Coding challenges

Mohamed Boucetta

Exercise 0.1 *Build a function which takes a square matrix whose entries are 0, 1 or 2 and returns 1 if there is a row and a column of 1, returns 2 if there is a row and a column of 2 and returns 0 otherwise. For instance*

$$M = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 2 \\ 1 & 2 & 0 \end{pmatrix} N = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 2 \\ 1 & 2 & 0 \end{pmatrix}, \ L = \begin{pmatrix} 1 & 2 & 1 \\ 1 & 2 & 0 \\ 2 & 2 & 2 \end{pmatrix},$$

for M the function will return 1, for N the function will return 0 and for L the function will return 2.

Solution : The idea is that if the items of a row (or a column) are all equal to 1 (resp. 2) then the row doesn't contains 0 and the sum of its items is equal to n (resp. $2n$), where n is the length of the row.

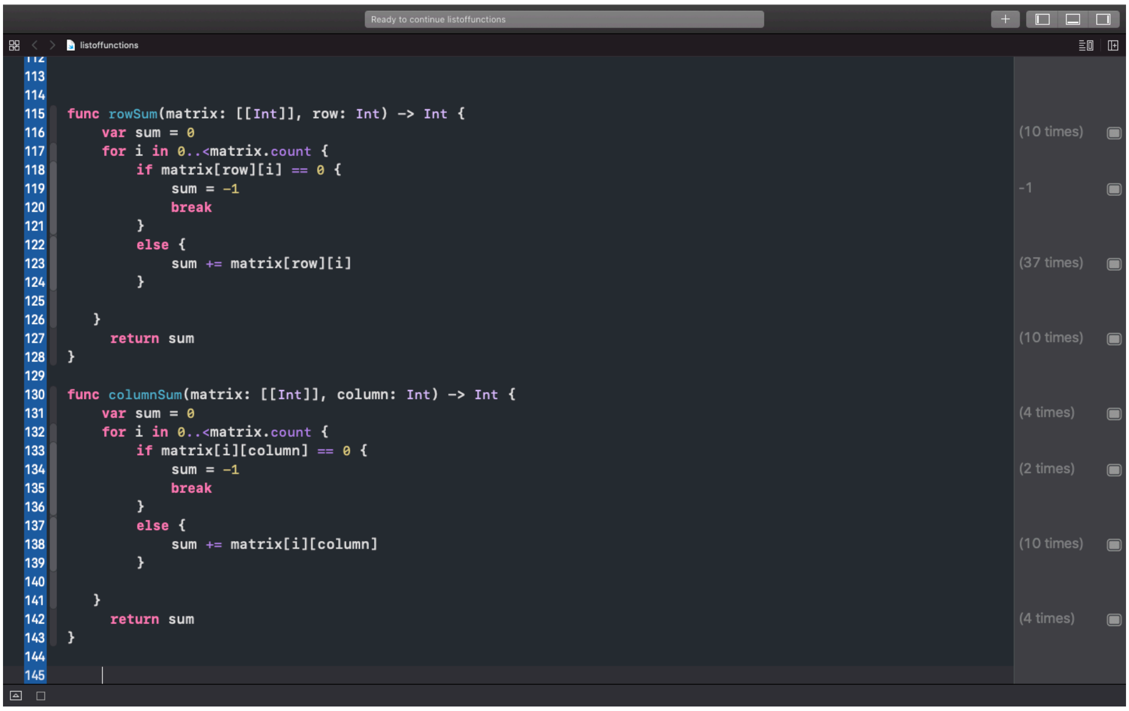
First introduce two functions sumRow and sumColumn. The function sumRow take a matrix and the index of a row and returns -1 if the row contains 0 or the sum of its items otherwise.

We set two counts, count1 and count2 to get, respectively, the number of rows with all items equal to 1 and the number of rows with all items equal to 2. This is done by the first for loop and by using the remark above.

