① Special Index:

Given an array of N elements.
Task is to find if it is possible to remove an element s.t the sum of even indexed and odd indexed array elements is equal.

$[2 \quad \cancel{1} \quad 6 \quad 4]$
$\quad\quad 0 \quad 1 \quad 2$

$\Rightarrow \quad 2 \quad 6 \quad 4$
$\quad\quad 0 \quad 1 \quad 2$

$1+4 = 5 \neq 6$

$\boxed{2+4 == 6}$ ✓

for i in range(n): $\rightarrow O(N)$

sum of even index == sum of odd index after removal $\rightarrow O(N)$

$\Rightarrow TC = O(N^2)$

$i=4 \Rightarrow$ remove

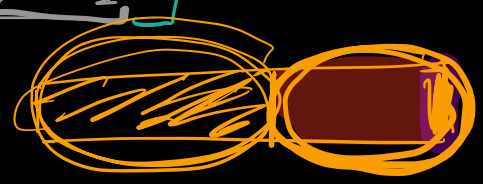
$a_0 \quad a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6 \quad a_7 \quad a_8$

$\hookrightarrow [a_0 \; a_1 \; a_2 \; a_3 \; a_5 \; a_6 \; a_7 \; a_8]$

$S_e = a_0 + a_2 + a_5 + a_7$

$S_o = a_1 + a_3 + a_6 + a_8$ ✓

a_4 Removal



a_5 Removal

$S_e = a_0 + a_2 + a_4 + a_7$

$S_o = a_1 + a_3 + a_6 + a_8$

$\Rightarrow$ Before $A_5 \rightarrow$ Sum of Even Indexed

After A_5 Sum of Odd Indexed

$S_e = E_sum[5-1] + O_sum[n-1] - O_sum[5]$

Odd, Even.

If we are removing i^{th} element

$a_0 + a_2 + a_4$ $\rightarrow$ Generalise formula.

$[\cancel{a_1} + \cancel{a_3} + \cancel{a_5} + a_7] - \cancel{a_1} - \cancel{a_3} - \cancel{a_5}$

Even Indexed Prefix Sum $=$ E_Sum[i] $=$ Sum of even indexed elements till i

Odd Indexed Prefix Sum

$\Rightarrow$ E_sum[i] $= a_0 + a_2 + a_4 \ldots a_i$

if i even

else, $i-1$

O_sum[i] $= a_1 + a_3 + a_5 \ldots a_i$

...

Implementation:

index index (0 index)

Build E_sum and O_sum

```
E_sum = [0] * n        O_sum = [0] * n

E_sum[0] = A[0]
for i in range(1, n):
    if i % 2 == 0:
        E_sum[i] += A[i]
    else:
        O_sum[i] += A[i]

    O_sum[i] += O_sum[i-1]
    E_sum[i] += E_sum[i-1]
```

1

	0	1	2	3
	2	1	6	4
E:	2	2	8	8
O:	0	1	1	5

$E[0] = a_0$

$E[1] = a_0$

$E[2] = a_0 + a_2$

```
if O[n-1] == E[n-1] - E[0]:
    return True
```

→ Boundary case 3 To remove a_0

2

```
for i in range(1, n):
    SE = E[i-1] + O[n-1] - O[i]
    SO = O[i-1] + E[n-1] - E[i]
    if SE == SO
        return True
return False
```

Try to remove ith element:

O(1) → space HW

Longest Consecutive sequence

Q Given an unsorted array of N integers.
Find the length of longest set of consecutive elements from the Array

[100, 4, 200, 1, 3, 2]

→ [1 2 3 4 4 100 200]

1 2 3 4 4 1 1

① Sorting → sort the array
→ Find max consecutive count

TC = O(NlogN)

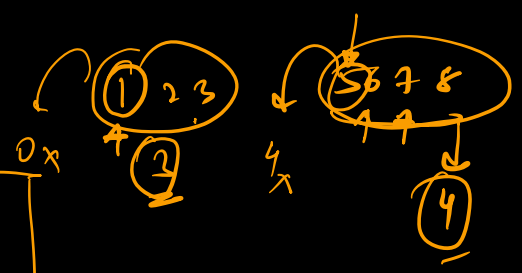
A

① → A.sort() ⇒ O(NlogN) [a_0]

```
ans = 1
cnt = 1
for i in range(1, len(A)):
    if a[i] == a[i-1]:
        continue
    if a[i] == a[i-1]+1:
        cnt += 1
        ans = max(ans, cnt)
```

```
else:
    cnt = 1
```

