for ( ) $\longrightarrow (n+1)$

for ( ) $\longrightarrow (n+1) \longrightarrow n(n+1)$

$\longrightarrow n$

$+n \longrightarrow n \times n$

$$S(n) = (n+1) + n(n+1) + n \times n$$

$$= n+1+n^2+n + n^2$$

$$= \cancel{2}n^2 + \cancel{2n+1}$$

$$= O(n^2)$$

A — n^2

B — n^2

C — n^2

n — 1

m → 1

i — 1

j — 1

$3n^2+4 \rightarrow O(n^2)$

Ex Algorithm Multiply (A, B, n)

for (i=0; i<n; i++)

$n+1$ — for (i=0; i<n; i++)

{

$(n+1) \times n$ — for (j=0; j<n; j++)

{

$n \times n$ — c[i,j]=0;

$(n+1) \times n \times n$ — for(k=0; k<n; k++)

$\underline{n \times n \times n}$ — { c[i,j]=c[i,j]+A[i,k]*B[k,j]

}

}

}

$$f(n) = 2n^3 + 3n^2 + 2n + 1$$

$O(n^3) \rightarrow$ cubic time complexity

3×3
$= 9$
$n \times n = n^2$

$A \longrightarrow n^2$

$B \longrightarrow n^2$

$C \longrightarrow n^2$

n — 1

i — 1

j — 1

k — 1

$$s(n) = 3n^2 + 4$$

$O(n^2) \rightarrow$ quadratic complexity

space

Ex

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

n

$O(n)$

Ex:

```
for ( i=n; i>0; i--)
{
   ----
}
```

n $O(n)$

Ex:

```
for ( i=1; i<n; i=i+2)
{
   ____
}
```

$n/2$

$O(n)$

$f(n) = \frac{n}{2}$

$O(n)$

Ex: for (i=1; i<n; i=i+20)

Ex: for (i=1; i<n; i= ...)

{

—

}

$$f(n) = \frac{n}{20}$$

$$O(n)$$

Ex: for (i=0; i<n; i++) → (n+1)

{ for (j=0; j<n; j++) → n(n+1)

{

—

}

}

$$O(n^2)$$

Ex: for (i=0; $\boxed{i<n}$; i++)   n

{ for (j=0; $\boxed{j<i}$; j++)

{

—

}

}

i	j	No of times
0	~~0~~	0
1	0, ~~1~~	1
2	0, 1, ~~2~~	2
3	0, 1, 2, ~~3~~	3
⋮	0, 1, ...	n

No of time execution

$$1+2+3+\dots+n = n(n+1)$$

$\overline{\quad 2 \quad}$ n

2
3
$n-1$

$$f(n) = \frac{n^2+1}{2}$$

$$O(n^2)$$

<u>Ex</u>

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

i	p
1	0+1=1
2	1+2=3
3	1+2+3=6
4	1+2+3+4
5	1+2+3+4+
⋮	
k	1+2+...k

<u>Induction proof-</u>

Assume $\boxed{p > n}$ *

$\therefore p = 1+2+\cdots k$

$$\boxed{p = \frac{k(k+1)}{2}}$$

$$p = \frac{k^2+k}{2} \rightarrow \text{Identified p value}$$

$1+2+\cdots k$

$\frac{k(k+1)}{2}$

Sum of natural numbers

$$\boxed{\frac{k^2+k}{2}} > n$$

$$\left(k^2 \quad k\right) > n$$

$f(n)\left(\frac{1}{\cancel{2}}+\frac{1}{\cancel{2}}\right)$

$$\boxed{k^2 > n}$$

sqroot $\rightarrow k > \sqrt{n}$

$$\boxed{O(\sqrt{n})}$$

Const, linear, quad, cubic, sqroot, log, exp

Ex:

```
for (i=1; i<n; i=i*2)
{
    ——
}
```

i
1
$1\times2=2$
$2\times2=2^2$
$2^2\times2=2^3$
$2^3\times2=2^4$
⋮
$i=2^k$

$$\boxed{\text{Assume } i>=n}$$

$i=2^k$

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

$2^k = n$

Apply logs

$$k = \log_2 n$$

$$O(\log_2 n)$$

Ex

```
for (i=1; i<=n; i++)
{  st...
}
```

Ex

```
for (i=n; i>=1; i=i/2)
{  ____
}
```

i/n

$n/2$

$n/2^2$

$n/2^3$

$\vdots$

$n/2^k$

Assume $i < 1$

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

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

$$n = 2^k$$

$n = 2^k$

$k = \log_2 n$

$O(\log_2 n)$

Ex:

```
p=0;
for (i=1; i<n; i=i*2)
{   p++;
}
for (j=1; j<p; j=j*2)
{   st ---
}
```

