Day 1 - Big O

20 June 2022 08:15

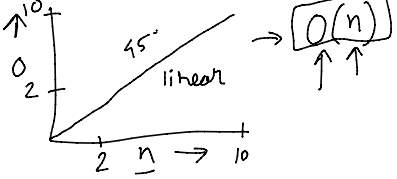
O(n), O(1), O(n²)

function( ), funtion( )

3 sec, 2 sec — 10 ms, 3 min

→ n ← size of input

n → 10



n

1 → 1000

1 → 100

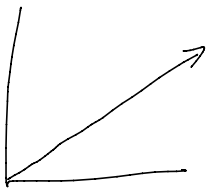
sum = 0;

for 1 → 100

sum += i

O(n)

linear



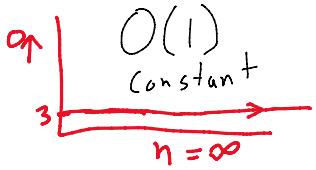
AP

n

1 → 100000000

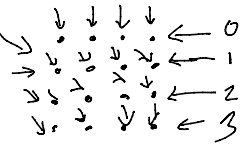
$Sum = \frac{100 * (100-1)}{2}$

$\frac{(n \times (n-1))}{2}$



O(1), O(n)

func(n)



for (0 → n) → row

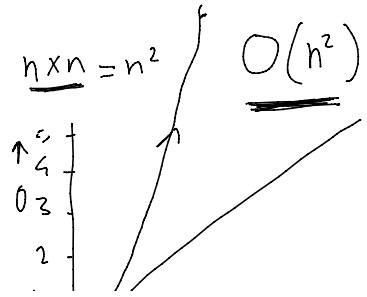
for (0 → n) → col

→ Print (*)

Print ('/n')

n × n = n²

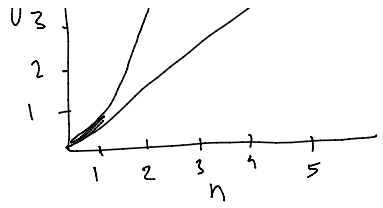
O(n²)



16 = 4 × 4

= n × n

= n²



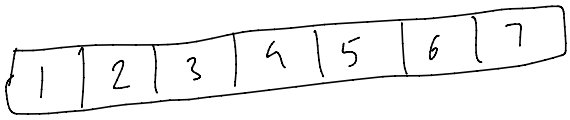
$= n^2$

25 1m

625 1mx1m

$O(n^2)$

n=7

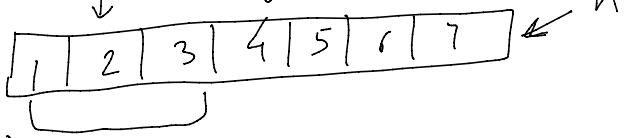


search(2) → 2

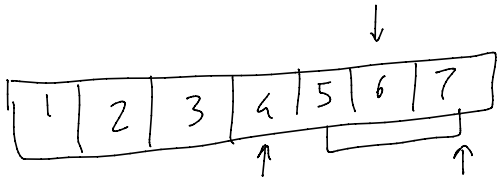
search(7) → 7 → O(n)

