Interview Problems

Starting 9:05

TABLE OF CONTENTS

1. Max consecutive is :
   - a. Atmost 1's replace
   - b. Atmost 1's swap
2. Majority Element
3. Row to Column Zero

Notes

Revision

1. 'A' → 65 ; 'a' → 97 , '0' - 48

2. Substring → a continuous part of a string.

3. "abc" → a, ab, abc , b, bc , c

$N\frac{(N+1)}{2} \rightarrow 3\frac{(3+1)}{2} = 6$

4. String Builder

5. "anamadamm"

< Question > : Given a binary array [ ]. We can almost replace a single 0 with 1. Find the maximum consecutive 1's we can get in the array[ ] after the replacement.

ex:- $\{1, 1, \overset{1}{\cancel{0}}, 1, 1, 0, 1, 1\}$

5. → Output

Q:- 1. $\{1, 1, 0, 1, 1, \overset{1}{\cancel{0}}, 1, 1, 1\}$

6.

Q:- 2. $\{0, 1, 1, 1, \overset{1}{\cancel{0}}, 1, 1, 0, 1, 1, 0\}$

~~4~~ 6

6.

Approach:- Iterate over the array. Whenever a '0' is found, cal. the consecutive ones on left (l) & right (r). Compare the global ans with (l+r+1). Update ans if needed.

$$\begin{array}{ccccccccccc} & & & & & & & j & & & \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \{0, & 1, & 1, & 1, & 0, & 1, & 1, & 0, & 1, & 1, & 0\} \\ & & & & & & & & & & i \end{array}$$

ans = ~~∅~~; ~~4~~ 6

r = 0

l = 2

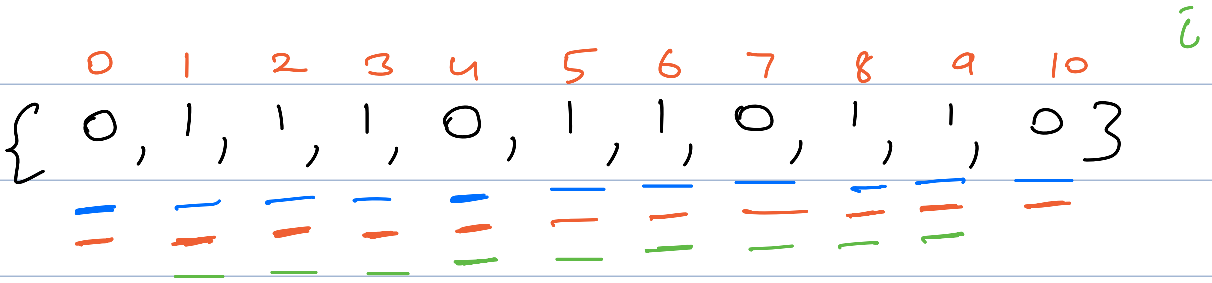
ans = max (ans, 0 + 2 + 1)

Code

```
int    solve ( int arr[], int N) {

    int  totalOnes = 0;
    for ( i=0; i< N; i++) {
    |     if (arr[i] == 1) { totalOnes ++; }
    }  if (totalOnes == N) {return N; }

    int  ans = 0;
    for (i=0 ; i<N; i++ ) {
    |        if (arr[i] == 0) {
    |        |      l=0; r=0;
    |        |      j = i+1;    # Consecutive 1's on right.
    |        |      while ( arr[j]==1 && j< N){
    |        |      |      j++; r++;
    |        |      }
    |        |      j = i-1;    # Consecutive 1's on left
    |        |      while (arr[j]==1 && j>=0){
    |        |      |   j--; l++;
    |        |      }
    |        |  ans = max(ans, l+r+1);
    }     }

    return ans;
}
```

Q:3

T.C → $O(N)$.



3 N

$$\rightarrow \{0, 0, \underbrace{1, 1}_{2}, 0, \underbrace{1, 1}_{2}, 0\}$$

Evry element is getting accesed at max 3 times.

