# JavaScript Flipping an Image

## Challenge

Given an `n x n` binary matrix `image`, flip the image horizontally, then invert it, and return the resulting image.

To flip an image horizontally means that each row of the image is reversed.

- For example, flipping `[1,1,0]` horizontally results in `[0,1,1]`.

To invert an image means that each `0` is replaced by `1`, and each `1` is replaced by `0`.

- For example, inverting `[0,1,1]` results in `[1,0,0]`.

### 1$^{st}$ Example

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

## 2nd Example

```
Input: image = [[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]]
```

## Constraints

- `n == image.length`
- `n == image[i].length`
- `1 <= n <= 20`
- `images[i][j]` is either `0` or `1`.

# Solution

```
const flipAndInvertImage = (image) => {
    for (let row in image) {
        image[row] = image[row].reverse();
        image[row] = image[row].map(x => 1 - x);
    }

    return image;
};
```

## Explanation

I've created a function called `flipAndInvertImage` that takes an image as input and returns the modified image. The purpose of this function is to flip the image horizontally and invert the colors of each pixel.

The function begins by iterating through each row of the image using a `for` loop.

Inside the loop, the current row of the image is reversed using the `reverse()` method. This reverses the order of the elements in the row, effectively flipping the image horizontally.

Then, the `map()` method is used to iterate through each element in the reversed row. For each element, an arrow function `(x => 1 - x)` is applied. This function subtracts the element from `1`, effectively inverting the color. Assuming the elements in the image are either `0` or `1`, this operation will switch the value from `0` to `1` or from `1` to `0`.

The modified row is then assigned back to the `image[row]` to update the image with the flipped and inverted row.

After the loop finishes, the modified image is returned as the output of the function.

In summary, this function takes an image as input and performs two operations on each row: it reverses the order of the elements to flip the image horizontally, and it inverts the colors of each pixel. The modified image is then returned as the output of the function.

Author: Trevor Morin