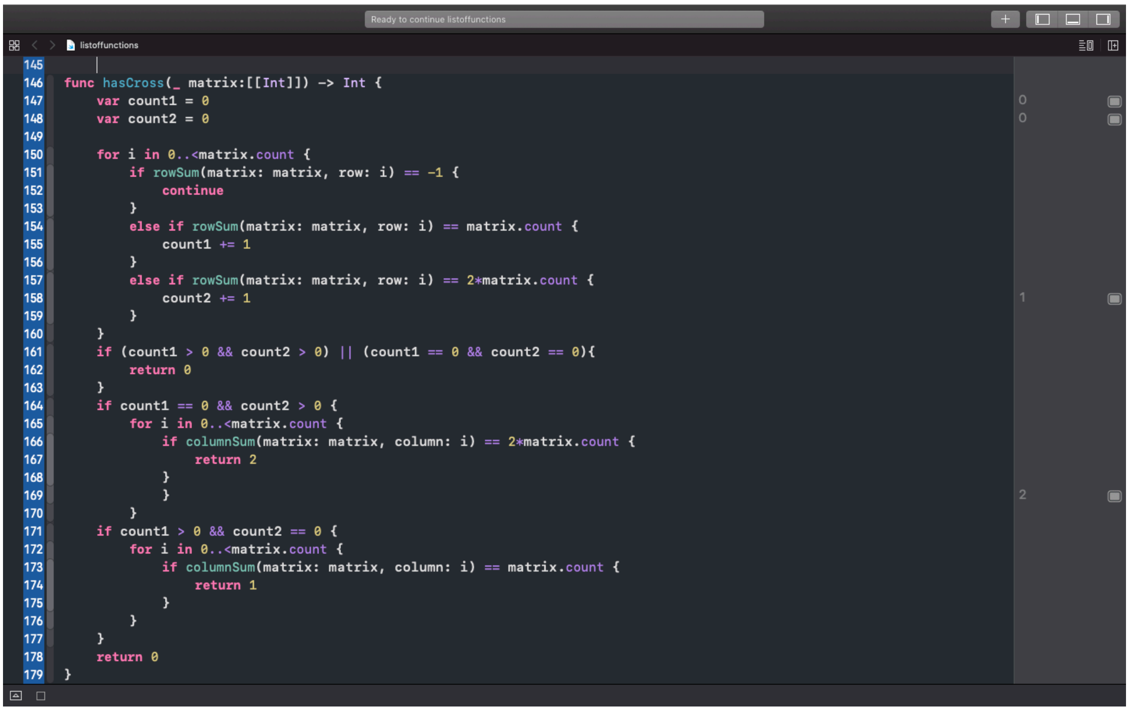
If count1 = 0 and count2 =0, we return 0, obviously. If count1 > 0 and count2 > 0, we are certain that there couldn't be neither a column with all its items equal to 1 or a column with all its items 2 and we return 0.

If count1 = 0 and count2 > 0, we look by the mean of a for loop of if there is a column with all its items equal to 2. We do the same if If count1 > 0 and count2 = 0.

The time complexity is $O(N^2)$ and the space complexity is $O(1)$.



```swift


func rowSum(matrix: [[Int]], row: Int) -> Int {
    var sum = 0                                              (10 times)
    for i in 0..<matrix.count {
        if matrix[row][i] == 0 {
            sum = -1                                         -1
            break
        }
        else {
            sum += matrix[row][i]                            (37 times)
        }

   }
     return sum                                              (10 times)
}

func columnSum(matrix: [[Int]], column: Int) -> Int {
    var sum = 0                                              (4 times)
    for i in 0..<matrix.count {
        if matrix[i][column] == 0 {
            sum = -1                                         (2 times)
            break
        }
        else {
            sum += matrix[i][column]                         (10 times)
        }

   }
     return sum                                              (4 times)
}

```



```swift
func hasCross(_ matrix:[[Int]]) -> Int {
    var count1 = 0
    var count2 = 0

    for i in 0..<matrix.count {
        if rowSum(matrix: matrix, row: i) == -1 {
            continue
        }
        else if rowSum(matrix: matrix, row: i) == matrix.count {
            count1 += 1
        }
        else if rowSum(matrix: matrix, row: i) == 2*matrix.count {
            count2 += 1
        }
    }
    if (count1 > 0 && count2 > 0) || (count1 == 0 && count2 == 0){
        return 0
    }
    if count1 == 0 && count2 > 0 {
        for i in 0..<matrix.count {
            if columnSum(matrix: matrix, column: i) == 2*matrix.count {
                return 2
            }
        }
    }
    if count1 > 0 && count2 == 0 {
        for i in 0..<matrix.count {
            if columnSum(matrix: matrix, column: i) == matrix.count {
                return 1
            }
        }
    }
    return 0
}
```