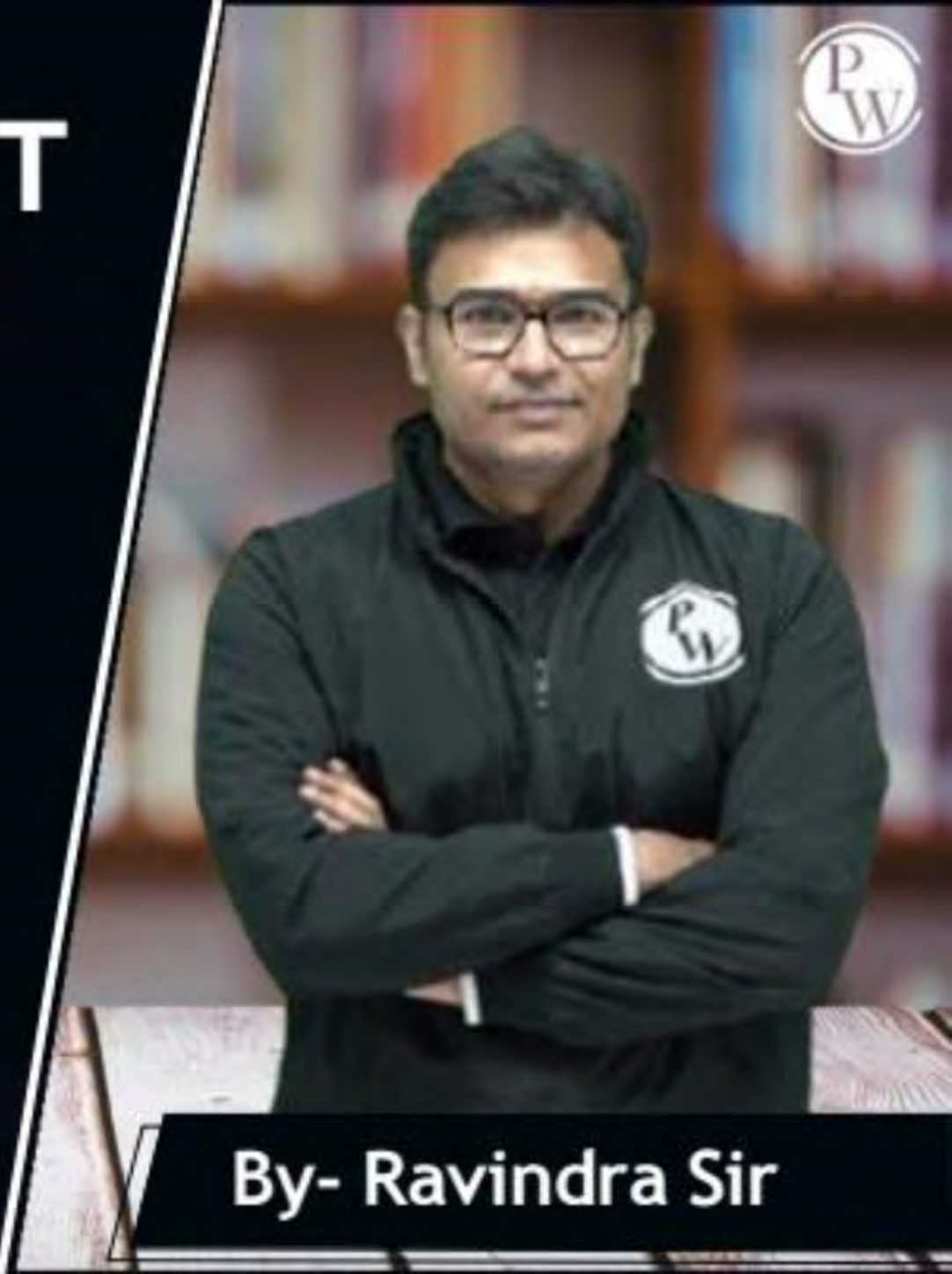
Computer Science & IT

ALGORITHMS

Algorithm

Lecture No. 2



By- Ravindra Sir

Recap of Previous Lecture

- Topic
- Topic
- Topic

TC

Topics to be Covered

- Topic
- Topic
- Topic

TC

Join Telegram

@RBRSIRPW

Inspiring Stories : M. Balakrishnan

Background: Professor who wanted to make life easier for the blind.

