Matrix (2D Array) Traversal

For each of the following create a function that takes a dynamic 2D array (assorted sizes of matrix) and returns a 1D array containing the output (items) from the given traversal. The best approach would be to create separate files for each of the numbered challenges and import these into a central main.py file for testing.

## Sample matrix…

[['🍌','🍎','😀','🐉'],

['👺','🍺','🍩','🚴‍♂️'],

['🚘','🦑','🚆','🏝️'],

['🌆','🛹','🕺','🍕']]

## Challenges

1. Row-wise Traversal:
   1. Forward: Iterate through each row from left to right.

['🍌','🍎','😃','🐉','👺','🍺','🍩','🚴','🚘','🦑','🚆','🏝️','🌆','🛹','🕺','🍕']

* 1. Reverse: Iterate through each row from right to left.  
     ['🐉','😃','🍎','🍌','🚴','🍩','🍺','👺','🏝️','🚆','🦑','🚘','🍕','🕺','🛹','🌆']

1. Column-wise Traversal:
   1. Forward: Iterate through each column from top-left to bottom-right.

['🍌','👺','🚘','🌆','🍎','🍺','🦑','🛹','😃','🍩','🚆','🕺','🐉','🚴','🏝️','🍕']

* 1. Reverse: Iterate through each column from bottom-left to top-right.

['🌆','🚘','👺','🍌','🛹','🦑','🍺','🍎','🕺','🚆','🍩','😃','🍕','🏝️','🚴','🐉']

1. Diagonal Traversal:
   1. Primary:
      1. Forward: Top-left to bottom-right.

['🍌','🍺','🚆','🍕']

* + 1. Reverse: Bottom-right to top-left.

['🍕','🚆','🍺','🍌']

* 1. Secondary:
     1. Forward: Top-right to bottom-left.

['🐉','🍩','🦑','🌆']

* + 1. Reverse: Bottom-left to top-right.

['🌆','🦑','🍩','🐉']

1. Spiral Traversal:
   1. Forward: Move in a clockwise spiral pattern from the top-left.

['🍌','🍎','😃','🐉','🚴','🏝️','🍕','🕺','🛹','🌆','🚘','👺','🍺','🍩','🚆','🦑']

* 1. Reverse: Move in an anti-clockwise spiral pattern from the top-left.

['🍌','👺','🚘','🌆','🛹','🕺','🍕','🏝️','🚴','🐉','😃','🍎','🍺','🦑','🚆','🍩']

1. Zigzag Traversal:
   1. Row First: Traverse rows left to right, then right to left, and repeat.

['🍌','👺','🍎','😃','🍺','🚘','🌆','🦑','🍩','🐉','🚴','🚆','🛹','🕺','🏝️','🍕']

* 1. Column First: Traverse rows right to left, then left to right, and repeat.

['🍌','🍎','👺','🚘','🍺','😃','🐉','🍩','🦑','🌆','🛹','🚆','🚴','🏝️','🕺','🍕']

1. Snake Traversal:
   1. Forward: Traverse rows left to right, then right to left, and move to the next row.

['🍌','🍎','😃','🐉','🚴','🍩','🍺','👺','🚘','🦑','🚆','🏝️','🍕','🕺','🛹','🌆']

* 1. Reverse: Traverse rows right to left, then left to right, and move to the next row.

['🐉','😃','🍎','🍌','👺','🍺','🍩','🚴','🏝️','🚆','🦑','🚘','🌆','🛹','🕺','🍕']

* 1. Down: Traverse columns top to bottom, then right to left, and move to the next row.

['🍌','👺','🚘','🌆','🛹','🦑','🍺','🍎','😃','🍩','🚆','🕺','🍕','🏝️','🚴','🐉']

* 1. Up: Traverse columns bottom to top, then top to bottom, and move to the next column.

['🌆','🚘','👺','🍌','🍎','🍺','🦑','🛹','🕺','🚆','🍩','😃','🐉','🚴','🏝️','🍕']

1. Boundary Traversal:
   1. Top:
      1. Forward: Left to right.

['🍌','🍎','😃','🐉']

* + 1. Reverse: Right to left.

['🐉','😃','🍎','🍌']

* 1. Right:
     1. Forward: Top to bottom.

['🐉','🚴','🏝️','🍕']

* + 1. Reverse: Bottom to top.

['🍕','🏝️','🚴','🐉']

* 1. Bottom:
     1. Forward: Left to right.

['🌆','🛹','🕺','🍕']

* + 1. Reverse: Right to left.

['🍕','🕺','🛹','🌆']

* 1. Left:
     1. Forward: Top to bottom.

['🍌','👺','🚘','🌆']

* + 1. Reverse: Bottom to top.

['🌆','🚘','👺','🍌']

* 1. Full:
     1. Forward: From top-left corner clockwise.

['🍌','🍎','😃','🐉','🚴','🏝️','🍕','🕺','🛹','🌆','🚘','👺']

* + 1. Reverse: From top-left corner anticlockwise.

['🍌','👺','🚘','🌆','🛹','🕺','🍕','🏝️','🚴','🐉','😃','🍎']