Data Structures

ADT

What?

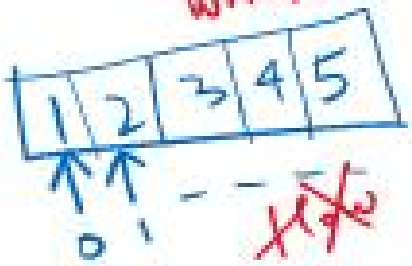
Defs for Data

Operation

Insert - ith

Remove

Add/Append



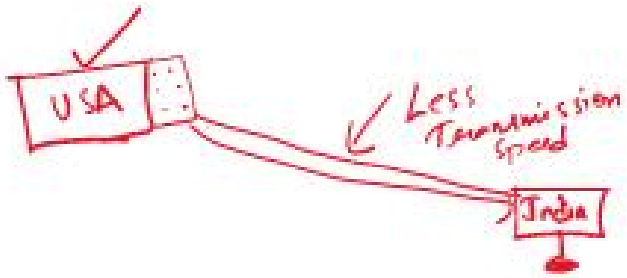
Pseudocode?

→ Partial Eng + Partial Code

→ Won't be following Syntax

Intuition

Idea/Feeling



$1 = I \quad 2 = II$

Big Oh (O)

$n = n^2+2$

$f(n) = O(g(n))$

iff $f(n) \leq c * g(n) \quad \forall n \geq n_0$

$f(n) = 2n+3$