Education: PhD in Computer Science, IIT Delhi.

Achievements: Co-created the SmartCane, a stick with sensors that helps blind people detect obstacles.

Impact: Thousands of visually impaired people in India move around more freely.

Inspiring Stories : D. Udaya Kumar

Background: Grew up in Tamil Nadu, passionate about type design.

Education: PhD in Design, IIT Bombay.

Achievements: Designed the ₹ symbol for the Indian Rupee.

Impact: Gave India a global currency symbol seen on every keyboard and note.

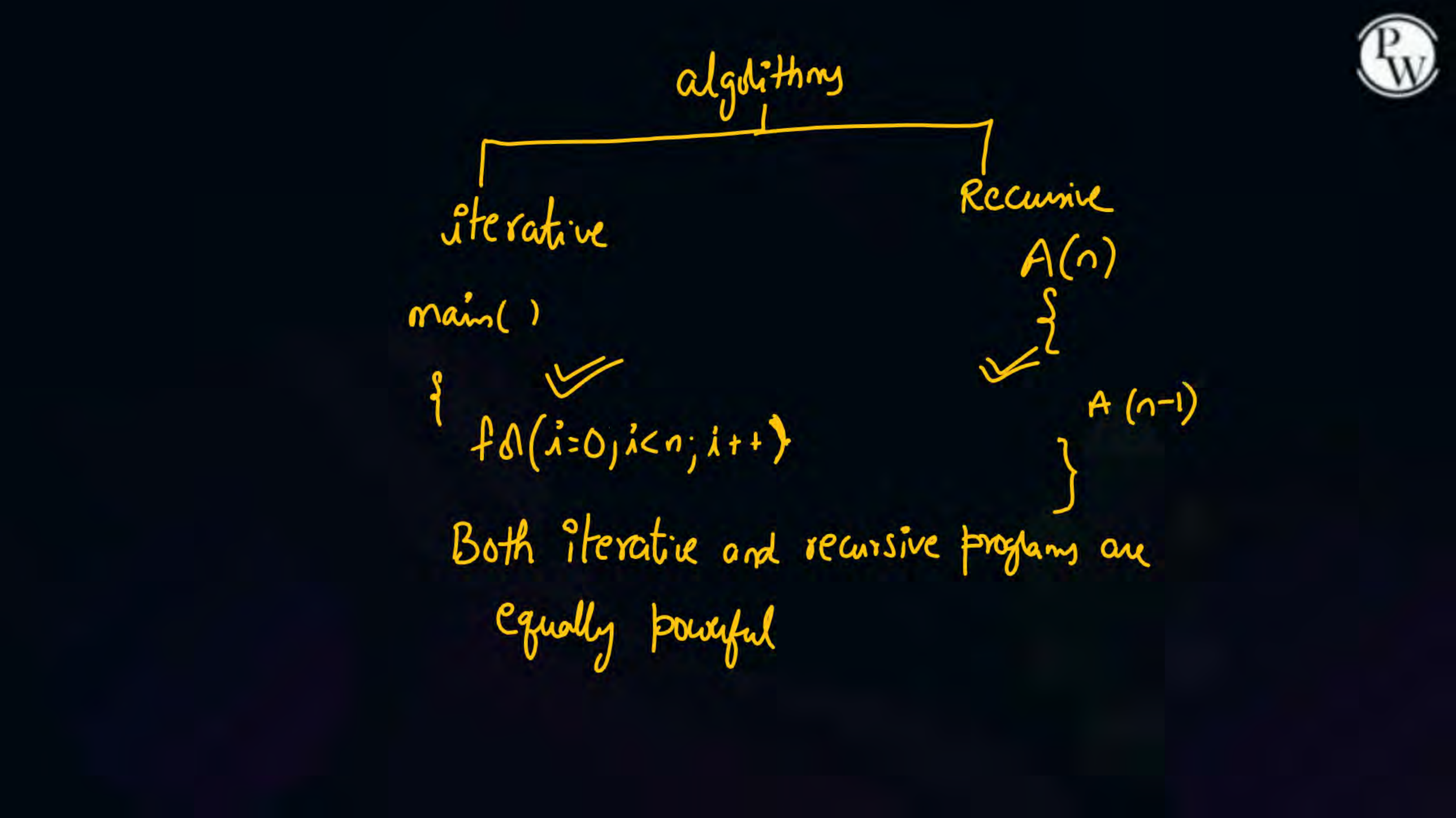
Inspiring Stories : Sirshendu De

Background: From West Bengal, wanted to solve the arsenic water crisis.

Education: PhD in Chemical Engineering, IIT Kanpur.

Achievements: Made cheap filters from local soil to clean arsenic from water.

Impact: Villages in Bengal and beyond now drink safe water at very low cost.


algorithms
iterative
Recursive
main( )
{
for(i=0; i<n; i++)
A(n)
{
A (n-1)
}
Both iterative and recursive programs are
equally powerful

```
main()
{
  a=b+c;
}
```

TC = O(1)

```
main()
{  for(int i=0; i<n; i++)
       Print(RBR)
}
```

O(n)

for(i=0; i<n; i=i+2)

Pf(RBR)

SC = O(1)

Let the program executes (k) times

iteration	1.	2.	3	----	k.	(k+1)
