DATE

LEETCODE | 6051

Can Place Flowers

Input : flowerbed = [1,0,0,0,1]

n = 1

Output : true

Step1 Find cases below (flowers can be placed in alternate positions where flowerbed[i]=0

Case 1 flowerbed with all zeros

$[\underline{0}, \overset{\times}{0}, \underline{0}, \overset{\times}{0}, \underline{0}, \overset{\times}{0}, \underline{0}]$ ← 7 nos → 4(max)

$[\underline{0}, \overset{\times}{0}, \underline{0}, \overset{\times}{0}, \underline{0}, \overset{\times}{0}]$ ← 6 nos → 3(max)

max = (length of flowerbed + 1)/2

Case 2 Leading or Trailing zeros

$[\underline{\overset{\downarrow}{0}\ 0\ \overset{\downarrow}{0}\ 0}\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ \underline{0\ \overset{\downarrow}{0}\ 0\ \overset{\downarrow}{0}\ 0}]$

4 nos → 2

5 nos → 2

max = $\dfrac{\text{count of leading / trailing zeros}}{2}$

Case 3 Middle zeros

$[0\ 0\ 0\ 0\ 1\ \underline{0\ \overset{\downarrow}{0}\ 0\ \overset{\downarrow}{0}\ 0}\ 1\ 0\ 0\ 0\ 0\ 0]$

$[0\ 0\ 0\ 0\ 1\ \underline{0\ \underset{\uparrow}{0}\ \overset{\text{or}\downarrow}{0}\ 0}\ 1\ 0\ 0\ 0\ 0\ 0]$

5 nos → 2

4 nos → 1

max = $\dfrac{c-1}{2}$

Step2

```
int fi = -1   <- 1st 1 encountered
int li = -1   <- last 1 encountered
int max = 0

for (int i=0; i< flowerbed.length; i++){
    if (flowerbed[i]== 1){
        if (fi ==-1){
            fi = i; li = i;
        }
        else
        {
            li = i;
        }
    }
}
```

Step3 If flowerbed has all zeros then fi = -1 still

```
So; if(fi==-1)
      ~~return~~ max= (flowerbed.length +1)/2;
```

Step4

For leading & trailing zeros;

```
max += (fi/2) + (flowerbed.Length -1- li)/2;
```

Step5 For middle zeros

```
int count =0;
for ( int i= fi+1; i<li; i++){
    if (flowerbed[i]==0){
        count ++;
    }
    else{
        max += (count-1)/2;
        count =0;        //reset count for next
                           middle zeros after
                           1 is encountered
    }
}
```

DATE □□ □□ □□□□

Step6

Since for last 1 & middle zero's loop coould have exhausted

```
if (count >0) {

        max += (count+1) /2;
```

Step 7 Retlurn n <= max;