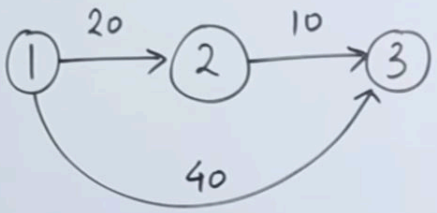
Dijkstras Algorithm

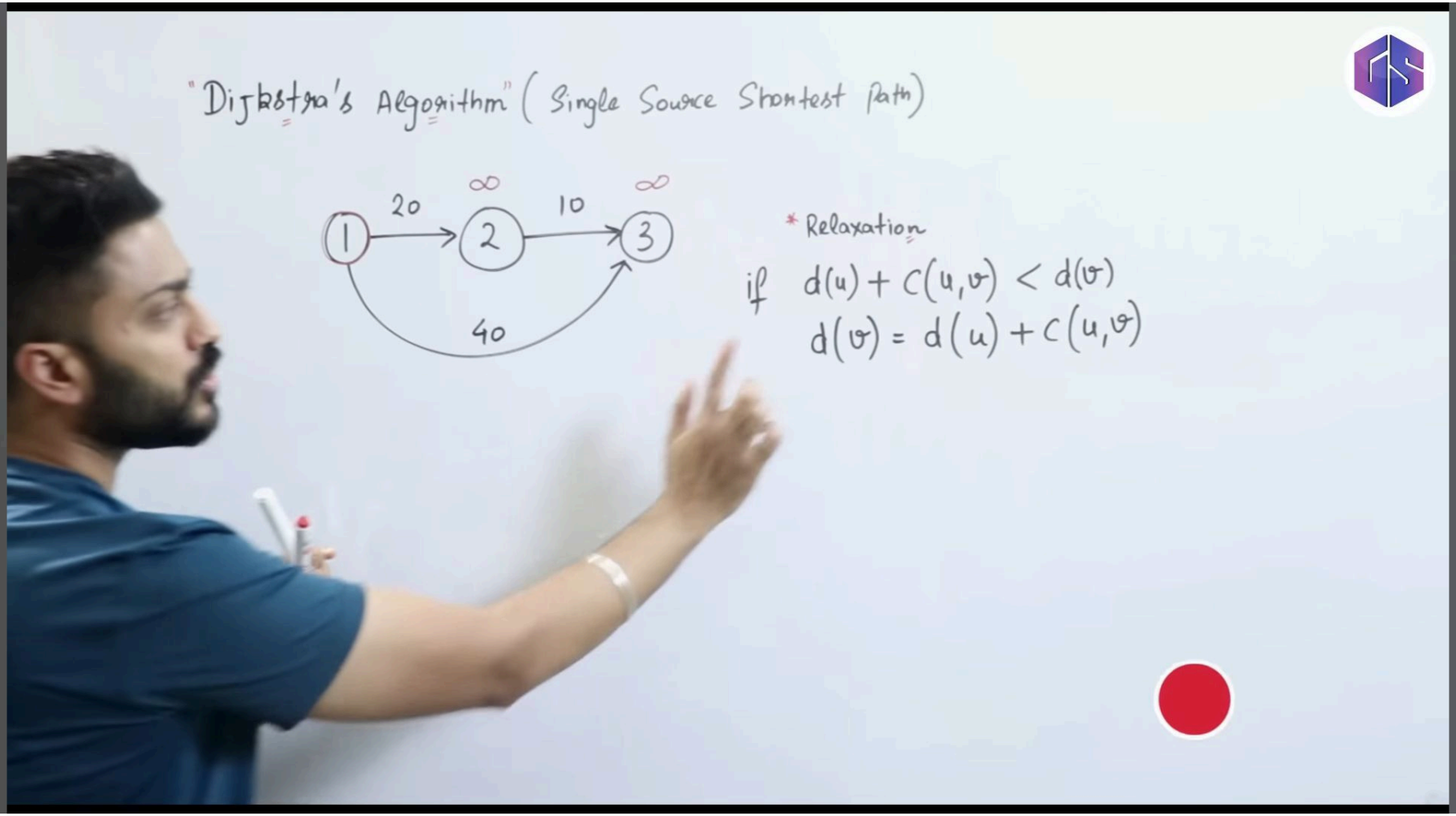
"Dijkstra's Algorithm" (Single Source Shortest Path)



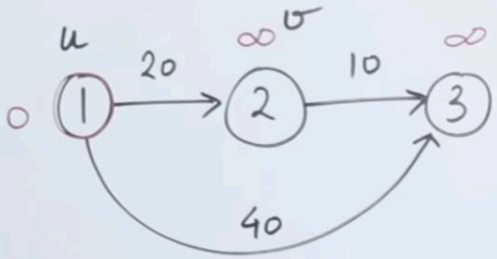
* Relaxation

if $d(u) + c(u,v) < d(v)$

$d(v) = d(u) + c(u,v)$


"Dijkstra's Algorithm" (Single Source Shortest Path)
20
10
40
∞
∞
1
2
3
* Relaxation
if d(u) + c(u,v) < d(v)
d(v) = d(u) + c(u,v)
SUBSCRIBE

"Dijkstra's Algorithm" (Single Source Shortest Path)



* Relaxation

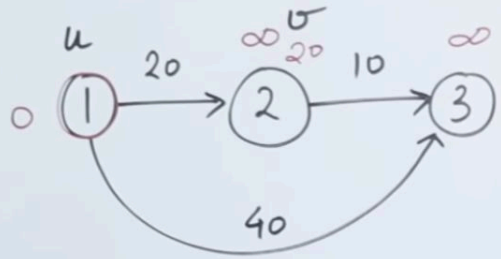
if $d(u) + c(u,v) < d(v)$

$d(v) = d(u) + c(u,v)$

$d(u) + c(u,v) \quad d(v)$

$0 \quad 20 \quad < \quad \infty$

"Dijkstra's Algorithm" (Single Source Shortest Path)