Value of i	0	2.	4.		2(k-1)	↓ fail (2k)

$i = n$

$2k = n \Rightarrow k = n/2$

TC = O(k) = O(n)

Ω(n)

Θ(n)

for (i=1; i<n; i=i*2)

Pf (RBR)

Let the program execute (k) time

iteration	1	2	3	k
i	1	2	2^2 2^3	2^{k-1}

(k+1) fail

2^k

$i = n$

$2^k = n \Rightarrow k = \log_2 n$

$O(k) = O(\log n)$

$= \Omega(\log n)$

$= \Theta(\log n)$

for($i=n$; $i \geq 2$; $i=\sqrt{i}$)

Pf(RBR)

Let loop runs for k times.

iteration	1	2	3	4	----	K	(K+1) fail
i	n	$n^{1/2}$	$n^{1/2^2}$	$n^{1/2^3}$	....	$n^{1/2^{K-1}}$	$n^{1/2^K}$

$i = 2$

$n^{1/2^K} = 2$

$\frac{1}{2^k} \log_2 n = 1 \Rightarrow 2^k = \log_2 n$

$\Rightarrow k = \log \log_2 n$

$O(k) = O(\log \log_2 n)$

```
int i=n
while (i>1)
{ pf(RBR)
  i=i/2
}
```

Let the loop execute K times

iteration	1	2	3	4	K	K+1
i	n	n/2	$n/2^2$	$n/2^3$	$\frac{n}{2^{K-1}}$	1
						$\frac{n}{2^K}$

$i = 1$

$\frac{n}{2^K} = 1 \Rightarrow 2^K = n \Rightarrow K = \log n$

$O(K) = O(\log n)$

$\Omega(\log n)$

$\Theta(\log n)$

```
for(i=1; i^2<n; i++)
  {  pf(RBR)
  }
```

```
for(i=1; i<√n; i++)
```

$O(\sqrt{n})$

$\Omega(\sqrt{n})$

$\Theta(\sqrt{n})$

```
for(i=n/2; i<n; i=2*i)
  { Pf(RBR)
  }
```

iteration	1	2	O(1)
i	n/2	n	

a) Rank ✓ b) Simple ✓

```
fA(i=0; i<n, i++)
    fA(j=0; j<n; j++)
    { pf(RBR) }
```

