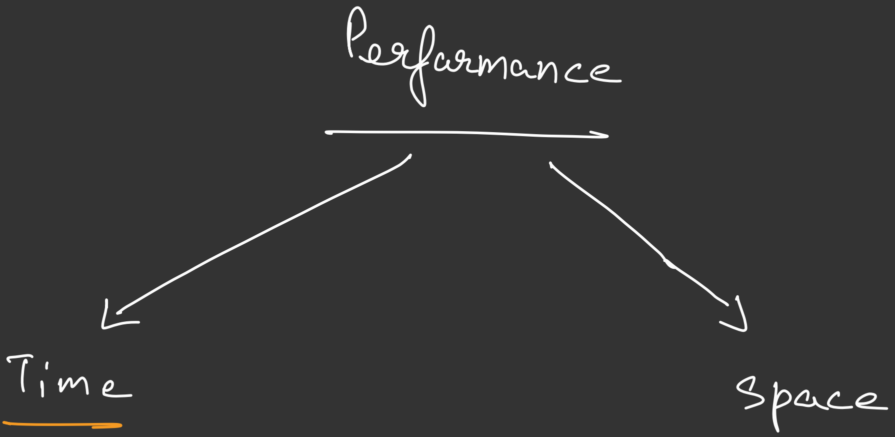
Time Complexity


Performance

Time

Space

↳ No of operations

↳ each operation is going to take a unit time.

Syso("Hello World") → 1 Operation

→ 1 unit of time

Syso(1) — Unit

syso(2) —

⋮

syso(5) —

5 Unit

```
for(int i=0; i< 100000; i++)
{
    Syso ("Hi"); // 1 operation
                      ↳ 10^5 times
}
                      10^5 Operations
```

1 → (n) → Input

size of input (n) and no of operations

↳ linear (n^1) → 10

$n \rightarrow 10$

$n \Rightarrow 10$ ↳ Quadratic (n^2) ⇒ 100

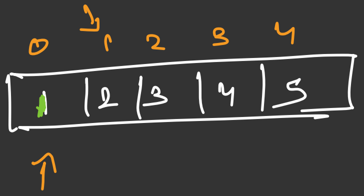
$n = 10$ ↳ Cubic (n^3) ⇒ 1000

$n = 10^5$; $(10^5)^3 \Rightarrow 10^{15}$

```
for(int i=0; i< arr.length; i++) → n
{ if (arr[i] == val) { // 1 operation
        return 1;
    }
}
```

Linear Search

n=5



$\boxed{1}$

Best case $\Omega(1)$

no of operations $= \boxed{1}$

Avg Case $\Theta(n)$

$$\frac{1+2+3+4+5+\dots n}{n} = \frac{\cancel{n}\times(n+1)}{2\cancel{n}}$$

avg $\Rightarrow \frac{(n+1)}{2}$

Worst case $O(n)$

$\boxed{n}$ comparision

max time / max no of operations your algo will perform

```
main() {
    Scanner scn = new Scanner();   // 1
    int n = scn.nextInt();          // 1
    for(int i=0; i<n; i++) {
        syso('1');   // 1 unit
    }
```

$\boxed{2}$

$2 \sim O(k)$

1 unit × n

⇒ n unit

⇒ $O(n)$

$n+2$

$O(n)$

n = 1000.0000

① $f(n) = n + \boxed{K}$ → constant

// $O(n)$

② $f(n) = 4n + 3 \Rightarrow O(4n) \Rightarrow O(n)$

③ $f(n) = 8n^2 + \boxed{4n + 4} \Rightarrow O(n^2)$

// Remove non significant and constant

② nested loop n => Input

```
for(int i=0; i<n; i++)
{
    for(int j=0; j<n; j++)
    {
        syso("Hi");
    }
}
```

i	j	No of operations
~~0~~	~~0~~,1,2,....n	$\underbrace{1\ 2\ 3}$ → n -
1	0—n	n:
2	0—n	n:
.		
.		

⋮		⋮
n	0 – n	n]

Total no = $\underbrace{n + n + n \cdots\cdots n}_{n\,times}$

$= n(1+1+1+1+1+1+\cdots\cdots 1)$

$= n \times n = n^2$

Time ⇒ $O(n^2)$

③ Nested Loop

```
for ( i → 1 – n) {
        for ( j → 1 – m {
                syso ("Hi");
}    }
```

i	j	
0	0 – m	m
1	0 – m	m
⋮		⋮
n	0 – m	m

$m + m + \cdots m_n$

⇒ $m \times n$

$O(n \times m)$ ⇒ $O(nm)$

④

```
for ( i → 1 – n) {
        for ( j = 0;  j < i ; j++) {
                syso ("Hi");
}    }
```