* Relaxation

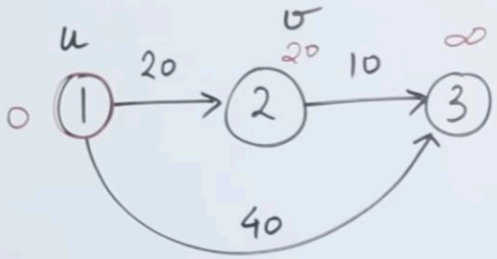
if $d(u) + c(u,v) < d(v)$

$d(v) = d(u) + c(u,v)$

$d(u) + c(u,v) \quad d(v)$

$0 + 20 \quad < \quad \infty$

"Dijkstra's Algorithm" (Single Source Shortest Path)



* Relaxation

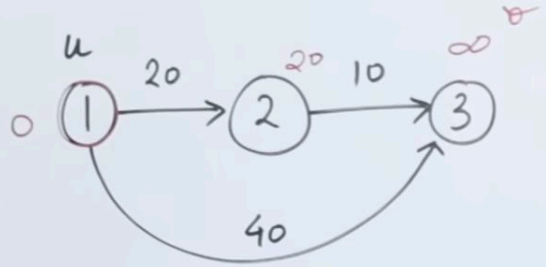
if $d(u) + c(u,v) < d(v)$

$d(v) = d(u) + c(u,v)$

$d(u) + c(u,v) \quad d(v)$

$0 + 20 \quad < \quad \infty$

"Dijkstra's Algorithm" (Single Source Shortest Path)



* Relaxation

if $d(u) + c(u,v) < d(v)$

$d(v) = d(u) + c(u,v)$

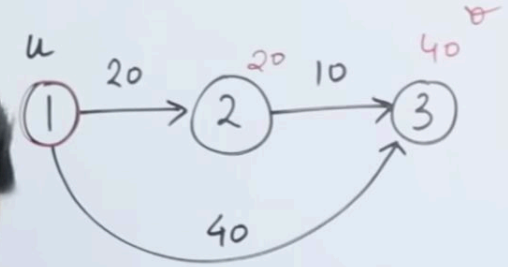
$d(u) + c(u,v)$ $\quad d(v)$

$0 + 20 < \infty$

$0 + 40 < \infty$

$40 < \infty$

"Dijkstra's Algorithm" (Single Source Shortest Path)



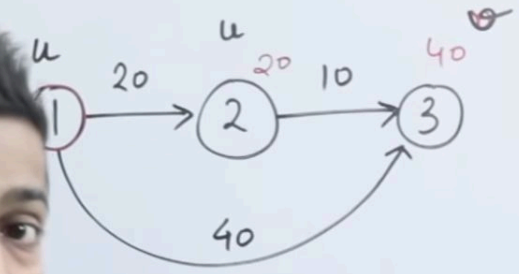
* Relaxation

if $d(u) + c(u,v) < d(v)$

$d(v) = d(u) + c(u,v)$

$d(u) + c(u,v)$		$d(v)$
$0 + 20$	$<$	∞
$0 + 40$	$<$	∞
40	$<$	∞

"Dijkstra's Algorithm" (Single Source Shortest Path)



* Relaxation

if $d(u) + c(u,v) < d(v)$

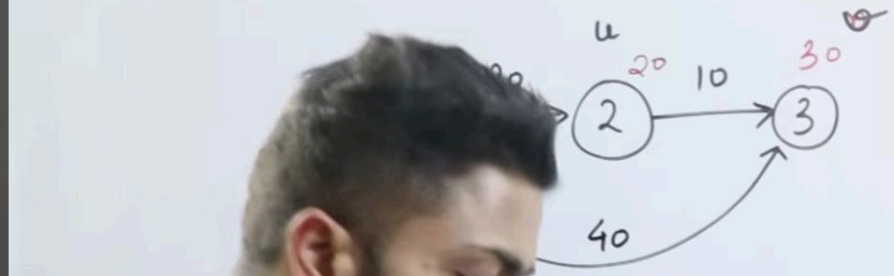
$d(v) = d(u) + c(u,v)$

$d(u) + c(u,v)$		$d(v)$
$0 + 20$	$<$	∞
$0 + 40$	$<$	∞
$40 < \infty$		

$20 + 10 < 40$

$30 < 40$

"Dijkstra's Algorithm" (Single Source Shortest Path)