$2n+3 \leq 5n \quad n \geq 1$

$(5n, 7n)$

$f(n) = O(n)$

$2n+3 \leq 5n^2$

$f(n)$ $\quad$ $g(n)$

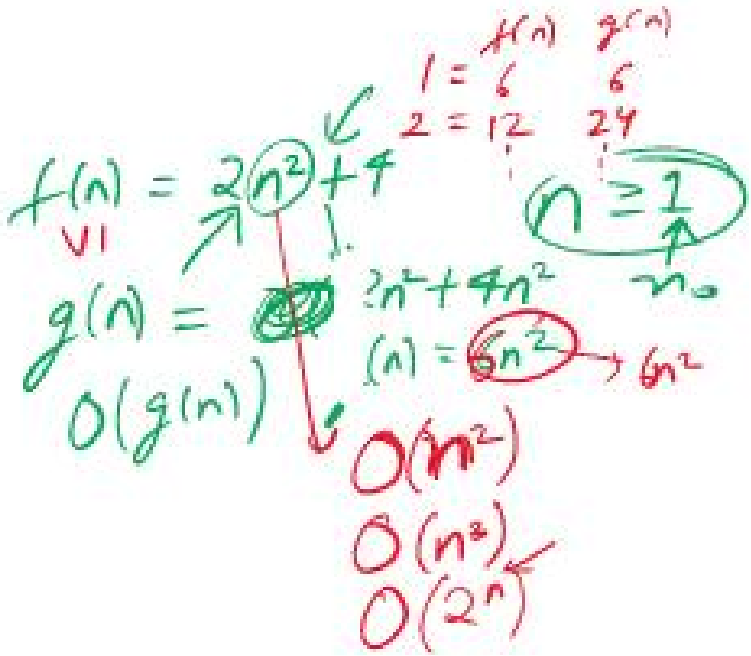
$f(n) = O(n^2)$

$2n+3 \leq 55n^3$

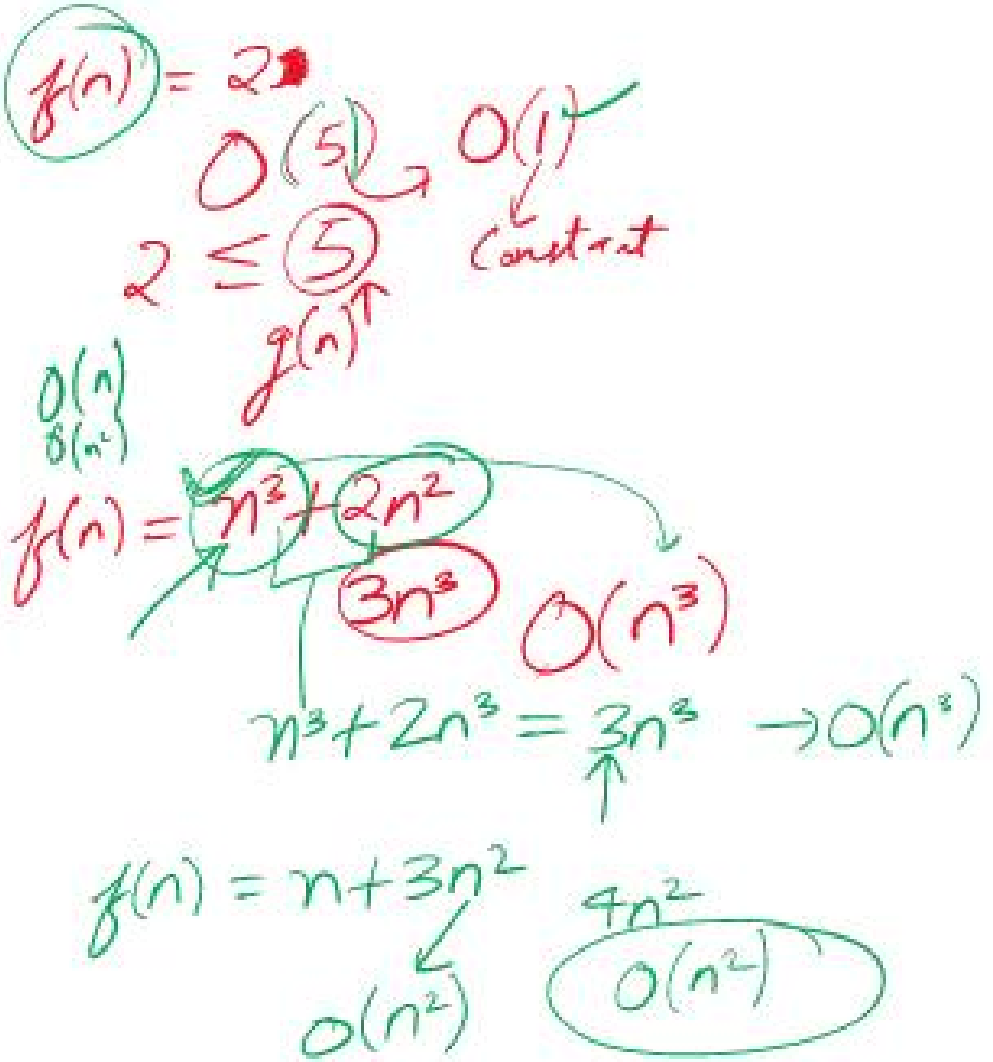
n^4

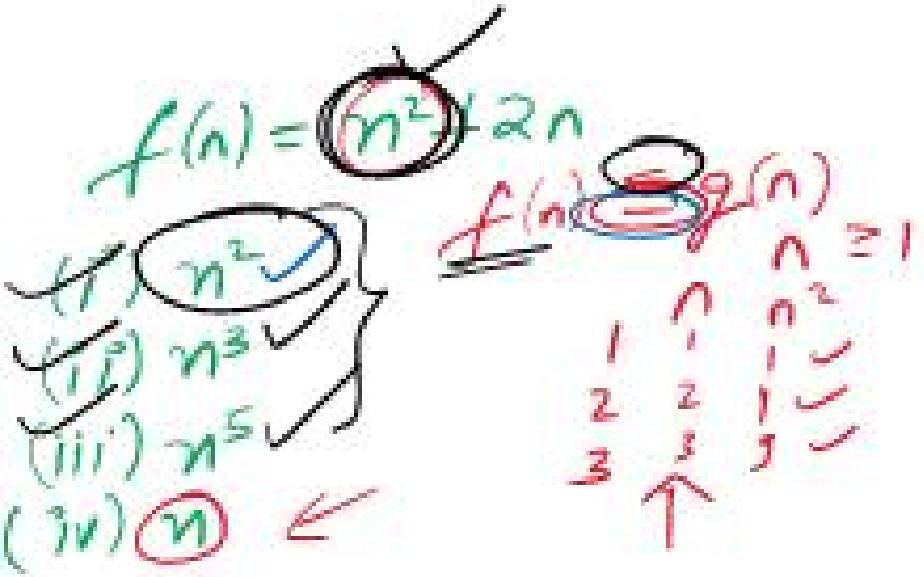
$f(n) = O(n^3)$

$O(n^4)$


f(n) g(n)
1 = 6 6
2 = 12 24
f(n) = 2(n²) + 7
n ≥ 1
n₀
g(n) =
3n² + 4n²
f(n) = 6n² → 6n²
O(g(n))
O(n²)
O(n²)
O(2ⁿ)