< Question > : Given a binary array [ ]. We can swap a single 0 with 1. Find the maximum consecutive 1's we can get in the array[ ] after almost 1 swap.

ex:- $\{ \overset{0}{\cancel{1}}, 0, 1, 1, \overset{1}{\cancel{0}}, 1 \}$

Q:- 4. $\{ \overset{0}{\cancel{1}}, 1, \overset{1}{\cancel{0}}, 1, 1, 1 \}$

Approach:- Same as previous, just need to check for the case when the sequence you are trying to create have enough ones to be supplied.

```
int     solve ( int arr[], int N) {

    int   totalOnes = 0;
    for ( i=0; i< N; i++) {
        if (arr[i] == 1) { totalOnes ++; }
    }
    if (totalOnes == N) { return N; }

    int  ans = 0;
    for (i=0; i<N; i++) {
            if (arr[i] ==0) {
                l=0; r=0;
                j = i+1;   # Consecutive 1's on right.
                while ( arr[j]==1 && j< N){
                    j++; r++;
                }
                j = i-1;   # Consecutive 1's on left
                while (arr[j]==1 && j>=0){
                    j--; l++;
                }
            if ( l+r == totalOnes) {      totalones
                ans = max(ans, l+r);
            }
            else {
                ans = max(ans, l+r+1);
            }
    }  }  }
    return ans;
}
```

T.C → O(N)

Majority Element

< *Question* > : Given array [N]. Find the majority element

↓

Elements which occurs more than N/2 times.

- *You can assume that majority element always exists.*

$N = 3$

ex :- { 2, 1, 4 } No majority element exists.

$N = 8$

ex :- { 3, 4, 3, 2, 4, 4, 4, 4 }

Atleast 5.

ans = 4.

$N = 8$

ex :- { 3, 3, 4, 2, 4, 4, 2, 4 }

No majority element.

$N = 11$

Q:- 5. {3, 4, 3, 6, 1, 3, 2, 5, 3, 3, 3}

Atleast 6.

ans = 3.

{1, 2, 3, 3, 3, 3, 3, 3, 4, 5, 6}.

Q:- 7 I am always going to have a single majority element.

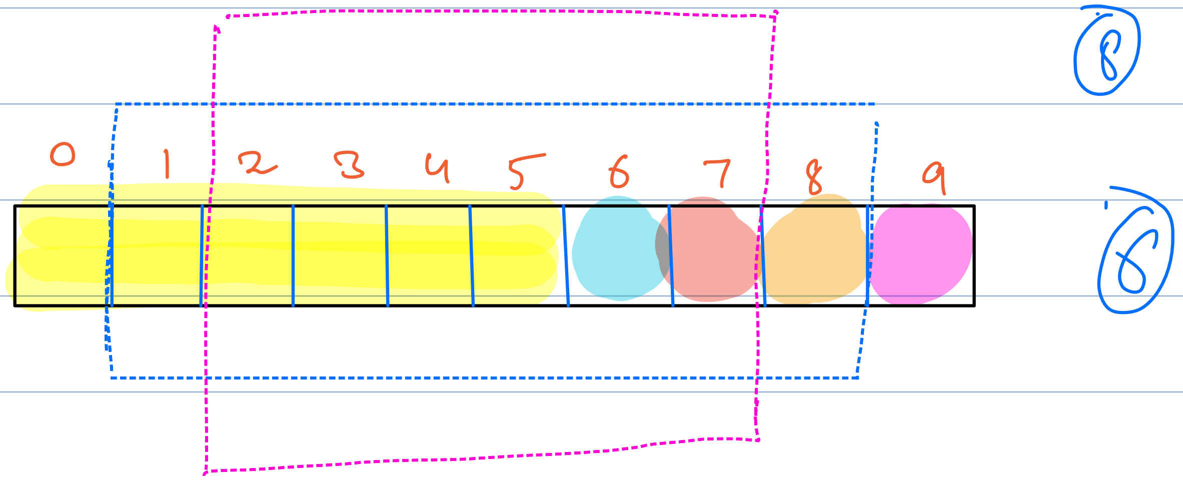
Brute Force :- ① Taking 2 loops. For each element, count it's frequency in another loop. T.C → O(N²).

