Solution Review: Problem Challenge 1

We'll cover the following

Problem Statement (hard)#

Given a binary matrix representing an image, we want to flip the image horizontally, then invert it.

To flip an image horizontally means that each row of the image is reversed. For example, flipping [0, 1, 1] horizontally results in [1, 1, 0].

To invert an image means that each 0 is replaced by 1, and each 1 is replaced by 0. For example, inverting [1, 1, 0] results in [0, 0, 1].

Example 1:

```
Input: [
  [1,0,1],
  [1,1,1],
  [0,1,1]
]
Output: [
  [0,1,0],
  [0,0,0],
  [0,0,1]
]
```

Explanation: First reverse each row: `[[1,0,1],[1,1,1],[1,1,0]]`. Then, invert the image: `[[0,1,0],[0,0,0],[0,0,1]]`

Example 2:

```
Input: [
  [1,1,0,0],
  [1,0,0,1],
  [0,1,1,1],
  [1,0,1,0]
]
Output: [
  [1,1,0,0],
  [0,1,1,0],
  [0,0,0,1],
  [1,0,1,0]
]
```

Explanation: First reverse each row: `[[0,0,1,1],[1,0,0,1],[1,1,1,0],[0,1,0,1]]`. Then invert the image: `[[1,1,0,0],[0,1,1,0],[0,0,0,1],[1,0,1,0]]`

Solution#

- **Flip**: We can flip the image in place by replacing *ith* element from left with the *ith* element from the right.
- **Invert**: We can take XOR of each element with 1. If it is 1 then it will become 0 and if it is 0 then it will become 1.

Code#

Here is what our algorithm will look like:

Java | Python3 | C++ | JS

```python
def flip_an_invert_image(matrix):
  C = len(matrix)
  for row in matrix:
    for i in range((C+1)//2):
      row[i], row[C - i - 1] = row[C - i - 1] ^ 1, row[i] ^ 1

  return matrix

def main():
    print(flip_an_invert_image([[1,0,1], [1,1,1], [0,1,1]]))
    print(flip_an_invert_image([[1,1,0,0],[1,0,0,1],[0,1,1,1],[1,0,1,0]]))

main()
```

Run | Save | Reset

Time Complexity#

The time complexity of this solution is $O(n)$ as we iterate through all elements of the input.

Space Complexity#

The space complexity of this solution is $O(1)$.