)


f(n) = 2
O(5) → O(1)
2 ≤ 5
constant
g(n)
O(n)
O(n²)
f(n) = n² + 2n²
3n³
O(n³)
n³ + 2n³ = 3n³ → O(n³)
f(n) = n + 3n²
O(n²)
4n²
O(n²)



Time funct categories

(i) $O(1)$ → Const

(ii) $O(\log n)$ → Logarithmic

(iii) $O(n)$ → Linear

(iv) $O(n^2)$ → Quadratic

(v) $O(n^3)$ → Cubic

⋮

(vi) $O(2^n), O(3^n) \ldots O(a^n) \ldots O(n^n)$ → Exponential

Const ≤ $1 < b$

#VV Big n

order

const ↓

$1 < \log n < \sqrt{n} < n < n\log n < n^2 < n^3 \ldots\ldots < 2^n < 3^n \ldots\ldots < n^n$

$n\log n$: 3 10

$B=2$

Poly.

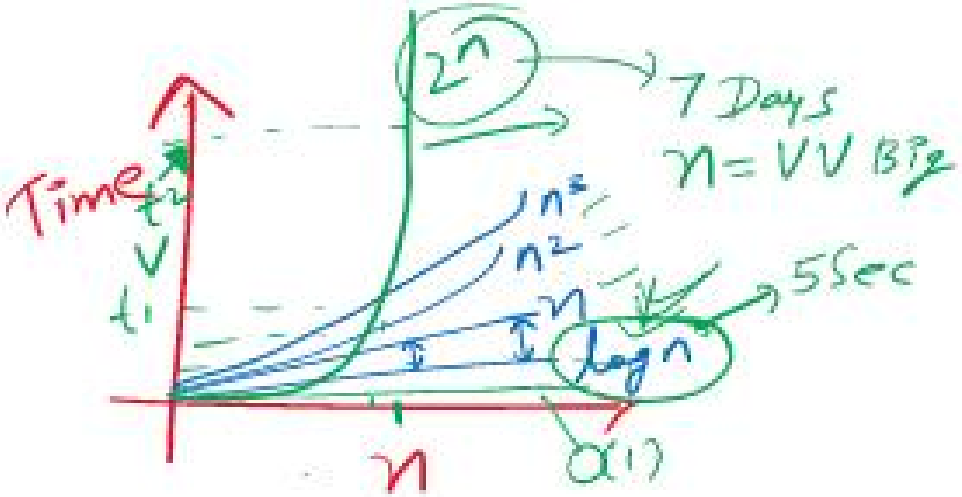
2^{n+1}

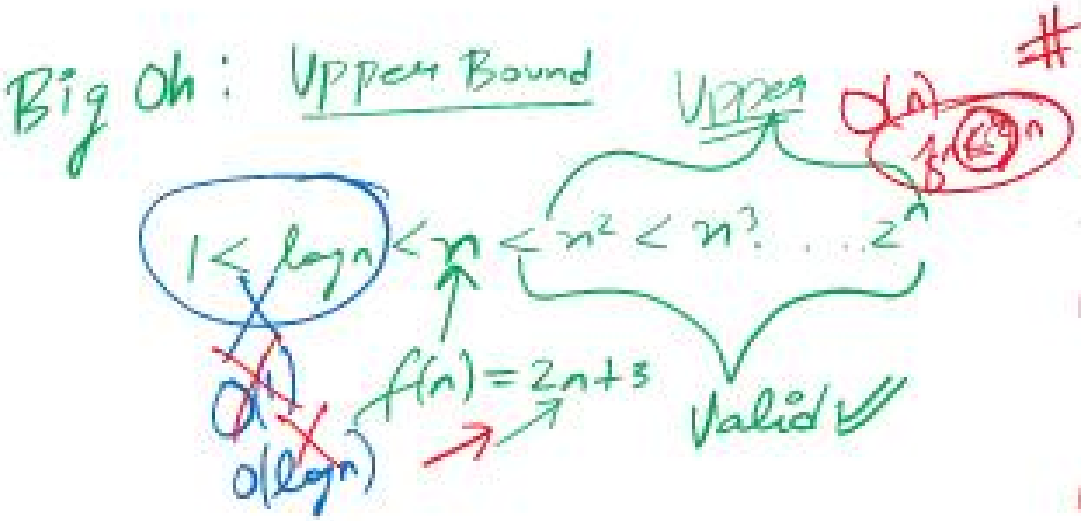
Exp.