* Relaxation

if $d(u) + c(u,v) < d(v)$

$d(v) = d(u) + c(u,v)$

$d(u) + c(u,v)$ $\quad d(v)$

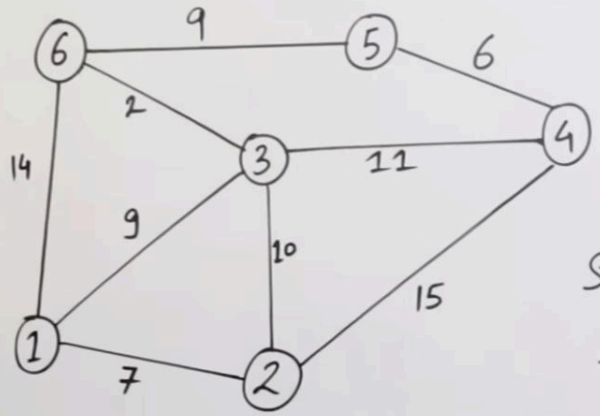
$0 + 20 < \infty$

$0 + 40 < \infty$

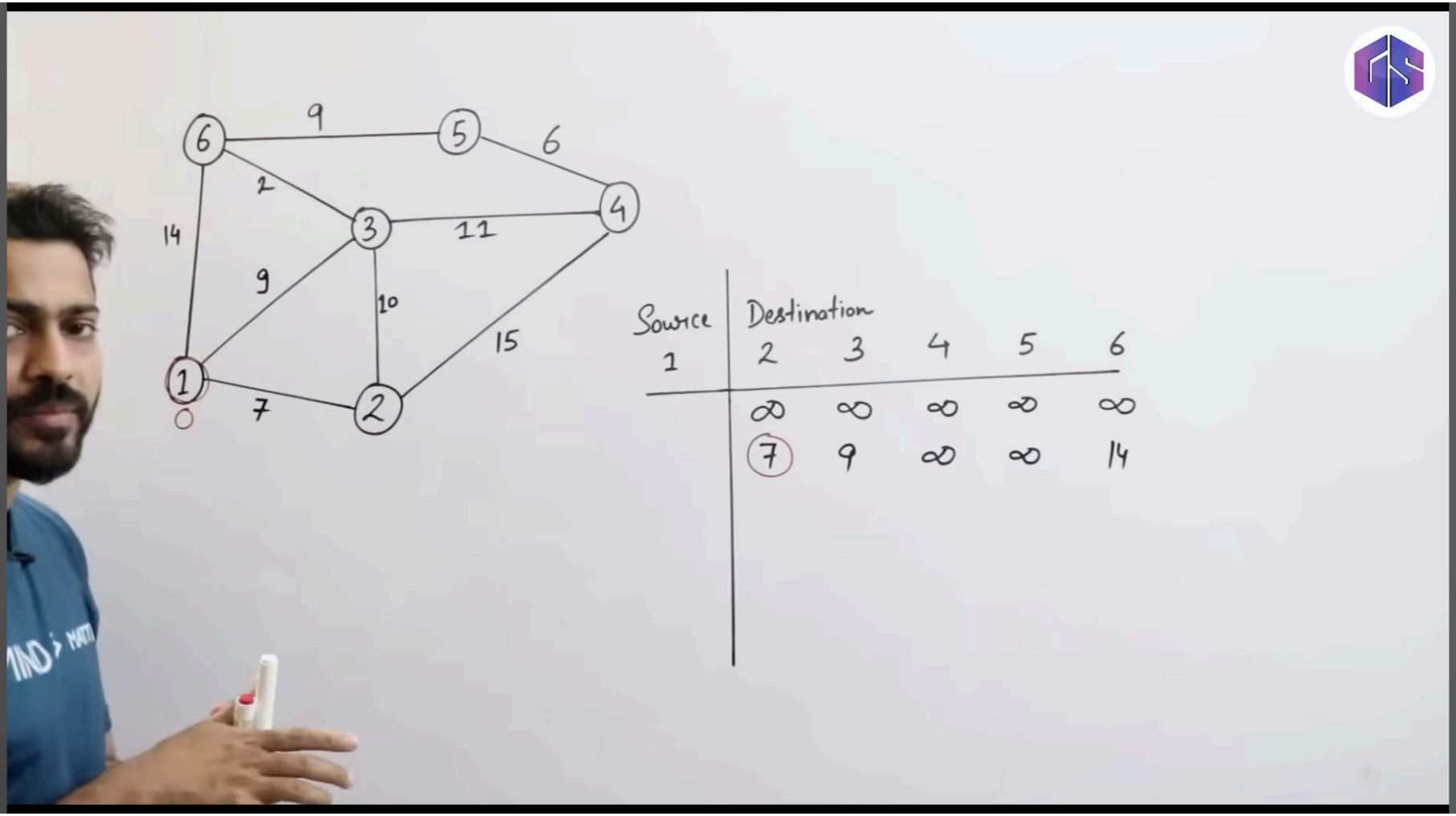
$40 < \infty$

