0 → Code → ✓

[C language → Compiler → Machine code]

C

[Machin Code → CPU → result]

[Binary language = Machine Code]

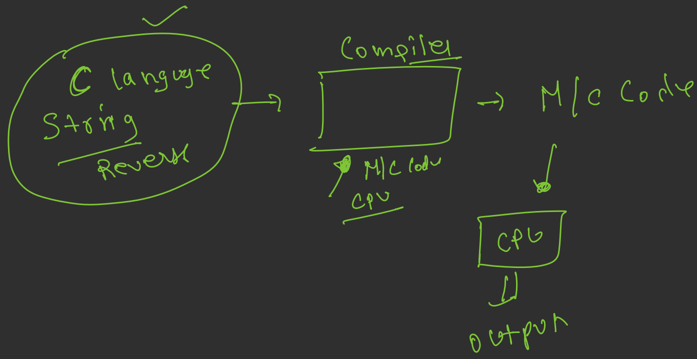
Compiler ⇒ Machine Code (0101010)

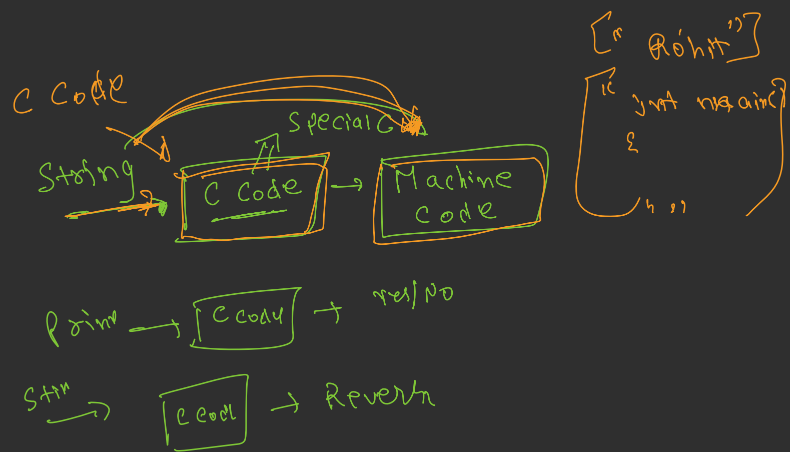
CPU

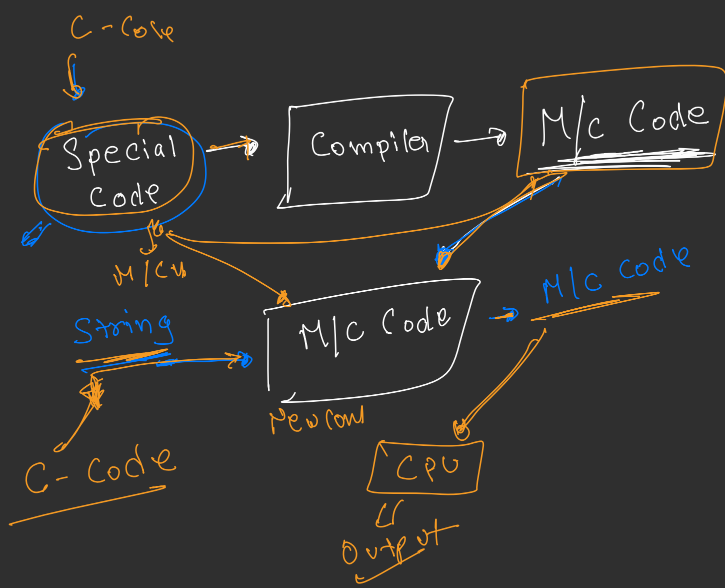
M/C → Binary code

Prime C → Compile → M/c Code

M/c → CPU → Output


C language
String
Reverse
Compiler
M/c Code
M/c code
CPU
CPU
Output


C Code
String
Special C
C Code
Machine code
["Rohit"]
int main()
{
}
Prime
C code
Yes/No
Str
C code
Reverse


C - Code
Special Code
Compiler
M/c Code
M/c
String
M/c Code
M/c code
New code
C- Code
CPU
Output

Compiler

input → C - Code → Binary (Machine) code

~~String~~

"Rohit" → Bina

int main ()

{

}

"Rohit is a good boy"

C languge

C - Code → Compiler → Machie code

M/C Code

C prime → M/ch Code

01010

CPU.