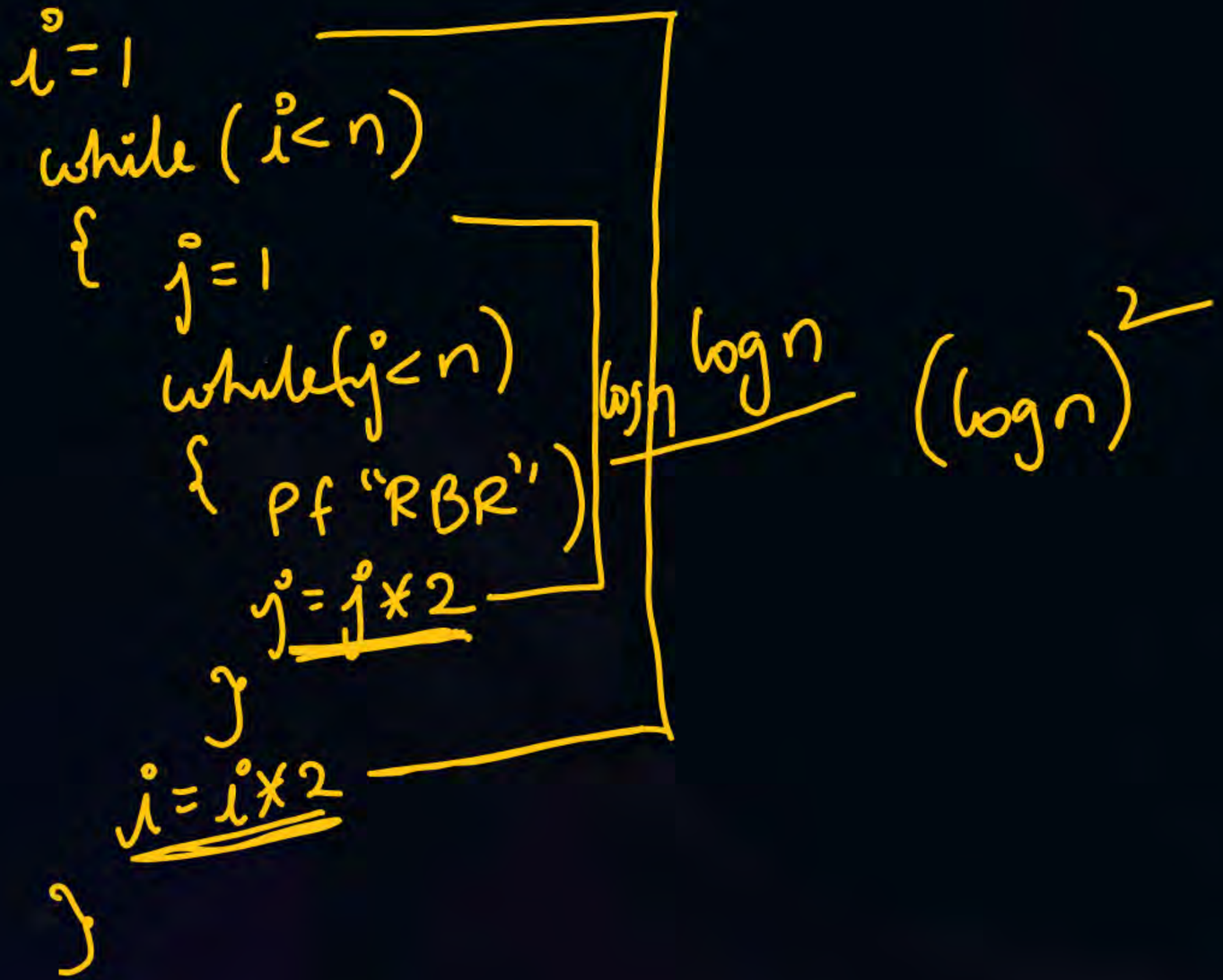
$O(n^2)$

$i=0$	$i=1$	$i=2$	..... $i=n-1$
$j=0,1,2 \sim n-1$	$j=0,1,2 \cdots n-1$	$j=0,1,2 \cdots n$	$j=0,1,2, \cdots n-1$
n	n	n	n