log¹ log²

$\log_2 n$	$<$	n	$<$	n^2	$<$	2^n
0	$<$	1		1 ✓		2 ✓
1	$<$	2		4 ✓		4 ✓
2	$<$	4		16 ✓		16 ✓
3	$<$	8		64 ↔ 256 ✓		
3.1	$<$	9		81 ↔ 512 ✓		

Big n


Time
t_2
t_1
2^n
7 Days
n = VV Big
n^3
n^2
n
$\log n$
5sec
n
$O(1)$


Big Oh : Upper Bound
#
Upper
Upper
$1 < \log n < n < n^2 < n^3 \ldots 2^n$
$f(n) = 2n+3$
Valid
$O(1)$
$O(\log n)$

Omega (Ω) Lower Bound

$f(n) \geq c * g(n)$

$\Omega(n), \Omega(n^2) \ldots \Omega(1), \ldots \Omega(2^n)$

$f(n) = \Omega(g(n))$

iff $\Rightarrow f(n) \geq c * g(n)$

$f(n) = 2n+3$

$n \geq n$

$2n+3 \geq n$

$\Omega(n)$

$f(n) = n^2 + 3$

(i) $\log n$

(ii) 1

(iii) n

(iv) n^2

Ω

\# $1 < \log n < n < n^2 \ldots\ldots$

$\Omega(1), \Omega(\log n)$ ~~$\Omega(n^2)$~~

Theta (Θ)