langu

CPU

C lan (M/c Code)

Clanguage

[input Code → compiler → M/c → CPU → Output]

0110 M/c Code

[① Machine Code
② Binary language]

Compiler

Code

C-Code → Compiler → M/c (New compiler) Code

Pur an Compiler

Machine

M/c - Code

CPU →

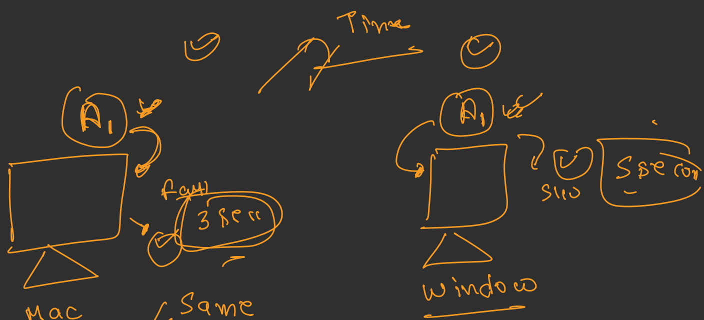
[Time Complexity]

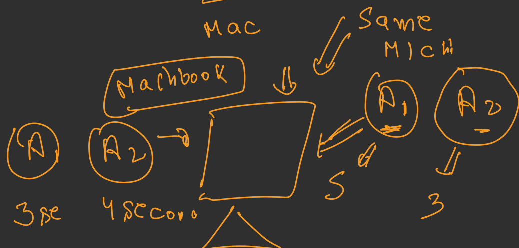
Problem

A1 A2 A3 A4

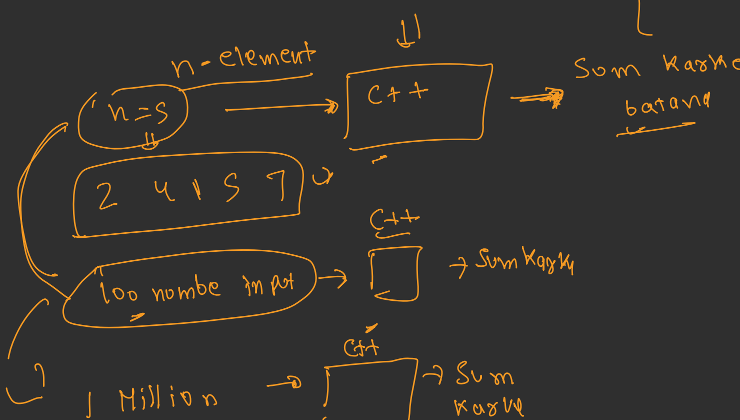
5 3 2 2

X






```
int arr[n]
int sum=0
for ( i=0; i<n ;i++)
        sum += arr[i];
```



// sum of n-number

int arr[n] = {10, 20, 30, 50}

int sum = 0;

for (int i = 0; i < n; i++)
{ sum += arr[i]
}

Cout << sum

n time

$n = 4$

Operation:

$i = 3$ $1+1+1+1 \Rightarrow 4 \Rightarrow 3+1$

$1+1+1 = 3$

$1+1+1 = 3$

$1+1+1 = 3$

$1 = 1$

19 oper

Total operation

$\Rightarrow \boxed{14}$ operation

$1+1+1+1 = 4$

$1+1+1 = 3$

$1+1+1 = 3$

$1+1+1 = 3$

$1+1+1 = 3$

$1 = 1$

17 oper

$\boxed{1+1+1}$

Input → Time Complexity → Output

$n^5 = 5$ second

$n = 100 = 100$ se

$n = 1M = 1$ Million second

Time Complexity ⇒ $\boxed{3n+4}$

$n \rightarrow$

$4 \Rightarrow$

$n \Rightarrow 16$ operation

$5 = n = 19$ operation

$6 = n = 22$ operation

$7 = n = 25$ operation

Number of operation

```
for ( i=0 ; i<n ; i=i+2 )
{
    cout << "Hello"
}
```

Time Complexity

$n/2 \rightarrow n$

Better

$\boxed{n}$

$\boxed{n^2}$

Time Complexity = $3n + 4$ — number 34, Better

Time complexity = $2n^2 + 6$ — $\boxed{206}$

```
for ( i=0 ; i<n ; i++ )
{
    ...
}
```

n bol sakta

```
for( i=0 ; i<n ; i=i+3)
{   cout<<" Hello"
}
```

No' of opera
⇓
$n/3$

$\Rightarrow \underline{n}$

T. Complexity ⇓

```
for (i=0 ; i<n ; i++)
{  cout <<"Hello";
}

for ( j=0 ; j<n ; j=j+5)
{  cout<<"No";
}
```

$\Rightarrow \underline{n}$

$\Rightarrow \underline{n/5}$

$n + \frac{n}{5} \Rightarrow \frac{6n}{5} \Rightarrow \boxed{n}$

```
Int arr[100] = {
```

```
for (int i = 0 ; i < 10 ; i++)
{   cout << "Hello"
}
```