$p = \log n$

$\log p$

$O(\log \log n)$

$O(\log p)$

$O(\log \log n)$

Ex: for(i=0; i<n; i++) → (n+1)

n+1

{
for (j=1; j<n; j=j*2) $n \times \log n$
{
st — $\log n$ $n \times \log n$
}
}

$n \log n$

$n \times$

$n \log n$

$\cancel{2} n \log n + \cancel{n} \cancel{+1}$

$O(n \log n)$

$2n \log n + n + 1$

$O(n \log n)$

Types of Time functions

$O(1) \rightarrow$ Constant

$O(\log n) \rightarrow$ Logarithmic

$O(n) \rightarrow$ Linear

$O(n^2) \rightarrow$ Quadratic

$O(n^3) \rightarrow$ Cubic

$O(2^n) \rightarrow$ Exponential

$O(3^n) \rightarrow$ " —

$O(n^n) —$ " —

$O(n)$

Imp

$$\left.\begin{aligned} f(n) &= 2 \\ f(n) &= 200 \\ f(n) &= 5000 \end{aligned}\right\} \Rightarrow O(1)$$

Imp

$$\left.\begin{aligned} f(n) &= 2n+3 \\ f(n) &= 500n+700 \\ f(n) &= \frac{n}{500}+6 \end{aligned}\right\} \Rightarrow O(n)$$

Imp

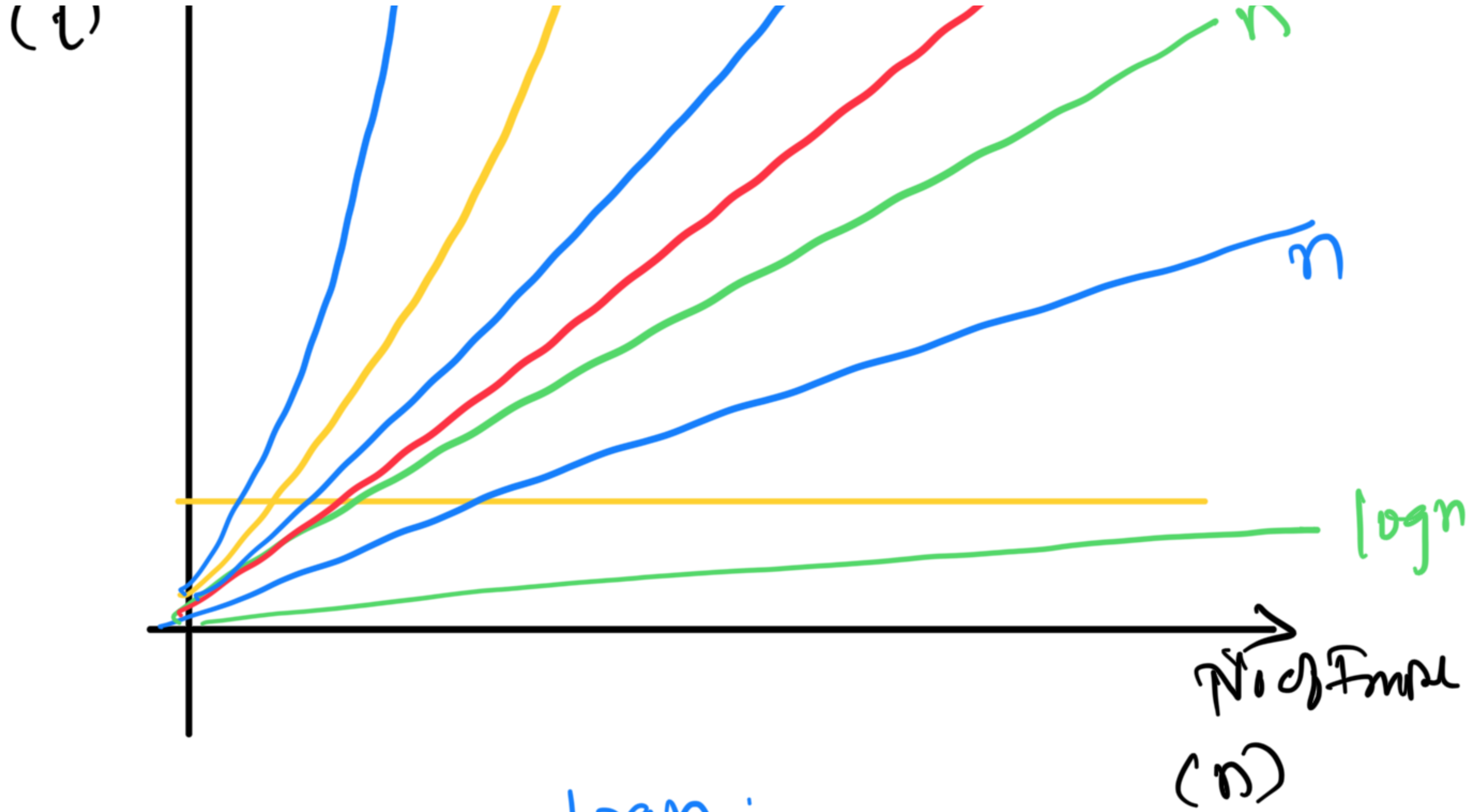
$$f(n) = 107\underline{n^3} + 10n^2 + 57n + 40$$
$$= O(n^3)$$

