {
11                 isObstacle = true;
12                 path[0][i] = 0;
13             } else {
14                 path[0][i] = 1;
15             }
16         }
17         isObstacle = false;
18         for(int i = 0; i < rows; i++) {
19             if(isObstacle || grid[i][0] == 1) {
20                 isObstacle = true;
21                 path[i][0] = 0;
22             } else {
23                 path[i][0] = 1;
24             }
25         }
26         // Ahora debemos llenar las rutas por DP
27         // Verificamos si es un obstaculo y sus ancestros
28         for(int i = 1; i < rows; i++) {
29             for(int j = 1; j < cols; j++) {
30                 if(grid[i][j] == 1) {
31                     path[i][j] = 0;
32                 } else {
33                     path[i][j] = path[i-1][j] + path[i][j-1];
34                 }
35             }
36         }
37         return path[rows-1][cols-1];
38     }
39 }
```

The left sidebar shows the submission details: Runtime: 0 ms, faster than 100.00% of Java online submissions; Memory Usage: 38 MB, less than 84.56% of Java online submissions. The table below shows the submission history:

Time Submitted	Status	Runtime	Memory
12/03/2021 21:11	Accepted	0 ms	38 MB
12/03/2021 17:33	Accepted	0 ms	38.3 MB
12/03/2021 16:57	Accepted	0 ms	38.1 MB
12/03/2021 16:57	Accepted	0 ms	38 MB
12/03/2021 16:57	Accepted	0 ms	38.1 MB
12/03/2021 16:56	Accepted	0 ms	38.1 MB
12/03/2021 16:56	Accepted	0 ms	38.4 MB
12/03/2021 16:56	Accepted	0 ms	38.3 MB

2. Book Shop - Casos de Prueba

The screenshot shows the CSES 'Book Shop' problem page. The submission details are as follows:

Task:	Book Shop
Sender:	ahincho
Submission time:	2021-12-04 04:18:24
Language:	Java
Status:	READY
Result:	ACCEPTED

The test results table is as follows:

test	verdict	time
#1	ACCEPTED	0.13 s
#2	ACCEPTED	0.19 s
#3	ACCEPTED	0.22 s
#4	ACCEPTED	0.22 s
#5	ACCEPTED	0.13 s
#6	ACCEPTED	0.94 s
#7	ACCEPTED	0.93 s
#8	ACCEPTED	0.95 s
#9	ACCEPTED	0.94 s
#10	ACCEPTED	0.94 s
#11	ACCEPTED	0.94 s
#12	ACCEPTED	0.13 s
#13	ACCEPTED	0.94 s
#14	ACCEPTED	0.13 s

The right sidebar shows a list of dynamic programming problems: Coin Combinations II, Removing Digits, Grid Paths, Book Shop (checked), Array Description, Counting Towers, Edit Distance, Rectangle Cutting. The 'Your submissions' table is as follows:

Submission Time	Status
2021-12-04 04:18:24	✓
2021-12-04 04:18:15	✓
2021-12-04 04:17:07	✓
2021-12-04 04:16:59	✓
2021-12-04 04:16:50	✓
2021-12-04 04:16:43	✓
2021-12-04 04:16:33	✓
2021-12-04 04:16:23	✓
2021-12-04 04:16:16	✓
2021-12-04 04:15:59	✓
2021-12-03 15:23:22	✓
2021-12-03 15:23:10	✓

3. Length of Longest Increasing SubSequence - Casos de Prueba

The screenshot shows a LeetCode interface with a Java solution for the problem 'Length of Longest Increasing SubSequence'. The code uses a dynamic programming approach with a 1D array 'length' to store the length of the longest increasing subsequence ending at each index. The runtime is 64 ms and memory usage is 38.7 MB.

```
1 import java.util.Arrays;
2
3 class Solution {
4
5     public int lengthOfLIS(int[] nums) {
6         int n = nums.length;
7         if(n == 0) {
8             return 0;
9         }
10        // Para este problema necesitaremos dos unidades
11        // de almacenamiento, dos arrays unidimensionales
12        int[] length = new int[n];
13        Arrays.fill(length, 1);
14        int lis = 1;
15        for(int i = 1; i < n; i++) {
16            for(int j = 0; j < i; j++) {
17                if(nums[j] < nums[i] && length[j] + 1 > length[i]) {
18                    length[i] = length[j] + 1;
19                }
20            }
21            lis = Math.max(lis, length[i]);
22        }
23        return lis;
24    }
25 }
```

Submission history table:

Time Submitted	Status	Runtime	Memory
12/03/2021 21:25	Accepted	64 ms	38.7 MB
12/03/2021 21:24	Accepted	63 ms	38.7 MB
12/03/2021 21:24	Accepted	65 ms	38.6 MB
12/03/2021 21:23	Accepted	64 ms	38.4 MB
12/03/2021 21:23	Accepted	64 ms	38.4 MB

4. Rectangle Cutting - Casos de Prueba

The screenshot shows a CSES problem set solution for 'Rectangle Cutting'. The submission is accepted, and the test results table shows 21 tests passed.