10 → 1

1 →

n = 10 ⇒ Sum of first n natural

①

```
int sum = 0
for (i=1 ; i <= n ; i++)
{   sum += i ; }

cout << sum
```

n Time complexity

②

```
sum = n * (n+1)/2
cout << sum ;
```

1

```
for (i=0; i<n; i++)                 n
{
    for (j=0; j<n; j++)
    {
        cout<<"Hello"
    }
}
```

$i=0$ | $j = 0$ to $n-1$ | n time

$i=1$ | $j = 0$ to $n-1$ | n time

$i=2$ | j | n time

$i = n-1$

$j \Rightarrow$ n time

n

$n-1$

$\{0 \quad 1 \quad 2$

$n + n + n + \ldots n$

$\Rightarrow n^2$

```
for (i=0; i<3; i++)
{   for (j=0; j<3; j++)
    {  cout<<"Hello"
    }
}
```

$\boxed{9 \text{ time}} = n^2$

n

for(int j=0; j<n; j++)
{
cout << "Hello")
}

n

for(int j=0; j<n; j++)
{
cout << "Hello")
}

n^2

$5N^3 + 2N^2 + 3N + \cancel{20}$

$N^3 + N^2 + N$

N^3

$4N^3 + 6N^2 + 8N + 20$

$n = 5$ $\boxed{\log_2 n}$ ✓

$i = 0 * 2$

$i = 0$

$i = 0 * 2$

$i = 0$

$i = 1 \times 2$

$i = 4$

$K+1$ = Total

```
for (int i = 1 ; i < n ; i = i * 2)
{
    cout << "Hello";
```

$K+1$

2^0	2^1	2^2	2^3		2^K
$i = 1$	$i = 2$	$i = 4$	$i = 8$	....	$i = n$
1	1	1	1		1

Total operat

$\underbrace{[1 + 1 + 1 \ldots\ldots 1]}$ = answer

$2^K = n$

$\log 2^K = \log n$

$K \log 2 = \log n$

$K = \dfrac{\log n}{\log 2}$

$\boxed{K = \log_2 n}$ ✓

T.Complexit $K+1$

$1 \; \log_2 n + 1$

$\boxed{K = \boxed{\log_2 n}}$ ✓

$2^K = n$

$\log 2^K = \log n$

$K \log 2 = \log n$

$K = \dfrac{\log n}{\log 2}$

$\boxed{K = \log_2 n}$

```
for (i = 0 ; i < n ; i++)
{ cout << "Hello"
```

$\underbrace{1 + 1 + 1 + \ldots\ldots 1}$ $= \boxed{n}$ ✓

```
for(int i=1 ; i<n ; i=i*3)
{   cout << "Hello"
}
```

$\boxed{\log_3 n}$

$a * b = c$

$\left[\left[a = \frac{c}{b} \right] * \text{galat statement} \right]$

$\frac{a * \cancel{b}}{\cancel{b}} = \frac{c}{b}$

$\left\{ a = \frac{c}{b} \right\}$

$\log_3 n$

(1)

```
int i = n;
while(i != 0)
{   cout << "Hello"
    i = i/3;
}
```

$\boxed{i = i \times 2}$

1 2 4 8 16 . . . . . . n

$\log_2 n$

0 1 . . . . $n/2$ n

$\log_2 n$

```
for (i=0 ; i<n ; i++)       n
{   for (j=0 ; j<n ; j++)   n
    {   for (k=0 ; k<n ; k++)   = n
        {   cout << "Hello";
        }
    }
}
```

n^3

```
for (i=1 ; i<n ; i++)
{   for (j=0 ; j<=i*i ; j++)
    {   cout << "Hello";
    }
}
```

n $i=1$ $1 \text{ to } n-1$ $2 \times 2 = 4$

n^2 n^3

$i=1$	$i=2$	$i=3$	$i=4$	. . . . $i=n-1$
$j=0$ to 0	$j=0$ to 3	$j=0$ to 8	$j=0$ to 15	
1 task	4 task	9 task	16	$(n-1)^2$