TC → O(N)

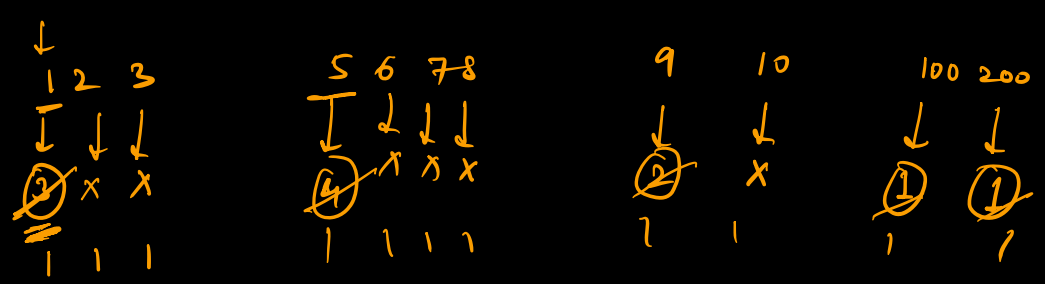



```
ans = 0
cnt = 0
s = set(A)

for x in A:
    if x-1 not in s:
        cnt = 0
        val = x
        while val in s:
            val += 1
            cnt += 1
        ans = max(ans, cnt)
```

O(1)

O(N)



1 2 3 5 6 7 8 O(N)

9 → 1 2 3 4 5 6 7 8 9

9 O(9) O(1) O(1)

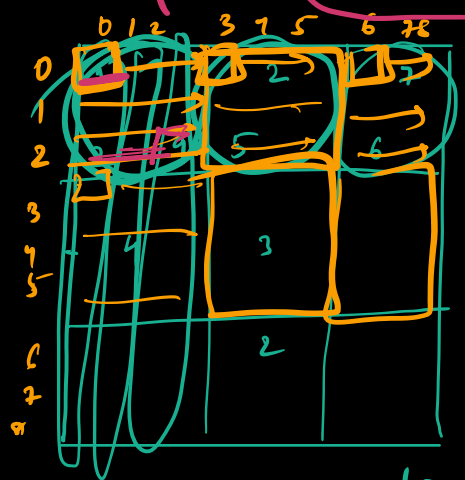
9 + 9 = 18 = 2N

O(N)

dic[i][j] = set( )

Valid Sudoko

0,0 → 0 0
0 1 0 0
0 2 0 0
2 2 0 0



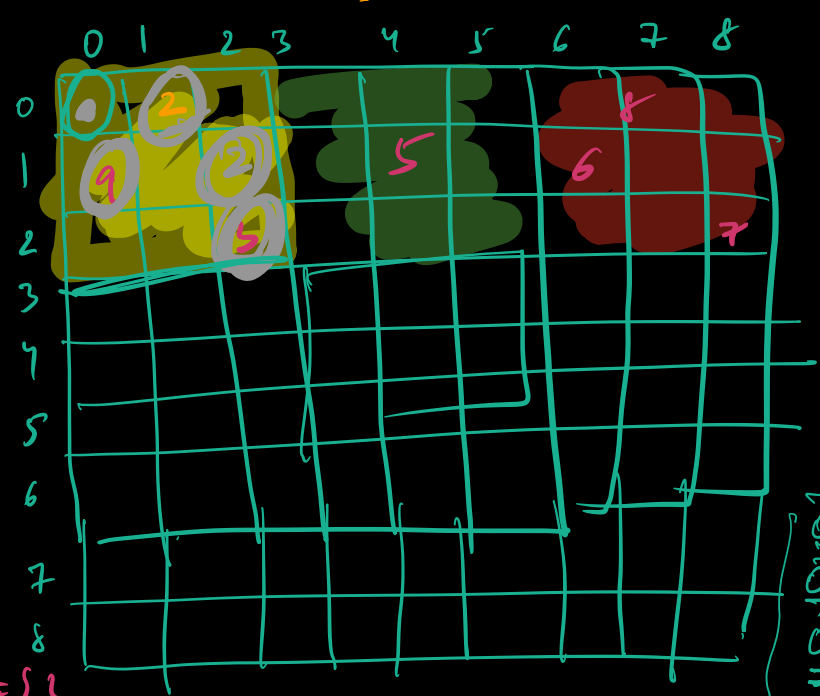
Constant ∀ elements in a box

1. Every Row should have No repeating elements (1 - 9)
2. Every Column " " " "
3. Every small box should also not have a repeating element.