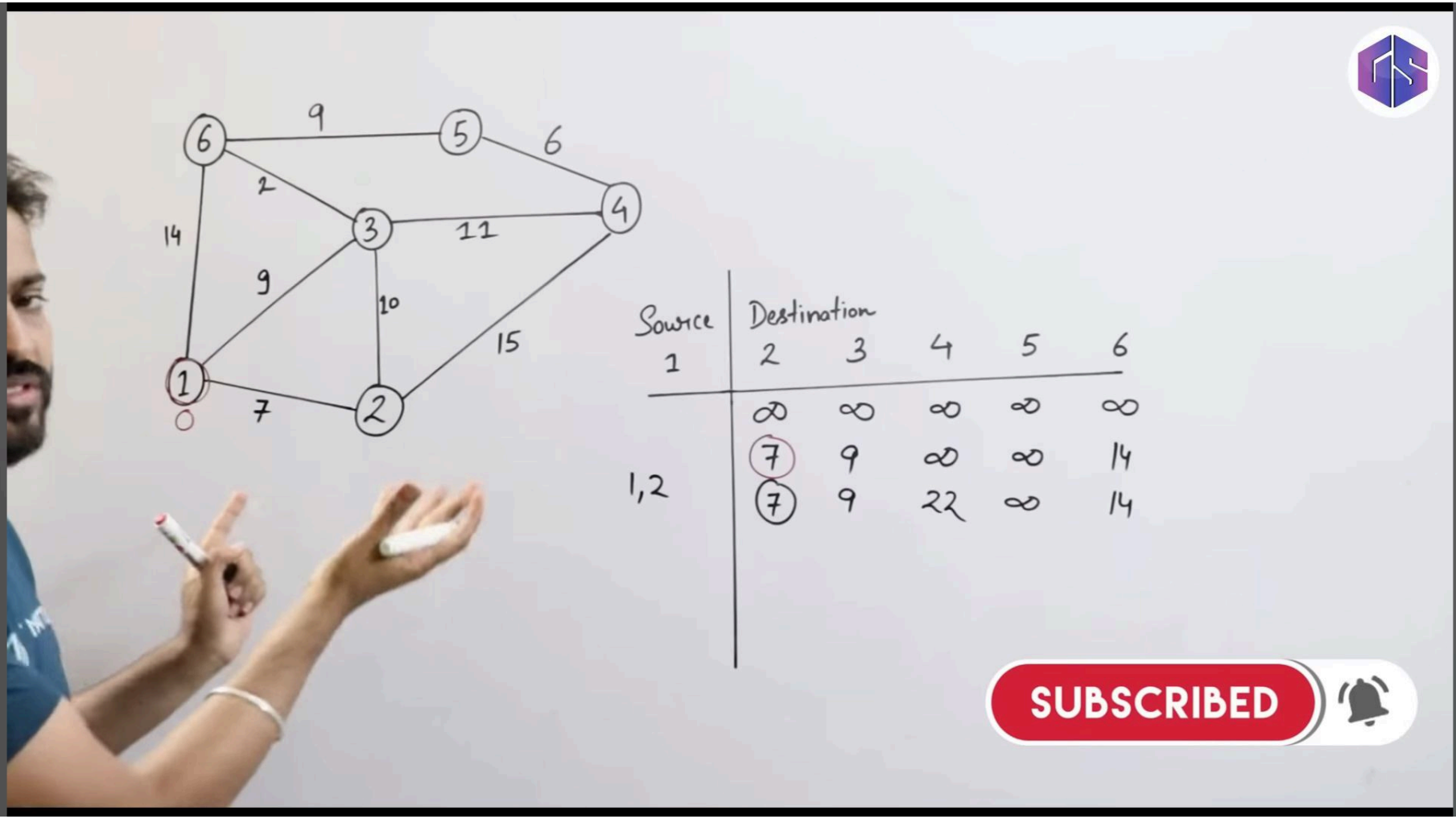
$20 + 10 < 40$

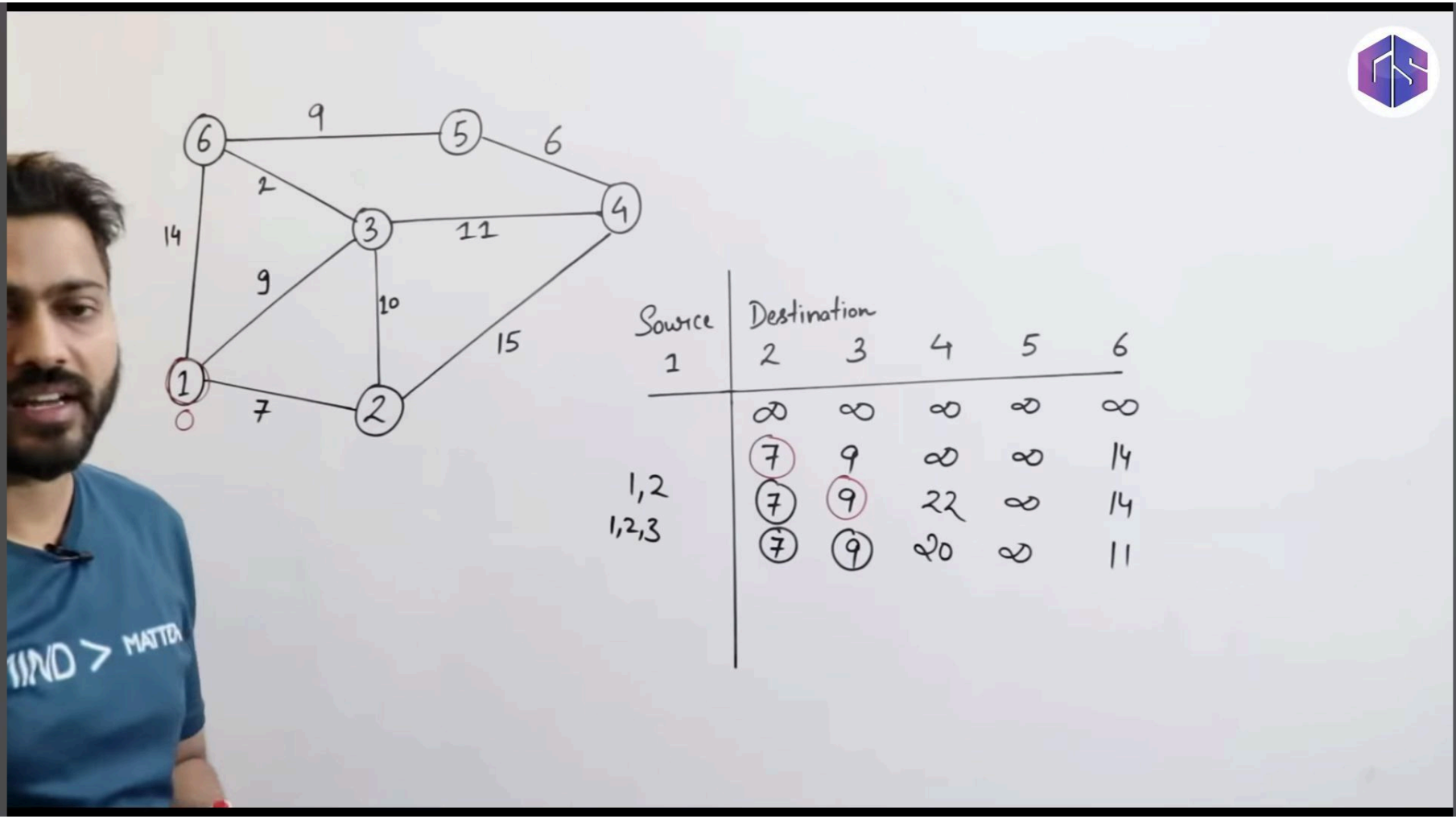
$30 < 40$

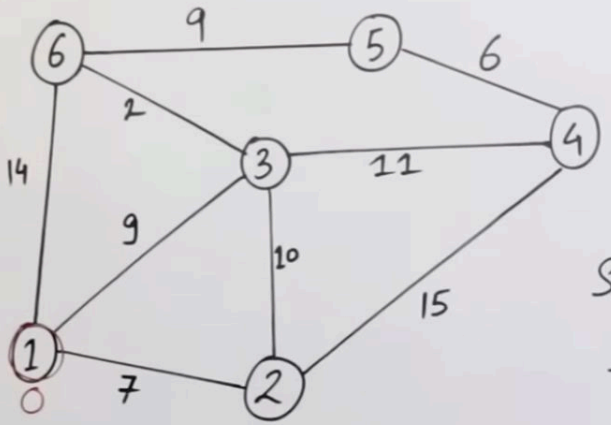


Source	Destination				
1	2	3	4	5	6


6
9
5
6
2
4
14
3
11
9
10
15
1
7
2