$1^2 + 2^2 + 3^2 + 4^2 + \dots (n-1)^2$

$$\boxed{n(n+1)(2n+1)/6} \Rightarrow n^3$$

$$(n-1)(n)(2n-1)/6$$

$$\frac{\cancel{2}n^3 + \cancel{6}\cancel{n^2} + \cancel{5}\cancel{n} + \cancel{8}}{\cancel{6}}$$

for (i = 1 ; i < n-4 ; i++)

{

}

$\boxed{n}$ — $n-\cancel{4}$

n

$$\left[1^2 + 2^3 + 3^2 + \cdots (n-1)^2\right]$$

$$n-1$$

$$\left[n(n+1)(2n+1)/6\right] \checkmark$$

$$(n-1)(n)(2n-1)/6$$

$$\frac{\cancel{2}n^3 + \cancel{6n^2} + \cancel{5x} \cancel{-1}}{\cancel{6}}$$

$$n^3 \checkmark$$

```
for(i=1; i<=n; i++)
{ cout<<"Hello";
}
```

$i=1$ | $i=2$ | $i=3$ | . . | $i=n$

! | ! | ! | | !

$\boxed{n} * n = n^2$

```
for ( int i = 1 ; i<n ; i++ ) → n
{    for ( j=1 ; j<n ; j++ ) } n
     {
       cout<<"Hello"
     }
```

n^2

$i = 1$ | $i = 2$ | $i = 3$ | . . . . $i = n-1$

$j = 1 \text{ to } n-1$ | $j = 1 \text{ to } n-1$ | $j = 1 \text{ to } n-1$ | $j = 1 \text{ to } n-1$

$n-1$ | $n-1$ | $n-1$ | $n-1$

$(n-1) + (n-1) + (n-1) \ldots (n-1)$

$(n-1) * (n-1) = n^2$

Worst	Average	Best
n	n/2 = n	1

n

```
int arr[] = {10,20,17,9,8,11}
element = 90;
for (i=0 ; i<n ; i++ )
{   if (arr[i] == element)
    {  cout<< "yes"
       return 0;
    }
}
cout<< "No";
```

Big O
Omega
theta

⇒ overrated or raddi ⇒ senior engineer

Omega

$O(n^5)$

T. Com = n → Big O

⇓

upperbound

$O(n^3)$

$O(n^4)$

method

⇓

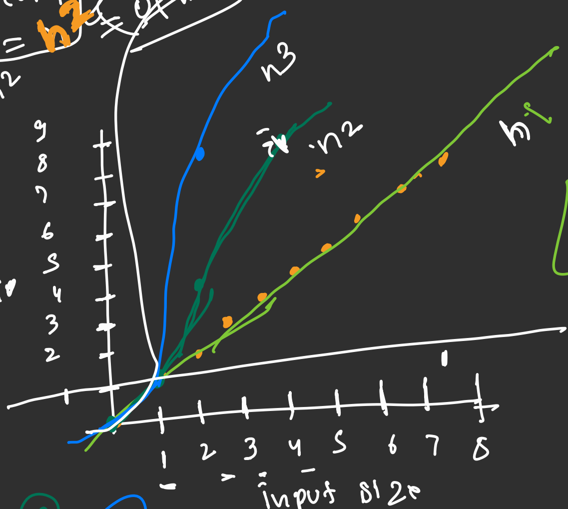
Time Comp = n2 = O(n)

no of operation

$\Omega(1)$

$\Omega(n)$

no of operation



// no of op.

```
for(i=0; i<n; i++)
{ cout<<"radhi"
}
```

$f(n) = n$

```
for(i=0; i<n2; i++)
{ cout<<"H
}
```

n=2

$f(n) = n^2$

n^3

$f(n) = 1$

Bigo → Upper Bound

⇓

Algorithm

⇓

{ equall, ya mersp neeche }

Method

Time Complexity = n^3

n^4, n^5, n^6

exact

theta

$O(\ )$ $\Theta(\ )$

theta

Θ = exact

n^2

$\Omega(n^2)$

$\Omega(n)$

$\Omega(1)$

n

$\Omega(1)$ ✓

true ✓

n^2

Time complexity → n^2 → after

Θ

$O(2^n)$

n^4

n^3

n^2

n

1

$\Omega(1)$

$+\ O(2^n)$ $\underline{99\%}$ time

$\Omega(1) = 100\%$ time

$O($

concentrate

O(n^4)

