LeetCode :- 861

🎯 Problem Understanding

We have a binary matrix where we can:

- Flip any row (0↔1)
- Flip any column (0↔1)
- Goal: Get maximum possible score (sum of binary numbers in each row)

🔄 Step-by-Step Solution

1️⃣ First Column Optimization

Why flip the first column?

- First digit in binary has highest value (2^(n-1))
- Example: In 4-bit number, first digit = 8, last digit = 1

```
// Before: [0,1,1,0] = 6
// After:  [1,0,0,1] = 9
if(grid[i][0] == 0) {
    // Flip entire row
    for(int j = 0; j < cols; j++)
        grid[i][j] = 1 - grid[i][j];
}
```

2️⃣ Column Optimization

For each remaining column:

- Count 1s and 0s
- Flip if zeros > ones (maximize 1s in each position)

```
int count1 = 0;
for(int i = 0; i < rows; i++)
```

```
    if(grid[i][j] == 1) count1++;
if(rows - count1 > count1) {  // if zeros > ones
    // Flip column
}
```

3️⃣ Score Calculation

Converting binary to decimal:

```
int x = 1;  // Start with 2^0
sum += grid[i][j] * x;
x *= 2;     // Multiply by 2 for each position (2^1, 2^2, ...)
```

🎮 Interactive Example

Initial Matrix:

```
0 0 1 1
1 0 1 0
1 1 0 0
```

After Step 1 (First column all 1s):

```
1 1 0 0
1 0 1 0
1 1 0 0
```

Final Matrix (After column flips):

```
1 1 1 1
1 0 1 0
1 1 0 0
```

Final Score = (15 + 10 + 12) = 39