Source
Destination
1
2
3
4
5
6
∞ ∞ ∞ ∞ ∞
7 9 ∞ ∞ 14
SUBSCRIBE

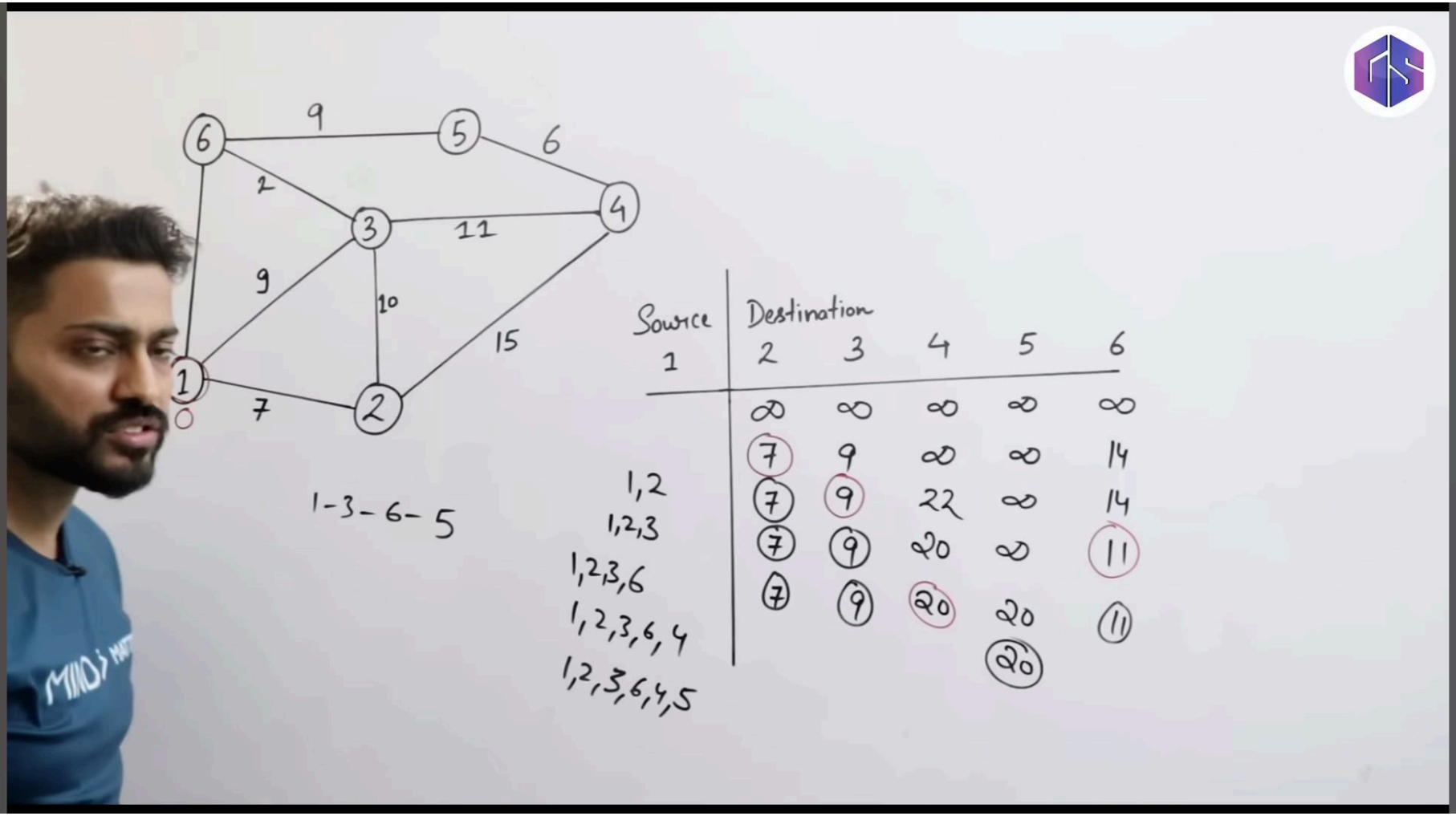

6
5
4
3
2
1
9
6
2
14
11
9
10
15
7
0
Source
Destination
1
2
3
4
5
6
∞ ∞ ∞ ∞ ∞
7 9 ∞ ∞ 14
1,2
7 9 22 ∞ 14
SUBSCRIBED
SUBSCRIBE

