Slot **2** – Question **1 [Row reduce]** Marks 10

Gist: A custom row reduction where every row should pass a row condition and column should pass a column condition. If violated, we want to count the number of times it happens and print them

Tasks:

- Writing two loops one simple and the other nested for loop
- Print the number of violations of the conditions separately

---

Grading Scheme: MANUAL

ANY FORM OF HARD-CODING ATTRACTS FULL PENALTY.

[1 + 2 Marks] : Writing the row violation and column violation for loops correctly

[1 + 1 Mark] : Find the r and c correctly

========= GOLD CODE ======================

```
#include <stdio.h>
int M[10][10];

int main()
{
	int N;
	int i;
	int j;
	int row_violations=0;
	int col_violations=0;
	int starting[10];
	scanf("%d",&N);
	/* read the matrix */
	for(i=0;i<N;i++){
		for(j=0;j<N;j++){
			scanf("%d",&M[i][j]);
		}
	}
	/* check the row conditions */
	for(i=0;i<N;i++){
		j=0;
```

```
            while( j<N && (M[i][j]==0)){
                j++;
            }
            starting[i] = j; /* may be a row of 0s */
            if(j<N && (M[i][j]!=1)){
                row_violations++;
            }
        }

        /* check the column conditions */
        for(j=0;j<N;j++){
            int flag = 0;

            /* check if j is a start column for some row */
            for(i=0;i<N;i++){
                if(starting[i]==j){
                    flag = 1;
                    break;
                }
            }
            /* if it's a start row, there must be no other
               non-zero entries in column j */
            if(flag){
                int num_non_zeroes = 0;
                for(i=0;i<N;i++){
                    if(M[i][j]){
                        num_non_zeroes++;
                    }
                }
                if(num_non_zeroes>1){
                    col_violations++;
                }
            }
        }

        printf("%d %d\n", row_violations, col_violations);
        return 0;
}
```