$f(n) = \Theta(g(n))$

iff $\Rightarrow$ $c_1 * g(n) \le f(n) \le c_2 * g(n)$

$f(n) = 2n + 3n$

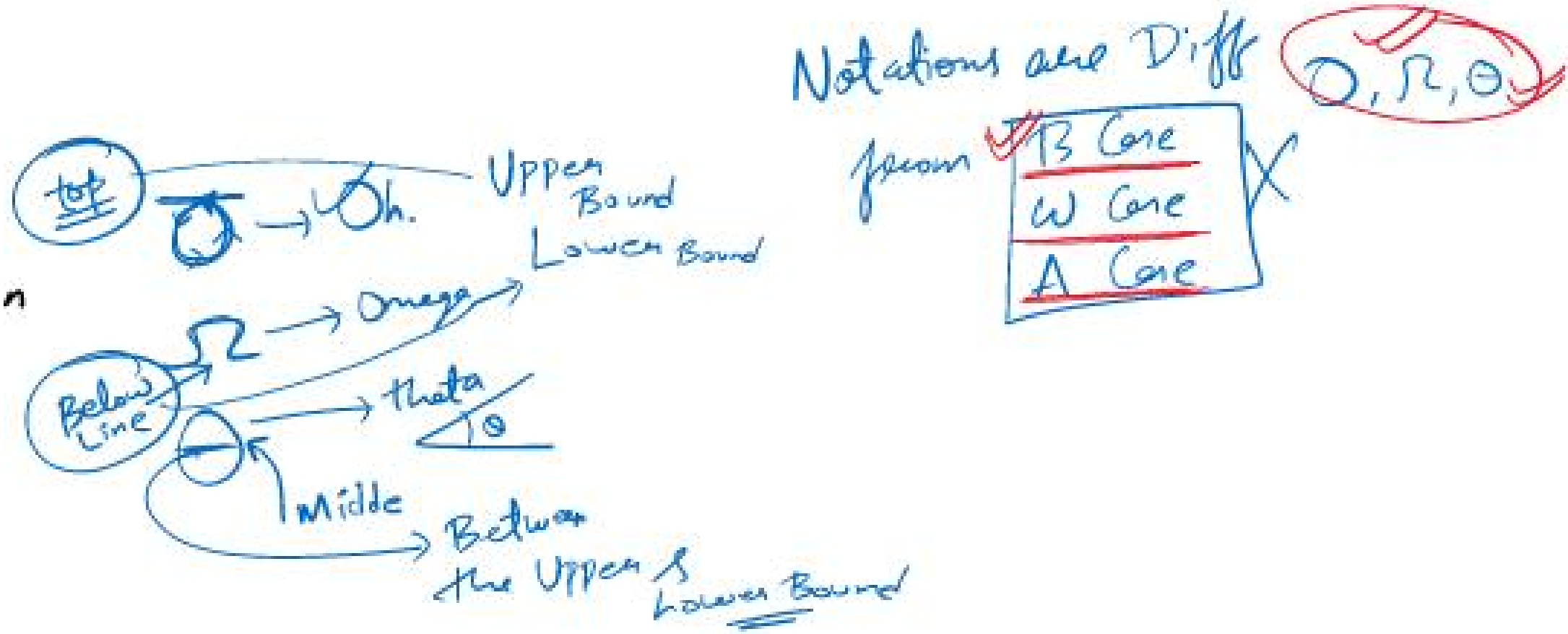
$2n \le 2n + 3 \le 5n$

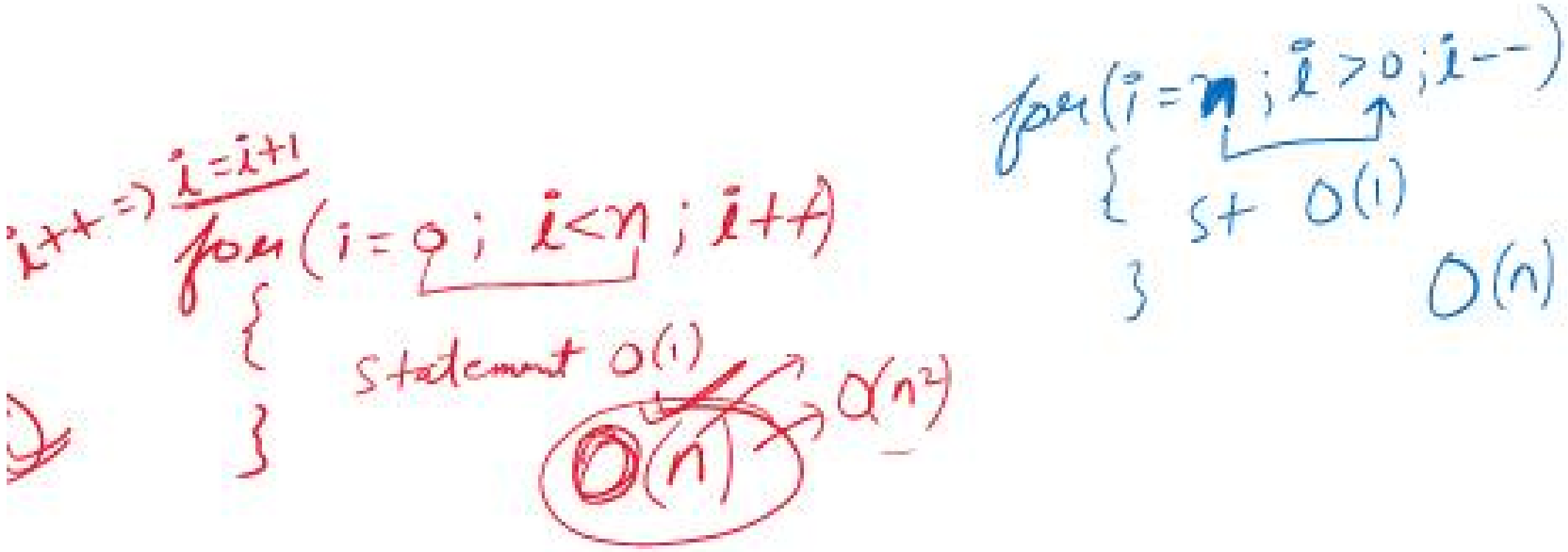
$\Theta(n)$

$2n, 3n, 4n, 5n$

$1 < \log n < n < n^2 < \ldots 2^n < n^n$

$g(n)$


Notations are Diff
from
B Case
W Case
A Case
O, Ω, Θ
Upper Bound
Lower Bound
Omega
Below Line
theta
Middle
Between the Upper & Lower Bound