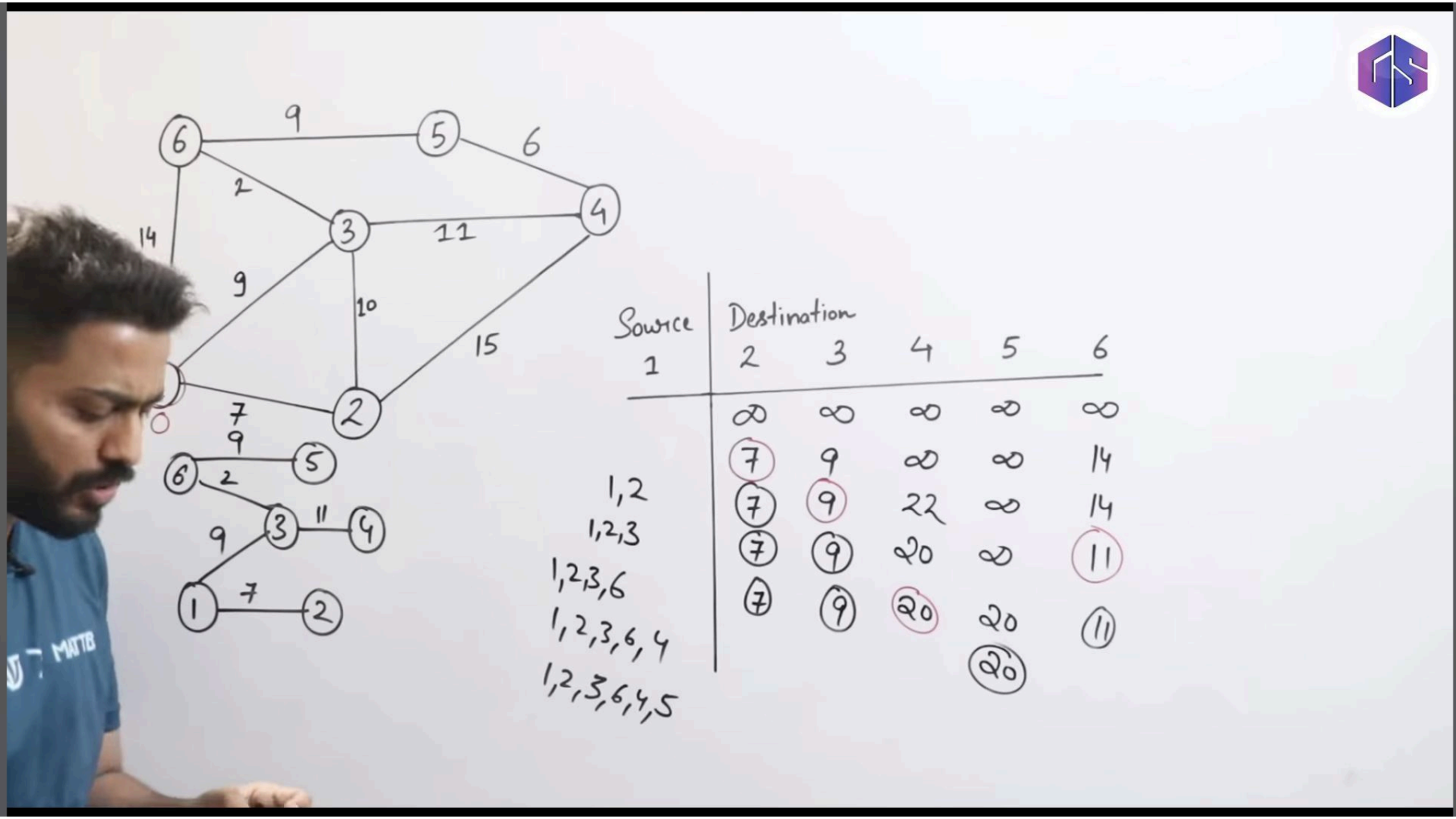

6
9
5
6
2
14
3
11
4
9
10
15
1
7
2
Source
Destination
1
2
3
4
5
6
∞ ∞ ∞ ∞ ∞
7 9 ∞ ∞ 14
1,2
7 9 22 ∞ 14
1,2,3
7 9 20 ∞ 11
SUBSCRIBE



1-3-6-5

Source 1	Destination 2	3	4	5	6
	∞	∞	∞	∞	∞
1,2	7	9	∞	∞	14
1,2,3	7	9	22	∞	14
1,2,3,6	7	9	20	∞	11
	7	9	20	20	11


6
9
5
6
2
3
11
4
9
10
15
1
7
2
0
1-3-6-5
Source
1
Destination
2
3
4
5
6
∞ ∞ ∞ ∞ ∞
7 9 ∞ ∞ 14
7 9 22 ∞ 14
7 9 20 ∞ 11
7 9 20 20 11
20
1,2
1,2,3
1,2,3,6
1,2,3,6,4
1,2,3,6,4,5
SUBSCRIBE


6
9
5
6
2
4
14
3
11
9
10
15
7
2
9
6
5
2
3
11
4
9
7
1
2
MITB
Source
Destination
1
2
3
4
5
6
∞ ∞ ∞ ∞ ∞
7 9 ∞ ∞ 14
7 9 22 ∞ 14
7 9 20 ∞ 11
7 9 20 20 11
20
1,2
1,2,3
1,2,3,6
1,2,3,6,4
1,2,3,6,4,5
SUBSCRIBE

Knapsack Problem

Knapsack Problem

objects	ob_1	ob_2	ob_3
Profit	25	24	15
Weight	18	15	10

Knapsack Capacity(M) = 20

→ for i = 1 to n
  Calculate Profit/weight

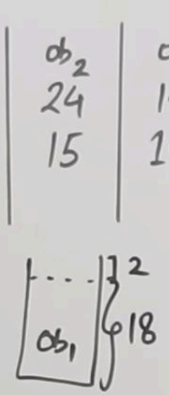
→ Sort objects in decreasing order of P/w Ratio