$n \times n = O(n^2)$

$= \Omega(n^2)$

$= \Theta(n^2)$


i=1
while (i<n)
{ j=1
while(j<n)
{ Pf "RBR")
j=j*2
}
i=i*2
}
logn logn
(logn)^2

fol(int $i=n^2$; $i>1$; $i=i/2$)

Pf(RBR)

Let us assume this loop runs fol k times

iteration:	1	2	3	4	----	k	(k+1)
i:	n^2	$n^2/2$	$n^2/2^2$	$n^2/2^3$		$\frac{n^2}{2^{k-1}}$	$\frac{n^2}{2^k}$ ✓

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

$\Rightarrow k=\log n^2=2\log n$ ✓

$O(k)=O(\log n)$

$\Omega(\log n)$

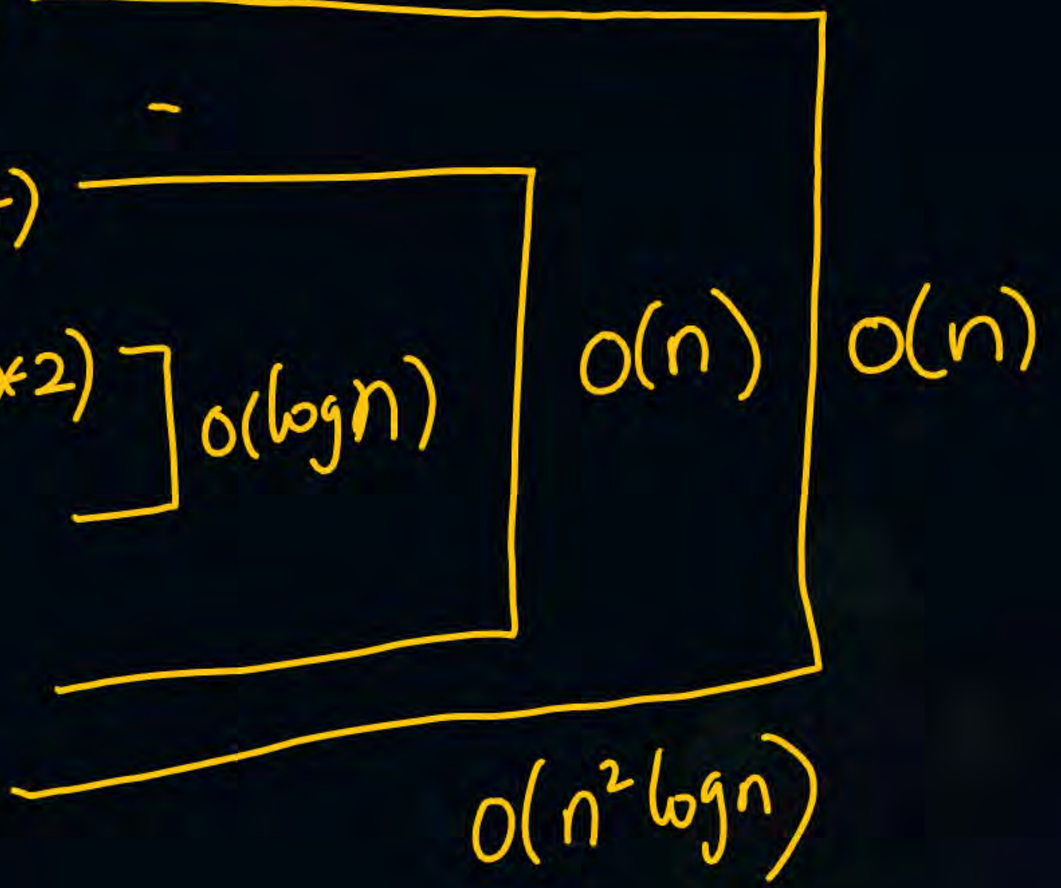
$\Theta(\log n)$

```
for(i=n^2; i>1; i=i/2)
{
    for(j=1; j<n^3; j++)
    {
        pf(RBR)
    }
}
```

