Second Distinct Column Maximum

Given a 2D numpy array A of shape (m, n), for each column independently return the second *distinct* maximum value. That is, for column j, let $M_j = \max_i A_{ij}$. If the maximum value occurs multiple times, you must ignore all occurrences of M_j and return the largest value strictly less than M_j.

It is guaranteed that each column contains at least two distinct values.

Input Format

2D numpy array A of shape (m, n)

Output Format

return a 1D numpy array of shape $(n,)$ where the j-th element is the second distinct maximum of column j.

Constraints

- $1 \leq m \leq 1000$ (number of rows)
- $1 \leq n \leq 1000$ (number of columns)
- $-10000 \leq A_{ij} \leq 10000$ (array elements)
- Each column has at least two distinct values

Sample Input

```
A = np.array([[7.0, 3.0, 5.0, 5.0],
              [7.0, 1.0, 9.0, 2.0],
              [4.0, 3.0, 9.0, -1.0],
              [0.0, 8.0, 9.0, 5.0]])
```

Sample Output

```
np.array([4.0, 3.0, 5.0, 2.0])
```

Implementation

Goal: Fill in the following function:

```
import numpy as np
def second_max_per_column(A):
    ...
    return ...
if __name__ == "__main__":
    # You can test anything inside this block and can send it to grader
    # The grader will use only the function that you have implemented
    # !!! DO NOT write anything outside this block
    A = np.array([
        [7.0, 3.0, 5.0, 5.0],
        [7.0, 1.0, 9.0, 2.0],
        [4.0, 3.0, 9.0, -1.0],
        [0.0, 8.0, 9.0, 5.0]
    ])
    print(second_max_per_column(A))
```