→ for i = 1 to n
  if M > 0 and $w_i \leq M$)
    $M = M - w_i$;
    $P = P + P_i$;
  else break;
if (M > 0)
  $P = P + P_i\left(\frac{M}{w_i}\right)$;

Knapsack Problem

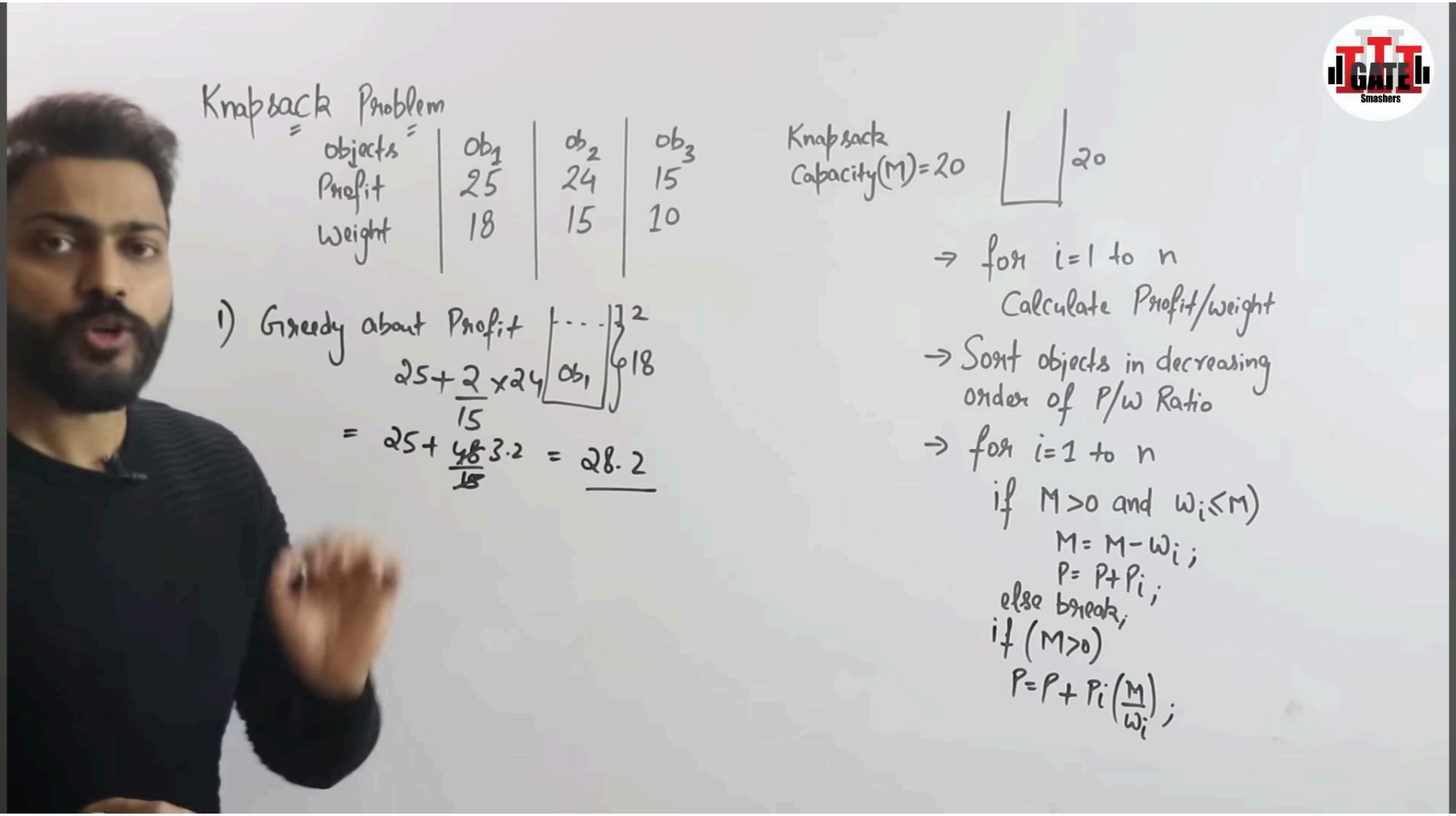
objects	ob_1	ob_2	ob_3
Profit	25	24	15
Weight	18	15	10

1) Greedy about Profit



Knapsack Capacity(M)=20

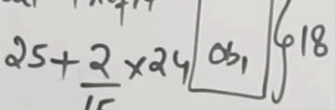
→ for i=1 to n
  Calculate Profit/weight
→ Sort objects in decreasing order of P/w Ratio
→ for i=1 to n
  if M>0 and $w_i \leq M$)
    $M = M - w_i$;
    $P = P + P_i$;
  else break;
if (M>0)
  $P = P + P_i\left(\frac{M}{w_i}\right)$;


Knapsack Problem
objects | ob_1 | ob_2 | ob_3
Profit | 25 | 24 | 15
Weight | 18 | 15 | 10
1) Greedy about Profit
$25+\frac{2}{15}\times 24$
$= 25+\frac{48}{15}3.2 = 28.2$
Knapsack Capacity(M)=20
20
→ for i=1 to n
Calculate Profit/weight
→ Sort objects in decreasing order of P/w Ratio
→ for i=1 to n
if M>0 and $w_i \leq M$)
$M = M - w_i$;
$P = P + P_i$;
else break;
if (M>0)
$P = P + P_i\left(\frac{M}{w_i}\right)$;
GATE Smashers
SUBSCRIBE

