

519. Random Flip Matrix

Description

There is an $m \times n$ binary grid `matrix` with all the values set `0` initially. Design an algorithm to randomly pick an index `(i, j)` where `matrix[i][j] == 0` and flips it to `1`. All the indices `(i, j)` where `matrix[i][j] == 0` should be equally likely to be returned.

Optimize your algorithm to minimize the number of calls made to the **built-in** random function of your language and optimize the time and space complexity.

Implement the `Solution` class:

- `Solution(int m, int n)` Initializes the object with the size of the binary matrix `m` and `n`.
- `int[] flip()` Returns a random index `[i, j]` of the matrix where `matrix[i][j] == 0` and flips it to `1`.
- `void reset()` Resets all the values of the matrix to be `0`.

Example 1:

Input

```
["Solution", "flip", "flip", "flip", "reset", "flip"]  
[[3, 1], [], [], [], [], []]
```

Output

```
[null, [1, 0], [2, 0], [0, 0], null, [2, 0]]
```

Explanation

```
Solution solution = new Solution(3, 1);  
solution.flip(); // return [1, 0], [0,0], [1,0], and [2,0] should be equally likely to be returned.  
solution.flip(); // return [2, 0], Since [1,0] was returned, [2,0] and [0,0]  
solution.flip(); // return [0, 0], Based on the previously returned indices, only [0,0] can be returned.  
solution.reset(); // All the values are reset to 0 and can be returned.  
solution.flip(); // return [2, 0], [0,0], [1,0], and [2,0] should be equally likely to be returned.
```

Constraints:

- $1 \leq m, n \leq 10^4$
- There will be at least one free cell for each call to `flip`.
- At most `1000` calls will be made to `flip` and `reset`.

