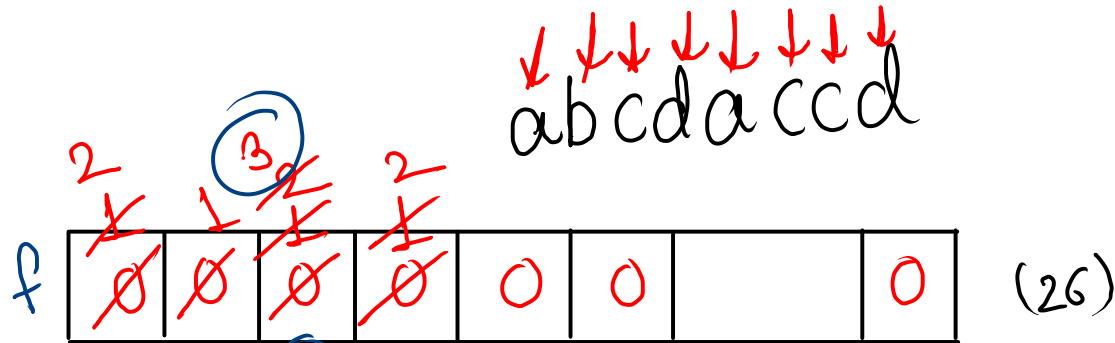


Maximum Freq Character



$arr[i] = arr[i] + 1;$
 $arr[i]++;$

a → 0
 b → 1
 c → 2
 d → 3
 e → 4
 ...
 z → 25

'a' - 'a' = 0
 'b' - 'a' = 1
 'c' - 'a' = 2
 'd' - 'a' = 3
 'a' - 'a' = 0
 'c' - 'a' = 2
 'c' - 'a' = 2
 'd' - 'a' = 3

ascii values

indexes

5 → f
 2 → c
 3 → d
 1 → b
 0 → a

```

public static void maxFreq(String str) {
    int[] freq = new int[26]; // because we have only 26 characters to count
    for (int i = 0; i < str.length(); i++) {
        char ch = str.charAt(i);
        int idx = ch - 'a';

        freq[idx] = freq[idx] + 1;
        // freq[idx]++;

    }

    int maxFreq = -1;
    int idx = -1;

    for (int i = 0; i < 26; i++) {
        int f = freq[i];
        if ( f > maxFreq ) {
            maxFreq = f;
            idx = i;
        }
    }

    char ans = (char)(idx + 'a');
    System.out.println(ans);
}

```

$$T.C = O(n + 26)$$

$$\approx O(n)$$

where, n is length of str

$$S.C = O(26)$$

$$\approx O(1)$$

$$\begin{aligned}
 'a' &= 97 \\
 + 2 \\
 \hline
 99 &= 'c'
 \end{aligned}$$

$$2 + 97 = 99 = 'c'$$

⇒ 2D array

Diagram illustrating a 2D array structure:

The array is represented as a grid with 8 rows and 8 columns. The columns are labeled 0 through 7, and the rows are labeled 0 through 7. A bracket above the columns is labeled "columns", and a bracket to the left of the rows is labeled "rows". Yellow arrows point from each row label to the corresponding row in the grid.

	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

declare:-

`datatype[][] arr_name = new datatype[rows][col];`

`int[][] arr = new int[n][n];`

no. of rows = `arr.length` ;
no. of col = `arr[0].length` ;

`for (int i = 0 ; i < arr.length ; i++) {`
 `for (int j = 0 ; j < arr[0].length ; j++) {`
 `System.out.println(arr[i][j]);`