Important

Best

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

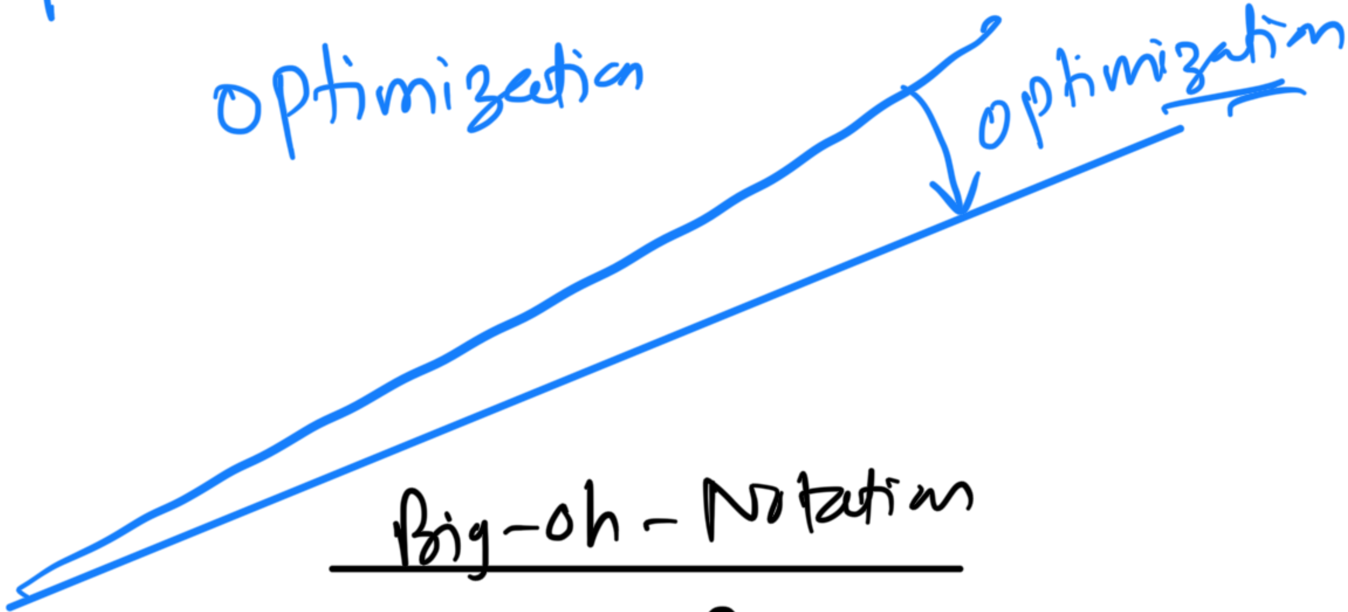
↑ Low ↑ High



$$n^2 \longrightarrow n \xrightarrow{\log n} O(1)$$

Constant

optimization



Big-oh-Notation

- $O(1) \rightarrow$ Best
- $O(\log n) \rightarrow$ Good
- $O(n) \rightarrow$ Fair
- $O(n \log n) \rightarrow$ Good / worst
- $O(n!) \rightarrow$ } worst
- $O(2^n) \rightarrow$ }

| $O(n^c) \to$)

Ex:

0									
8	6	12	5	9	7	4	3	16	18
↑									↑

Search key → 8

Best case → No of comp. ⇒ 1 index

key 8 → $O(1)$

→ $\Omega(1)$

Worst case → No of comp ⇒ 9 index

key 18 → $O(n)$

$W(n) \to n \to O(n)$

Average Case → No of comp ⇒ 4/7

key = 5, 4 → $O(n)$

$\left.\begin{array}{l} O(n) \\ \Theta(n) \end{array}\right\}$ $\frac{n+1}{2}$

Average case = $\dfrac{\text{All possible case times}}{\text{No of cases}}$

$$\text{Avg time} = \frac{1+2+3+\dots n}{n}$$

$$= \frac{\frac{\cancel{n}(n+1)}{2}}{\cancel{n}} = \frac{n+1}{2}$$

$\uparrow$

$O(n)$

$\downarrow$

$\Theta(n)$

Remember

Linear Search

$B(n) = 1$

$B(n) = O(1)$

$B(n) = \Omega(1)$

$B(n) = \Theta(1)$

Worst $(n) = n$

$W(n) = O(n)$

$W(n) = \Omega(n)$

$W(n) = \Theta(n)$

$$\boxed{\begin{aligned} B(n) &= O(1) \\ W(n) &= O(n) \\ A(n) &= \frac{n+1}{2} \end{aligned}}$$