i =	j	no. iterations
0	0 – 0	0
1	0 – 1	1
2	0 – 2	2
3	0 – 3	3
⋮ n	0 – n	n

⇒

$0 + 1 + 2 + \cdots\cdots n$

⇒ $\frac{n \times (n+1)}{2}$

⇒ $\frac{n^2}{2} + \frac{n}{2}$

⇒ $O(n^2)$

(5)

```
for(i => 1-n)
{
    a
}
```

2n

$\Rightarrow O(n)$

```
for(i => 1-n)
{
}
```

$O(n)$

Time complexity $\Rightarrow O(n + n)$

$\Rightarrow O(2n)$

$\Rightarrow O(n)$

```
for(i => 1-n) {
}
```

$O(n)$

```
for(i => 1-m) {
}
```

$O(m)$

$\Rightarrow \underline{O(n+m)}$

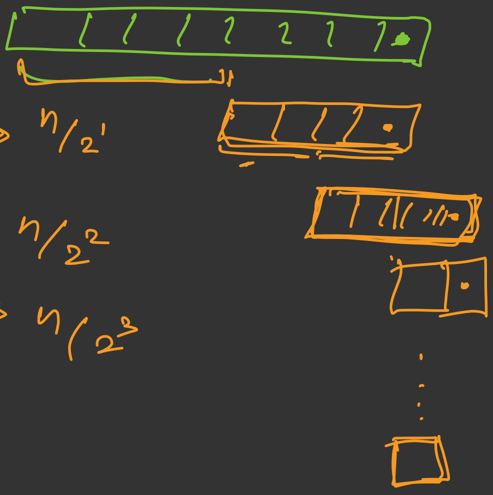
$n >> m$

10^5 2

$\hookrightarrow \underline{O(n)}$

Linear Search ⇒ $O(n)$

Binary Search



. .	—	n	
1	—	$n/2$]	⇒ $n/2^1$
2	—	$n/4$	⇒ $n/2^2$
3	—	$n/8$	⇒ $n/2^3$
k		$n/2^k$	

$$\frac{n}{2^k} = 1$$

$$n = 2^k$$

$$\log_2 n = \log_2 2^{\boxed{k}}$$

$$\log_2(n) = k \boxed{\log_2 2}$$

$$k \Rightarrow \underline{\log_2(n)}$$

no of operations

$O(\log(n))$

n	n^2	n^3	n^4 ...
1	1	1	1
2	4	8	16
3	9	27	81
10^5	10^{10}	10^{15}	10^{20}

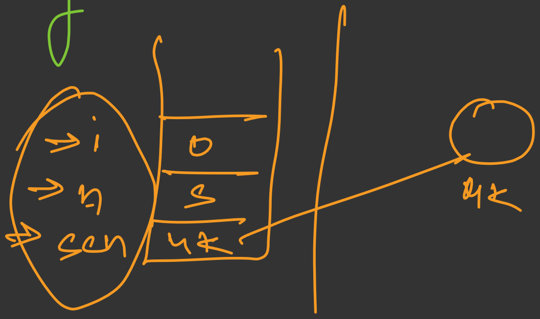
$O(1) < O(\log(n)) < O(n) < O(n\log n)$

$< O(n^2) < O(n!)$

$< 2^n$

Space Complexity

```
Scanner scn = new Scanner()
  int n = scn.nextInt()
for (int i = 0, i < n; i++)
      syso(Hi)
```



$O(3) \Rightarrow O(1)$

$n = 10 \Rightarrow 10^7$

Space

Space Complexity *

→ Any space taken by your program to execute

Auxilary Space

space taken by you algo apart from input

→ Input

O(n)

eg ⇒ Linear search

arr

i (1)

n (1)

arr (n)

$O(n + 1 + 1)$

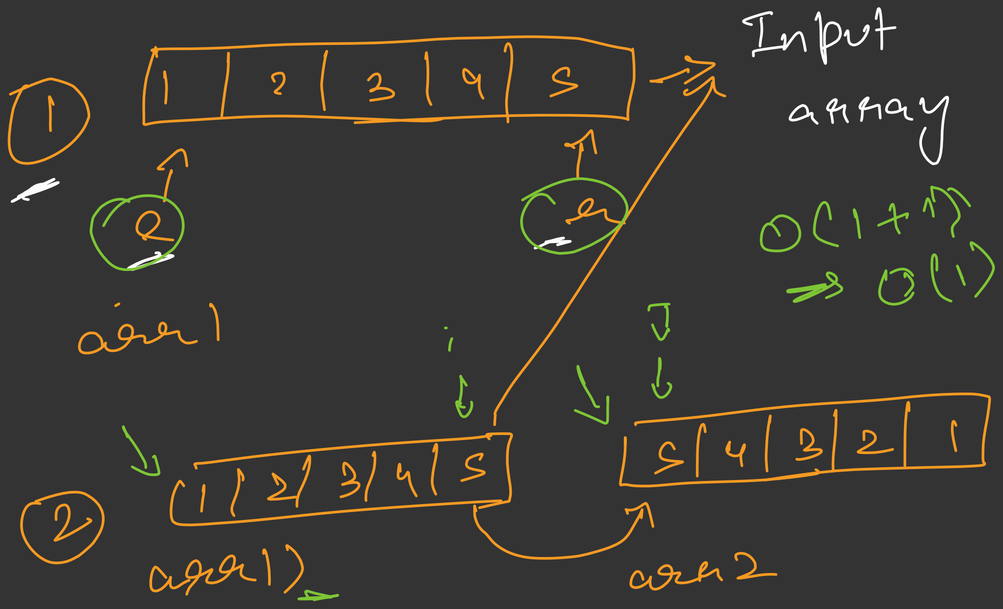
$\Rightarrow O(n)$

Auxilary space

+

Input space

Reverse array



	Space complexity	Auxilary space
1	$O(n)$	$O(1)$
2	$O(n+n+1+1)$ $\Rightarrow O(n)$	$O(n)$

Space - time trade off]

	Time	Space
①	$O(n)$	$O(n)$
②	$O(n^2)$	$O(1)$