for(i=n; i>0; i--)
{ s+ O(1)
}
O(n)
i++ => i=i+1
for(i=0; i<n; i+)
{
}
statement O(1)
O(n)
O(n²)

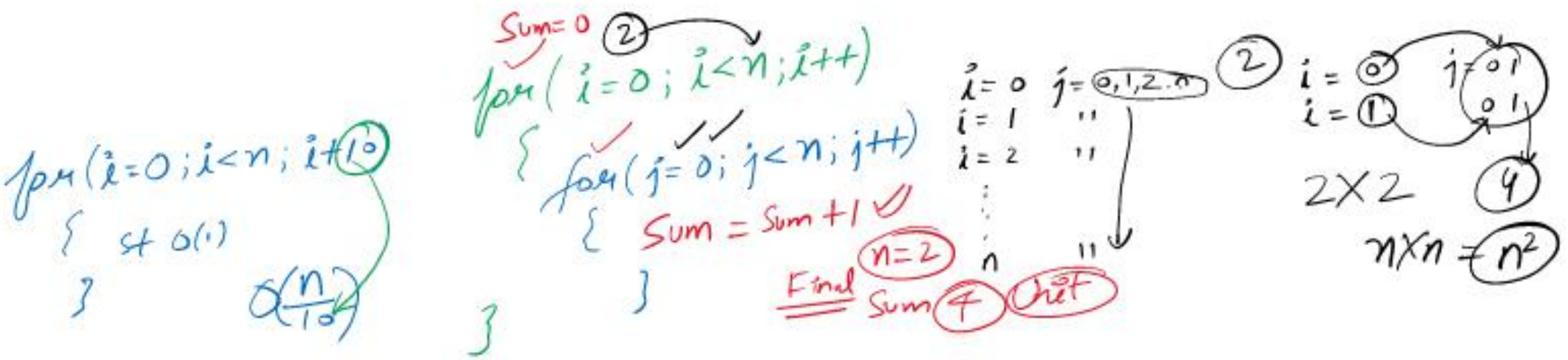
```
for(i=0; i<n; i+2)
{
    st O(1)
}
```