Submission details:

- Task: Rectangle Cutting
- Sender: ahincho
- Submission time: 2021-12-04 04:29:30
- Language: Java
- Status: READY
- Result: ACCEPTED

Test results:

test	verdict	time
#1	ACCEPTED	0.13 s
#2	ACCEPTED	0.13 s
#3	ACCEPTED	0.13 s
#4	ACCEPTED	0.13 s
#5	ACCEPTED	0.13 s
#6	ACCEPTED	0.24 s
#7	ACCEPTED	0.23 s
#8	ACCEPTED	0.17 s
#9	ACCEPTED	0.20 s
#10	ACCEPTED	0.15 s
#11	ACCEPTED	0.19 s
#12	ACCEPTED	0.17 s
#13	ACCEPTED	0.24 s
#14	ACCEPTED	0.15 s
#15	ACCEPTED	0.16 s
#16	ACCEPTED	0.25 s
#17	ACCEPTED	0.15 s
#18	ACCEPTED	0.19 s
#19	ACCEPTED	0.17 s
#20	ACCEPTED	0.44 s
#21	ACCEPTED	0.44 s

Dynamic Programming list:

- Array Description
- Counting Towers
- Edit Distance
- Rectangle Cutting
- Money Sums
- Removal Game
- Two Sets II
- Increasing Subsequence

Your submissions:

Submission Time	Status
2021-12-04 04:29:30	✓
2021-12-04 04:29:19	✓
2021-12-04 04:29:08	✓
2021-12-04 04:28:12	✓
2021-12-04 04:27:42	✓
2021-12-04 04:26:54	✓
2021-12-04 04:26:44	✓
2021-12-03 16:45:34	✓
2021-12-03 16:45:26	✓
2021-12-03 16:45:17	✓
2021-12-03 16:45:08	✓
2021-12-03 16:44:53	✓
2021-12-03 16:44:26	✓

5. Area of Maximal Square - Casos de Prueba

LeetCode

Explore

Problems

Interview

Contest

Discuss

Store

LeetCode Challenge + GIVEAWAY! Premium

Description

Solution

Discuss (9...)

Submissions

Java

Autocomplete

Success

Details >

Runtime: 3 ms, faster than 98.64% of Java online submission Square.

Memory Usage: 42 MB, less than 86.53% of Java online sub Maximal Square.

Next challenges:

Maximal Rectangle

Largest Plus Sign

Show off your acceptance:

f

t

in

Time Submitted	Status	Runtime	Memory
12/03/2021 21:33	Accepted	3 ms	42 MB
12/03/2021 21:33	Accepted	4 ms	42 MB
12/03/2021 21:33	Accepted	4 ms	42.3 MB
12/03/2021 21:32	Accepted	4 ms	42 MB
12/03/2021 21:32	Accepted	4 ms	42 MB
12/03/2021 21:32	Accepted	4 ms	42 MB
12/03/2021 21:32	Accepted	4 ms	42.2 MB
12/03/2021 21:32	Accepted	4 ms	42 MB

```
1 * class Solution {
2 *     public int maximalSquare(char[][] matrix) {
3 *
4 *         int rows = matrix.length;
5 *         if(rows == 0) {
6 *             return 0;
7 *         }
8 *
9 *         int cols = matrix[0].length;
10 *         int[][] board = new int[rows + 1][cols + 1];
11 *         int biggest = 0;
12 *
13 *         for(int i = 1; i < rows + 1; i++) {
14 *             for (int j = 1; j < cols + 1; j++) {
15 *                 if(matrix[i - 1][j - 1] == '1') {
16 *                     board[i][j] = 1 + Math.min(board[i - 1][j], Math.min(board[i][j - 1], board[i - 1][j - 1]));
17 *                     if(biggest < board[i][j]) {
18 *                         biggest = board[i][j];
19 *                     }
20 *                 }
21 *             }
22 *         }
23 *
24 *         return biggest * biggest;
25 *     }
26 * }
```

Your previous code was restored from your local storage. [Reset to default](#)

Console

Contribute

Run Code

Submit

angelhincho21

My List

My Playground

Notebook

Submissions

Sessions

Progress

Points

Subscription

Orders

Sign out