② Sort the array. T.C → O(N log N).

Iterate over the array & keep count of consecutive N/2 + 1 elements.

Observations Moore's Voting Algorithm

1. There is going to be only 1 majority element.
2. Removing 2 distinct elements from my array has no impact on majority element.



Another Example :-

OP : 웃 웃 웃 웃 웃 웃 웃 웃 웃 = 9

yP : 웃 웃 웃 = 3

RP : 웃 웃 = 2

GP : 웃 웃 웃 = 3

Atleast 9.

Removing	Count Of:-				Winner
	Orange	Yellow	Red	Green	
	9	3	2	3	Orange
1 o & 1 R	8	3	1	3	Orange.
1 y & 1 G	8	2	1	2	Orange

Algo:-

0 1 2 3 4 5 6 7 8 9 10 i

ex:- {3, 4, 3, 6, 1, 3, 2, 5, 3, 3, 3}

majEleIndex = ~~0~~; ~~2~~ ~~4~~ ~~6~~ 8

count = ~~1~~; ~~0~~ ~~1~~ ~~0~~ ~~1~~ ~~0~~ ~~1~~ ~~0~~ ~~1~~ ~~2~~ 3

Important Point:- The above algo gives us the potential majority element. We need to verify once whether the element pointed by majEleIndex has a frequency > N/2 or not.

Code.

```
int  majEleIndex = 0;
int     cout = 1;

for( i=1 ; i< N; i++) {
        if (cout == 0){
            majEleIndex = i;

            cout = 1;
        }

    else {
      if (arr[i] == arr[majEleIndex]){
                cout++;
        } else {
            cout--;
        }
    }
}
```

majEleIndex is pointing at a potential majority Element.

So, verify it.

```
int    count = 0;
  for (i=0; i< N; i++) {
      if (arr[i] == arr[majEleIndex]) {
              count++;
      }
  }
  if (count > N/2) {
      return arr[majEleIndex];
  }
     else {
     return -1;   # No maj element exists.
     }
```

{3, 3, 3, 4}.

10:50

T.C → O(N)

S.C → O(1)

< Question > : Given array [N] [M].

Make all elements in a row and column zero if arr[i][j] $= 0$

Input

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 1 & 2 & \cancel{3}\ 0 & \cancel{4}\ 0 \\ 1 & \cancel{5}\ 0 & \cancel{6}\ 0 & \cancel{7}\ 0 & \cancel{0}\ 0 \\ 2 & \cancel{9}\ 0 & \cancel{2}\ 0 & \cancel{0}\ 0 & \cancel{4}\ 0 \end{array} \longrightarrow \begin{bmatrix} 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Output

mat[3][4]

rows

0	1	2
0	1	1

cols

0	0	1	1
0	1	2	3

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 0 \\ 9 & 2 & 13 & 4 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 \\ 9 & 2 & 13 & 0 \end{bmatrix}$$

Approach:-

Take two arrays, 1 of size rows & other of size column.
Initial both these arrays with 0.

Iterate on every cell of the matrix.
if arr[i][j] == 0;
update row[i] = 1 & col[j] = 1;

Iterate again over the entire matrix. For every cell (i,j) if either row[i] is 1 or col[j] is 1, mark that cell to 0.

Todo → Code.

T.C → $O(N*m)$
S.C → $O(N+m)$

90 + PSP

4 Q:-

1 → Assignment.
2 Q → Easy / Medium.
1 Q → Medium / Hard.

Passing Criteria → 75% marks.