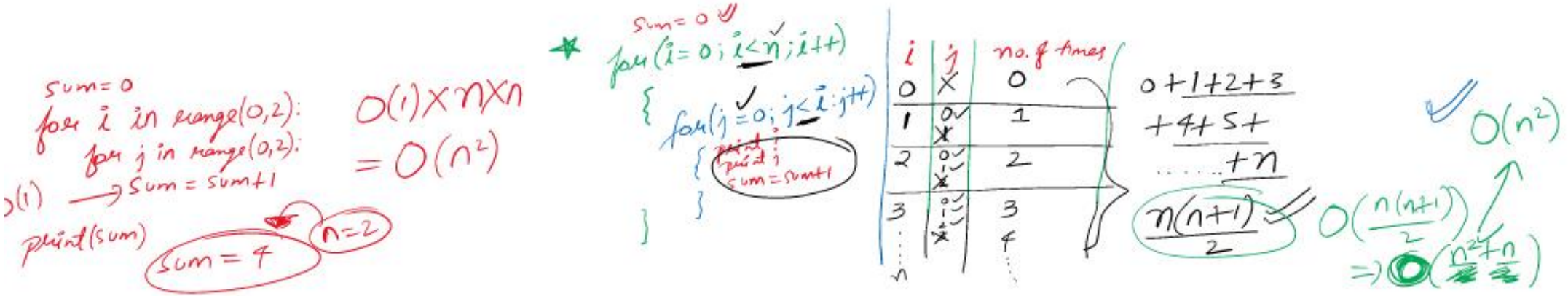
```
for(i=0; i<n; i+4)
{ st O(1)
}
```

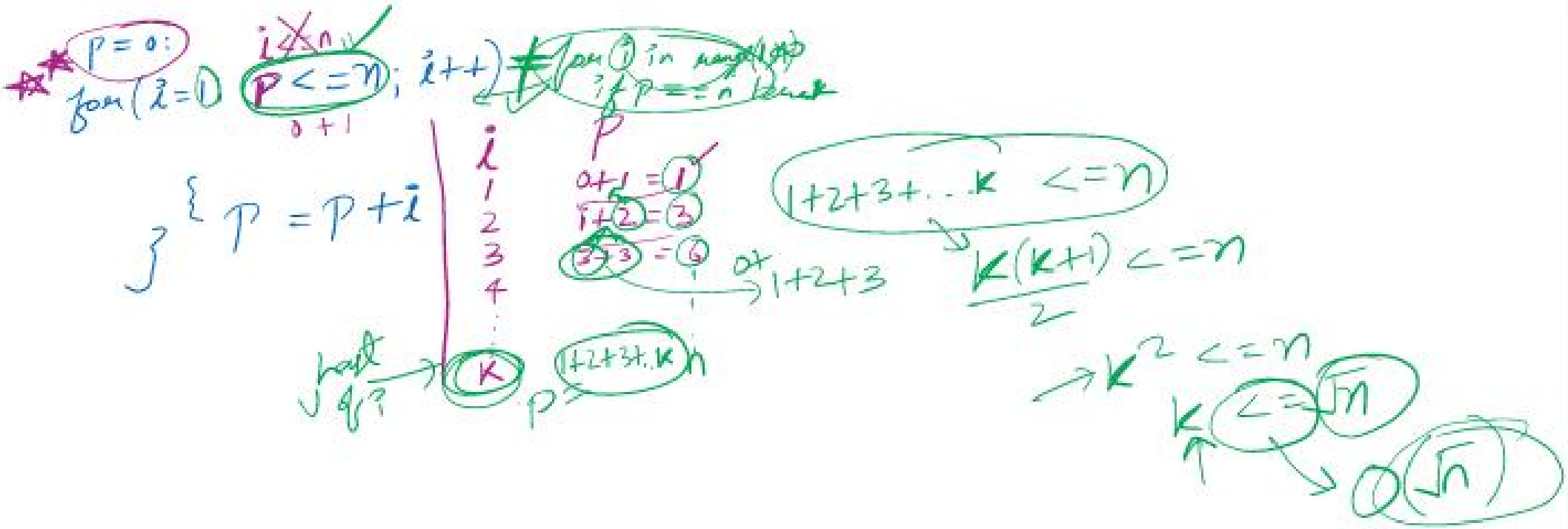
$O(\frac{n}{4})$

1 2 3 4 5 6 7 8

$\frac{8}{4}$ 2 $\frac{n}{4}$


for(i=0; i<n; i++) 10
{ st O(1)
}
O(n/10)
Sum=0 2
for(i=0; i<n; i++)
{
for(j=0; j<n; j++)
{ Sum = Sum+1
}
}
n=2
Final Sum 4 Ref
i=0 j=0,1,2..n 2
i=1 "
i=2 "
n "
i=0 j=0 1
i=1 0 1
2X2 4
nXn = n²


sum = 0
for i in range(0,2):
for j in range(0,2):
O(1) → sum = sum+1
print(sum)
sum = 4
n=2
O(1) × n × n
= O(n²)
sum = 0
for (i = 0; i < n; i++)
{
for (j = 0; j <= i; j++)
{
printf("%d", j);
sum = sum+1
}
}
i
j
no. of times
0 + 1 + 2 + 3
+ 4 + 5 +
..... + n
n(n+1)/2
O(n²)
O(n(n+1)/2)
=> O(n²/2 + n/2)


p = 0;
for(i=0
p <= n; i++)
0 + 1
{ p = p + i
i
1
2
3
4
K
p
0+1 = 1
1+2 = 3
3+3 = 6
or 1+2+3
1+2+3+...K <= n
K(K+1)/2 <= n
K^2 <= n
K <= √n
O(√n)