worse case

n=7



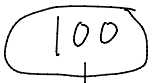
search(2) → 2



search(7) = 3

n/2

$(\log n)$



50 1

75 2

88 3

94 4

97 5

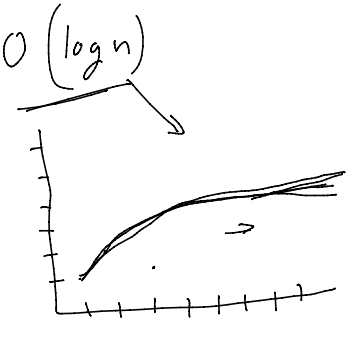
99 6

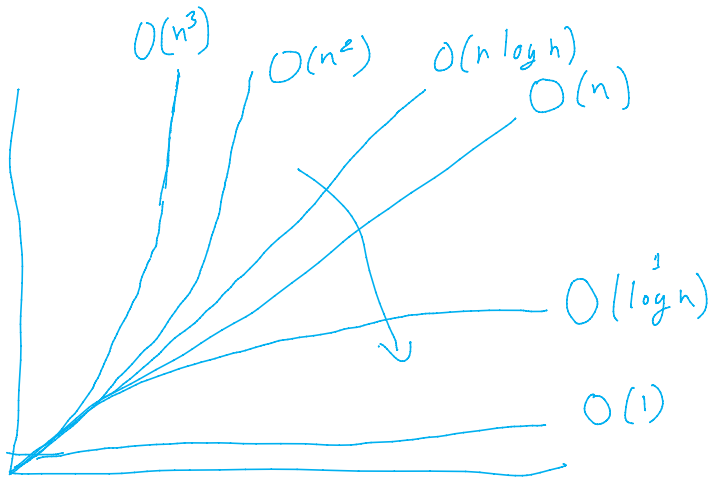
7

100 → 10

$\log(100) = 10$

$O(\log n)$





$$O(1) < O(\log n) < O(n) < O(n \log n) < O(n^2) < O(n^3) < O(2^n) < O(n^n)$$

good

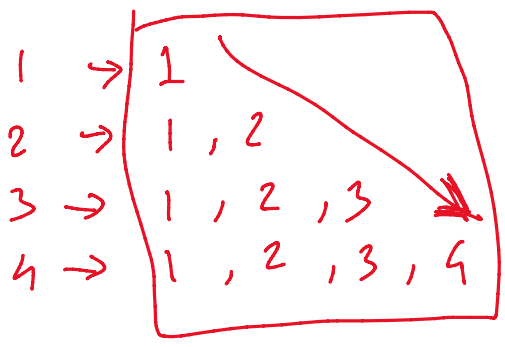
i = 1, 2, 3, 4

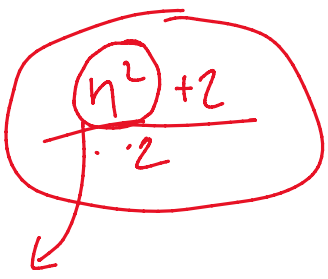
```
for 1 → 5 i
    for 1 → 5 j
        Print(j)
    Print('\n')
```

4 → 1, 2, 3, 4
1, 2, 3, 4
1, 2, 3 4
1, 2, 3 4

3 → 9
4 → 16
5 → 25
n → $O(n^2)$

```
for i in 0 → 4
    for j in 0 → i
        Print(j)
    Print('\n')
```





4 → [1, -1, 2, 1, 7]

4 → 10 time

$O(n^2)$

space → RAM consumed

A → 1 Kb, 2 Kb ← 2 byte

B → 5 KB, 10 KB ← 4 byte

O( )

func(n) → 0 → n

10 → 1
1000 → 1
10000 → 1

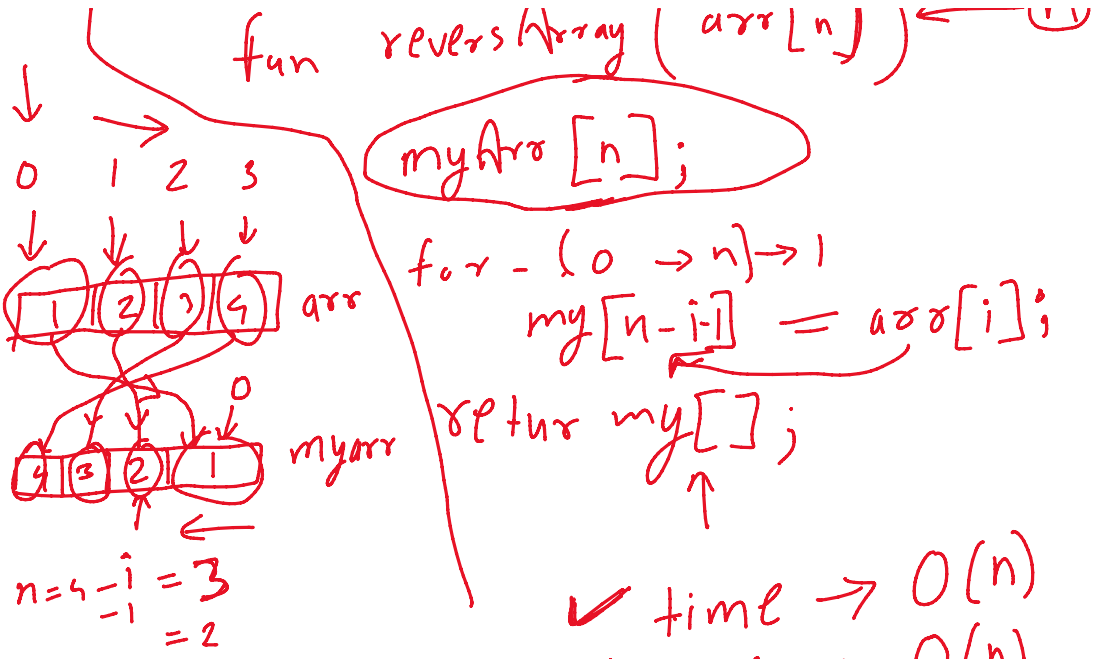
int sum = 100 → 1

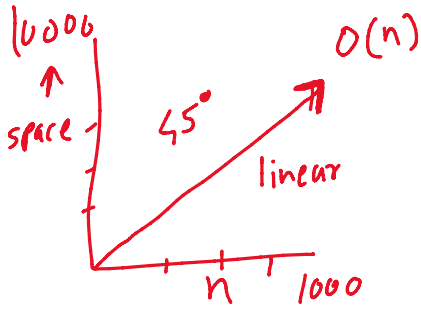
O(1) space
Constant

sum(a, b)
c = a+b
return

→ O(1) time
O(1) space

1. fun reversArray(arr[n]) ← n (100000)





✓ time → O(n)

✓ space → O(n)

space → no. of mem element

time → no. of operations