Big O

upar

Big O (n^4) =>

Omega

Θ

[Time complexity = | n^2] => Method:

Algorithm

n

log n

1

lower

bound

Ω(n) Ω(logn)

Ω(1)

Best = Ans

Avg = Ans

worst = Ans

Auxillary Space

Space Complexity:

$\boxed{O(1)}$ ✓

```
for(int i=0; i<n; i++)
{
  cout<<i;
}
```

reverse the array:

```
int arr[n] = {10, 20, 11, 12};
int newArr[n];
int j = n-1
for(int i=0; i<n; i++)
{
    newArr[j] = arr[i];
    j--;
}
return newArr;
```

input - O(n)
n - O(n)
O(n)

$O \Rightarrow$ worst

Θ = Average

Ω = Best case

Froc

T.Comp : $\boxed{n^2}$

O

Θ

Ω

n

worst avg best

$O(n^2)$

$\Theta(n^2)$

$\Omega(n^2)$

$O(n)$

$\Theta(n)$

$\Omega(n)$

Upper bound

$O(n^3)$ $O(n^5)$

$O(n^4)$

O = Upper bound

Θ

Ω = Lower bound

Time complexity — n^2

$\Omega(n)$

n

$\Omega(1)$

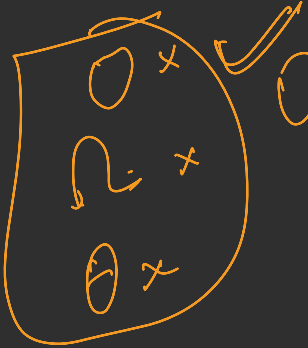
$\boxed{\text{Worst case}}$

Best | Average | Worst case

Intoruic w ∃

p knoowedge

$O(n^4)$



Dijksta $\rightleftarrows$ Professor (PhD)

Course - End

$\Downarrow$

Data Structure

[Sorting → ascending order]

[10, 1, 3, 9, 5] → 1 3 5 9 10

problem

[0-4]

[10, 1, 9, 3, 5] ascending Order

Selection sort (1-4)

1, 10, 9, 3, 5

2-4 1, 3, 9, 10, 5

1, 3, 5, 10, 9

1, 3, 5, 9, 10

worst case ⇒ n^2 ⇒ $O(n^2)$

→ $\Theta(n^2)$

→ $\Omega(n^2)$

Best case ⇒ -

int a=10 , b=20

int c = b

b = a

a = c

c = 20 swap

b = 10

a = 20

$\boxed{a = 20 \ , \ b = 10}$

void swap(&a, &b) {

}

[1, 2, 3, 4, 5]