row

```
for row in mat:
    s = set()
    for cell in row:
        if cell == '.':
            continue.
        if cell in s:
            return False.
        s.add(cell)
```

col



$i//3$

$j//3$

1 2

0 0

0,0 3,0 6,0

0,3 3,3 6,3

0,6 3,6 6,6

```
dic = {}
for i in range(9)
    if i//3 not in dic: dic[i//3] = {}
    for j in range(9)
        if j//3 not in dic[i//3]
            dic[i//3][j//3] = set()
```

```
if mat[i][j] == '.':
    continue
if mat[i][j] in dic[i//3][j//3]:
    return False
dic[i//3][j//3].add(mat[i][j])
```

{

0 : {0 : 2} {1 : set} {2 : set}

1 : {0 : set} {2 : set 2 : set

2 :

}

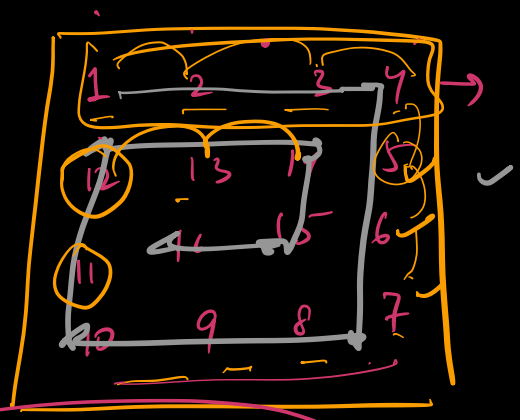
0,0	→ ( )	2,0	= ( )
0,1	= ( )	2,1	= ( )
0,2	( )	2,2	= ( )
1,0	( )		
1,1	( )		
1,2	( )		

dic → key → tuple

(0,0) → set( )

0,1

Spiral Matrix



L → R → 4
T → B → 3
R → L → 3
B → T → 2

L → R → 2
T → B → 1
R → L → 1
B → T → 0

Just inc y coordinate

Just inc x coord

[[0, 1] [1, 0] [0, -1] [-1, 0]]

L, R — T → B — R L — B T

0

4 → 3 → 2 → 1

steps = [A, A-1]

Horizo Vertical

directions = [[0, 1], [1, 0] [0, -1], [-1, 0]]

dir = 0 — 0 — 1 — 2 — 3

x = 0
y = -1

k = 1

steps left in this

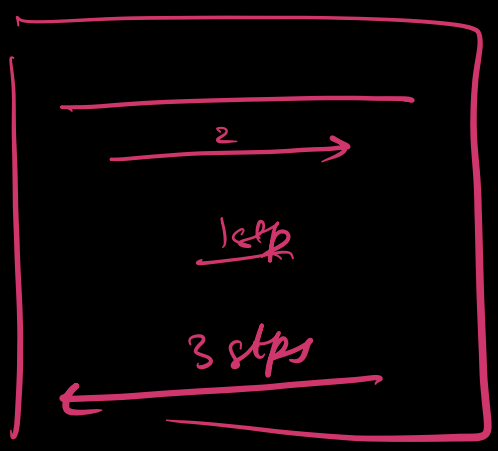
```
while steps[dir % 2] != 0:
    for i in range(steps[dir % 2]):
        x += directions[dir][0]
        y += directions[dir][1]
        mat[x][y] = k
        k += 1
    steps[dir % 2] -= 1
    dir = (dir + 1) % 4
```

motion

x = 0
y = 1
dir = 2
steps = [2, 2]
k = 2

	0	1	2	3
0	1	2	3	4
1	12			5
2	11			6
3	10	9	8	7

x = 3
y = 3



4 step x = 3
y = 2

1 move in the correct direction dir

for Required no. of steps

0
1
2
3

K → count

steps [_ _]

H V