$O(n^3)$ $O(\log n)$

$O(n^3 \log n)$

```
for(i=n/2 ; i<n; i++)
 {  for(j=1; j<=n/2; j++)
    {  for(k=1; k<=n; k=k*2)
       {  Pf("RBR")
       }
    }
 }
```



```
int i=1, S=1;
while (S<=n)
  {
    i++;
    S = S+i;
    Pf(RBR)
  }
```

Let the loop execute k times

iteration	value i	Value S
1	2	1+2
2	3	1+2+3
3	4	1+2+3+4
⋮	⋮	⋮
K	k+1	1+2+3+4 ⋯ k+1 ✓ k^2 ✓

$S > n$

$1+2+3+\cdots+k+1 > n$

$\frac{(k+1)(k+2)}{2} > n \Rightarrow k^2 = O(n)$

$\Rightarrow k = O(\sqrt{n})$

```
int i=1, S=1
{ while(S<=n)
  { i++;
    S=S+i^2;
    Pf(RBR)
  }
}
```

iteration	i	S
1	2	$1+2^2$
2	3	$1+2^2+3^2$
3	4	$1+2^2+3^2+4^2$
4	5	⋮
⋮		⋮
K	K+1	$1+2^2+3^2+\cdots+(K+1)^2 = O(K^3)$

$S > n$

$O(K^3) > n \Rightarrow O(K) = O(\sqrt[3]{n})$

a) Rank b) Simple algo

p=0

for(i=1; i<=n; i=i*2) } logn

{ p++ logn

}

for(j=1; j<p; j=j*2) } loglogn

Pf (RBR)

logn

O(logn + loglogn)

O(logn)

```
for(i=1; i<=n; i++)
{   j=2
    while(j<=n)
    {   j=j^2;
    }
}
```

loglogn

O(n)