Algorithm

Bubble sort

1st code

2	4	5	10	18

$j < n-1$

$j \quad 0 < (n-1)$

0 - 3 $\quad 0 - < n-2$

0 - 2 $\quad 0 - (n-3)$

0 - 1

0 - 0

0	1	2	3	4
2	.	5	10	18

5 10 4 2 18 ← 1 pass

5 4 2 10 18 ← 2 pass

4 2 5 10 18 ← $\boxed{\text{3 pass}}$

2 4 5 10 18

1	2	3	4	5

$$\frac{n+n^2}{2} = n^2$$

Average Case = $\boxed{n^2}$

Insertion Sort

ascending

0	1	2	3	4	5
20	18	4	6	7	2

4 18 20 6 7 2

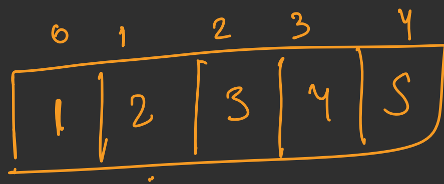
4 6 18 20 7 2

4 6 7 18 20 2

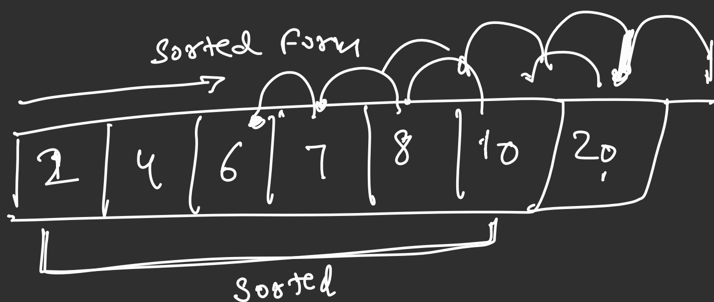
2	4	6	7	18	20

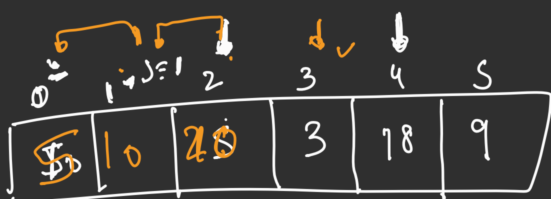
Chote element

3	4	5	6	7	15

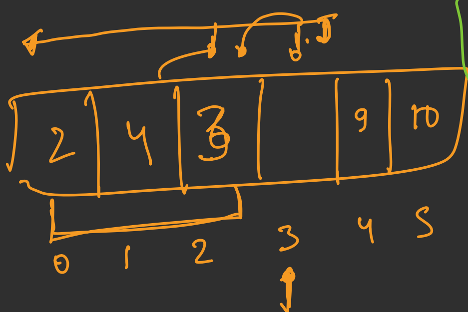


j--






```
j = 3 ; j > 0 ; j--
  if (arr[j] < arr[j-1])
  { swap (arr[j], arr[j-1])
  } else
      break;
```

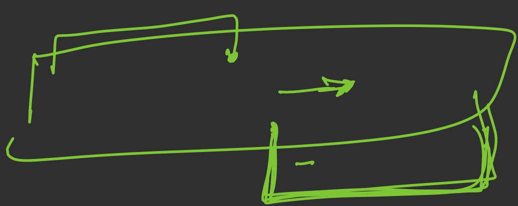


Multiple Practice

arr[j] < arr[j-1]