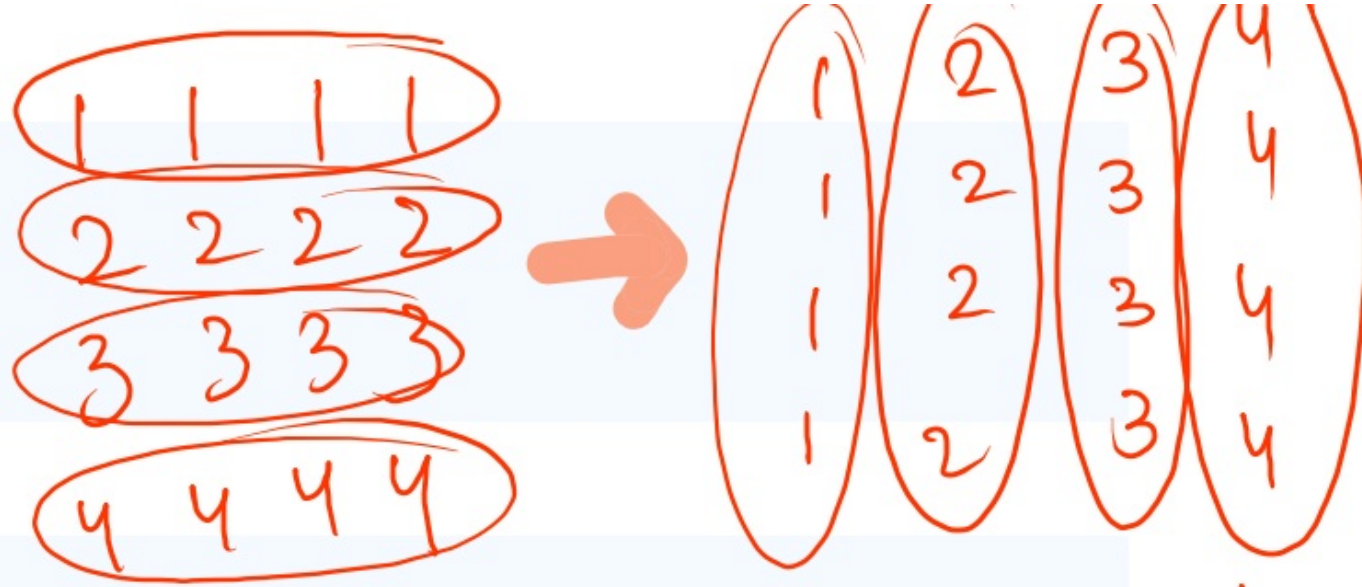
rows

col

}

}

Transpose of Matrix of $N \times N$

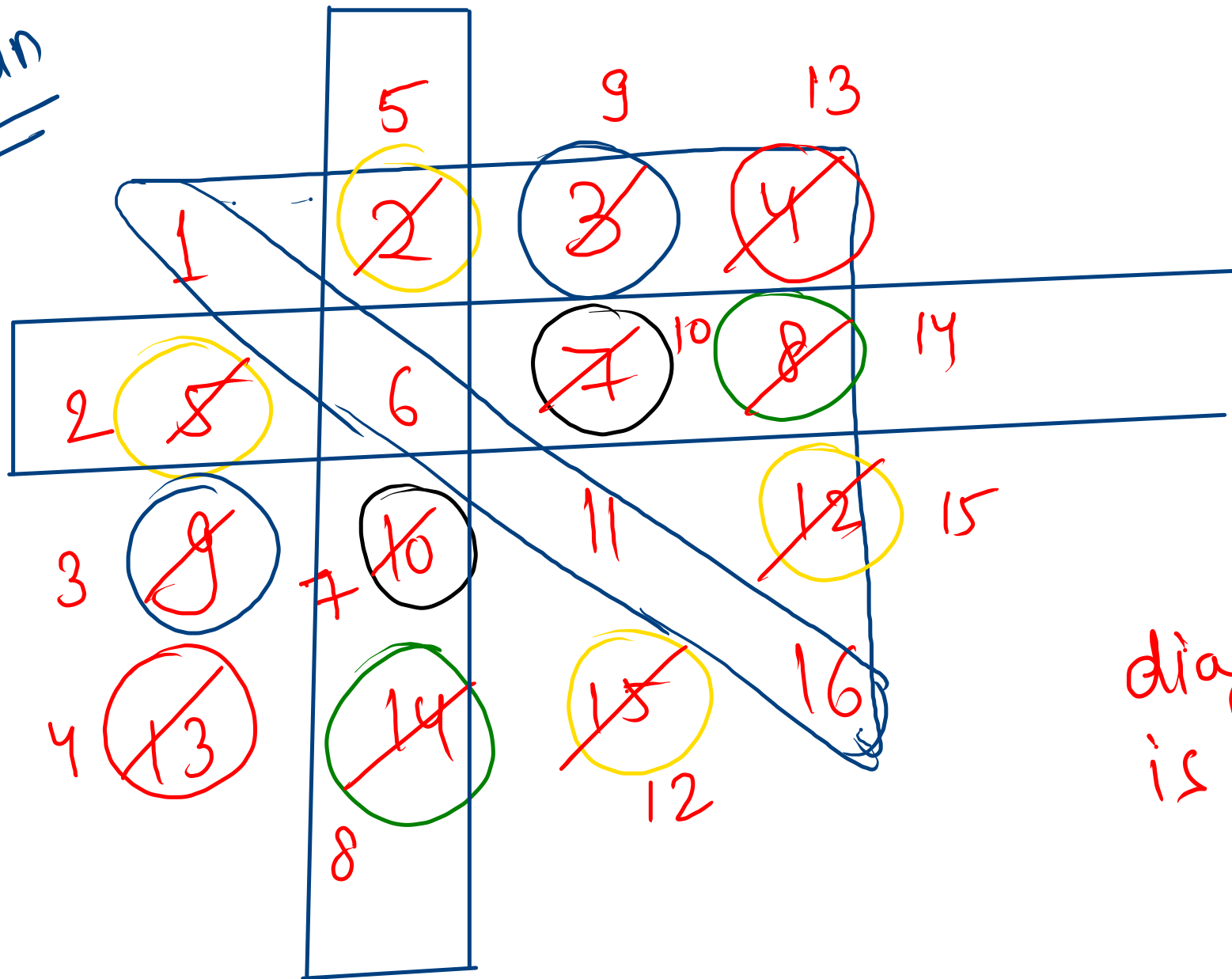


Convert all rows into column
or all columns into rows

	2	3	4	
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4

```
public static void transpose(int[][] arr, int n) {  
    for (int i = 0; i < n; i++) {  
        for (int j = 0; j < i; j++) {  
  
            int temp = arr[i][j];  
            arr[i][j] = arr[j][i];  
            arr[j][i] = temp;  
  
        }  
    }  
  
    for (int i = 0; i < n; i++) {  
        for (int j = 0; j < n; j++) {  
            System.out.print(arr[i][j] + " ");  
        }  
        System.out.println();  
    }  
}
```

dry run



diagonal part
is constant