iteration	j
1	2
2	2^2
3	2^{2^2}
4	2^{2^3}
⋮	⋮
K	$2^{2^{K-1}}$
K+1	2^{2^K}

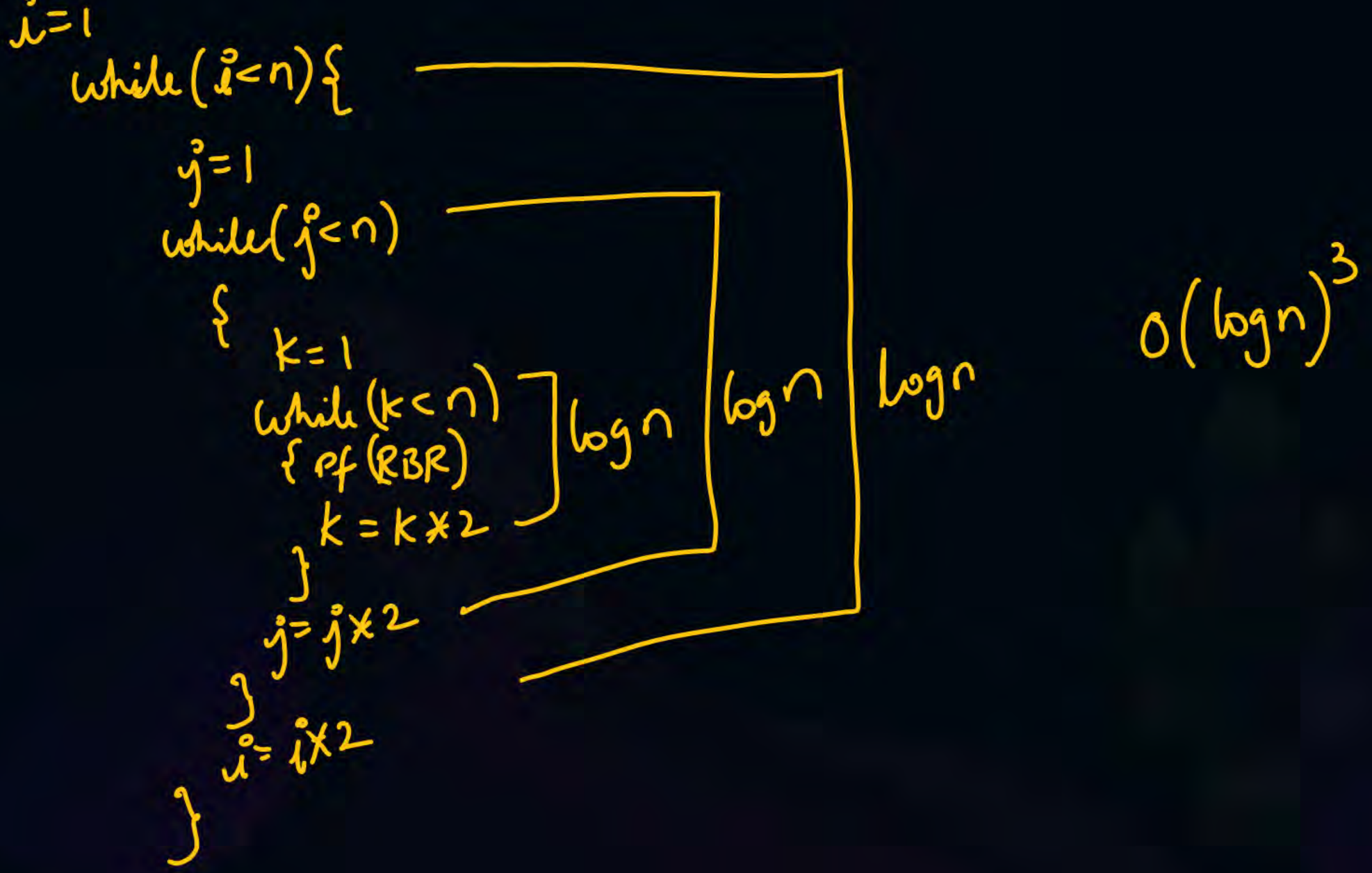
$j > n$

$2^{2^k} > n$

$2^k = \log n$

$k = \log\log n$

$O(n \log\log n)$


i=1
while(i<n){
j=1
while(j<n)
{
k=1
while(k<n)
{ pf(RBR)
k=k*2
}
j=j*2
}
i=i*2
}
logn
logn
logn
O(logn)^3

```
for(i=1; i<=n; i++)
{ for(j=1; j<=n; j=j+i)
    { pf(RBR)
    }
}
```

i	j
1	1, 2, 3, 4. ---- n time
2	1, 3, 5. ---- ---- n/2 time
3	1, 4, 7. ---- n/3 times
⋮	⋮
n	1 — 1 time

$$\text{Total} = n + \frac{n}{2} + \frac{n}{3} + \frac{n}{4} + \cdots + \frac{n}{n}$$

$$= n\left(1 + \tfrac{1}{2} + \tfrac{1}{3} + \tfrac{1}{4} + \cdots + \tfrac{1}{n}\right)$$

$$O(n \log n)$$ ($\log n$ terms)

```
i = 2
while (i < n)
    i = i^2;
```

iteration :-	i
1	2
2	2^{2^1}
3	2^{2^2}
4	2^{2^3}
⋮	⋮
K	$2^{2^{K-1}}$
K+1	2^{2^K}

Why fail

$i = n$

$2^{2^k} = n$

$2^k = \log n$

$K = \log\log n$

```
i = 25
while(i<n)
    i = i^3;
```

iteration	i
1	25^{3^0}
2	25^{3^1}
3	25^{3^2}
⋮	⋮
k	$25^{3^{k-1}}$
k+1	25^{3^k}

$i = n$

$25^{3^k} = n$

$3^k = \log_{25} n$

$k = \log_3 \log_{25} n$

$O(k) = O(\log \log n)$

THANK - YOU