j = ~~3~~

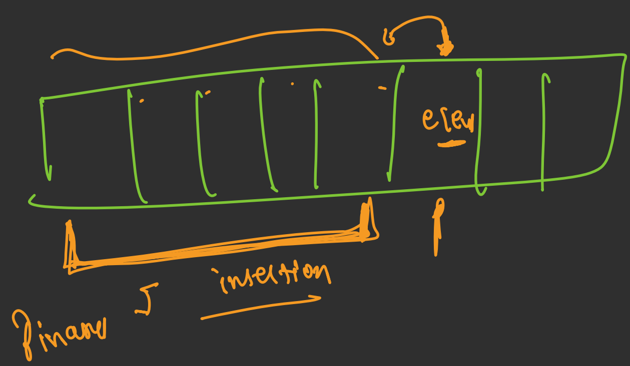
j = 2

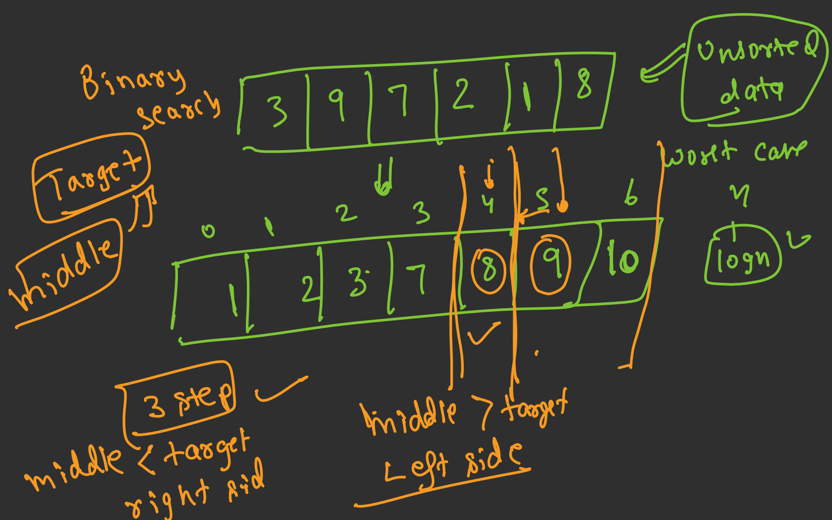


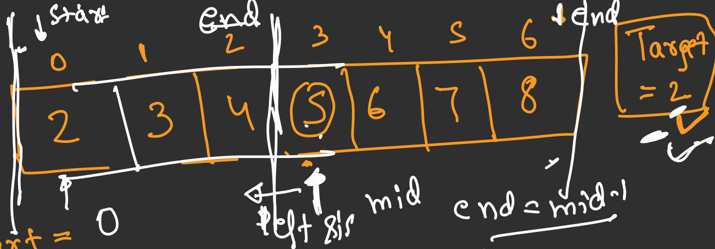
Just 4 sort 0

works

-ve last pos


elem
insertion
Binary


Binary search
3 9 7 2 1 8
Unsorted data
Target
middle
0 1 2 3 4 5 6
1 2 3 7 8 9 10
worst case
n
logn
3 step
middle < target
right sid
middle > target
Left side

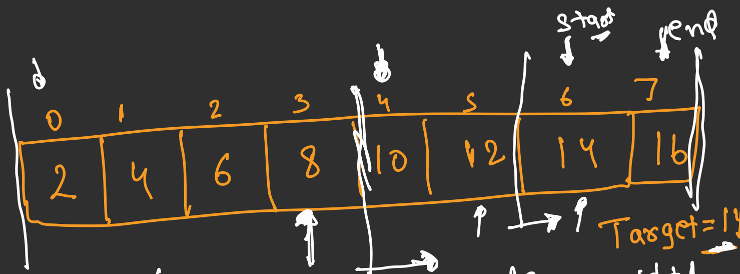


start = 0

end = 6

middle = $\frac{start+end}{2}$ → $\cancel{3}$

midd = 0



Start = ~~0~~ ~~4~~ 6

end = 7

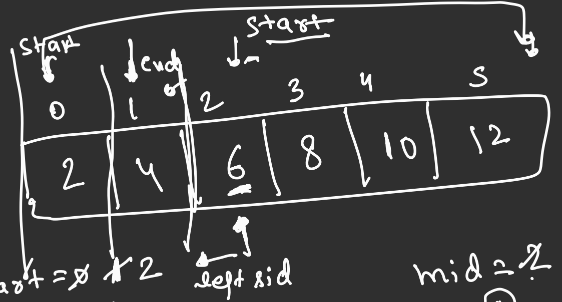
middle = $\frac{0+7}{2}$ → $\cancel{3}$ $\cancel{5}$ 6

if (arr [mid] < target)

start = mid+1

left side if (arr [mid] > target)

end = mid-1



target = 5

start = ~~0~~ ~~1~~ 2 left sid
end = ~~5~~ 1

mid = ~~2~~ 0

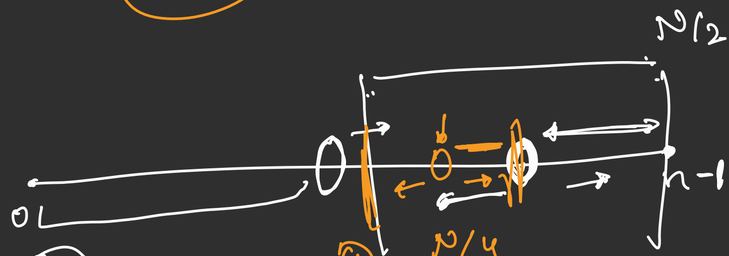
$\frac{1+1}{2} = \boxed{1}$

if (arr[mid] == target)
return mid;

else if (arr[mid] < target)
Start = mid+1

else {
end = mid-1
}

log₂n worst



N/2 → cases Element 1

N/4 = Case remove 1

N/8 =

:

1

$\boxed{\log_2 n}$ — Best case

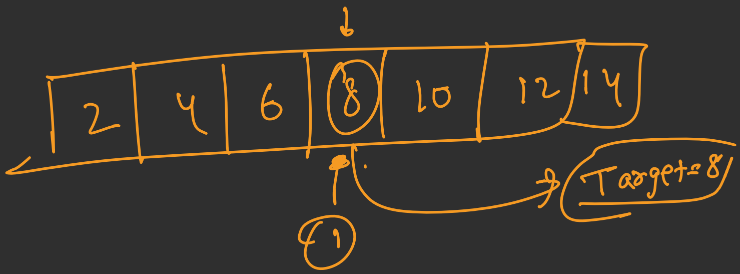
N : 1 operat

$N/2$: 1 operat

$N/4$: 1 oper

⋮

1 elem : 1 operat


2
4
6
8
10
12
14
Target = 8
1