Knapsack Problem

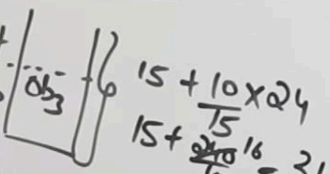
objects	ob_1	ob_2	ob_3
Profit	25	24	15
weight	18	15	10

1) Greedy about Profit



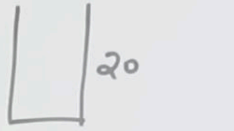
$$25 + \frac{2}{15} \times 24 = 25 + \frac{48}{15} 3.2 = 28.2$$

2) Greedy about weight



$$15 + \frac{10}{15} \times 24 \quad 15 + \frac{240}{15}^{16} = 31$$

Knapsack Capacity(M)=20

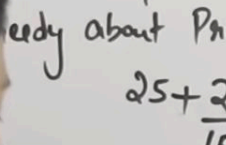


→ for i=1 to n
  Calculate Profit/weight
→ Sort objects in decreasing order of P/w Ratio
→ for i=1 to n
  if M>0 and $w_i \leq M$)
    M= M−w_i;
    P= P+P_i;
  else break;
  if (M>0)
    P=P+P_i($\frac{M}{w_i}$);

Knapsack Problem

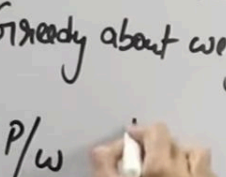
objects	ob_1	ob_2	ob_3
Profit	25	24	15
weight	18	15	10
P/w	1.3	1.6	1.5

Greedy about Profit



$$25 + \frac{2}{15} \times 24 = 25 + 3.2 = 28.2$$

Greedy about weight



$$15 + \frac{10}{15} \times 24 = 15 + 16 = 31$$

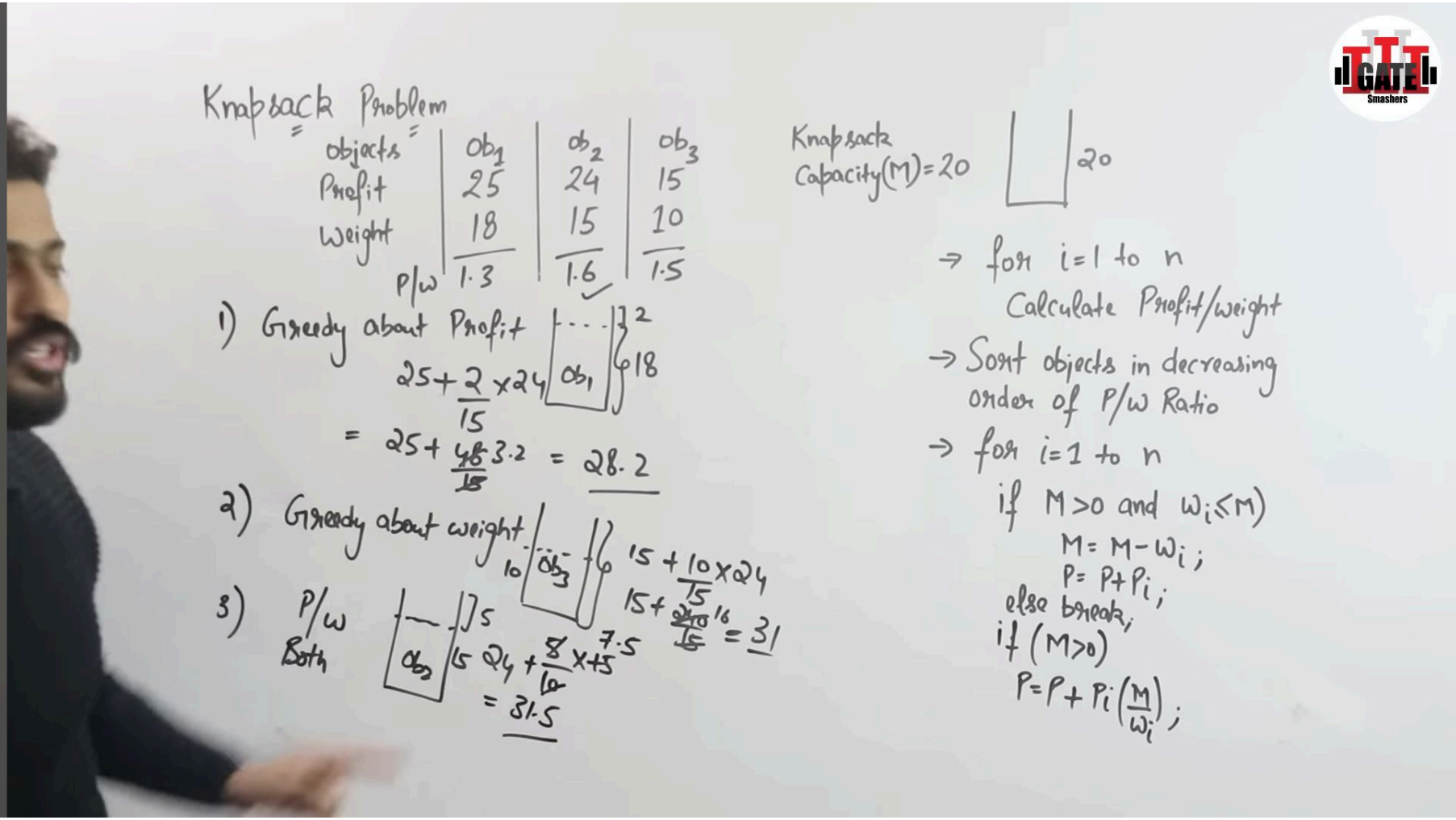
P/w

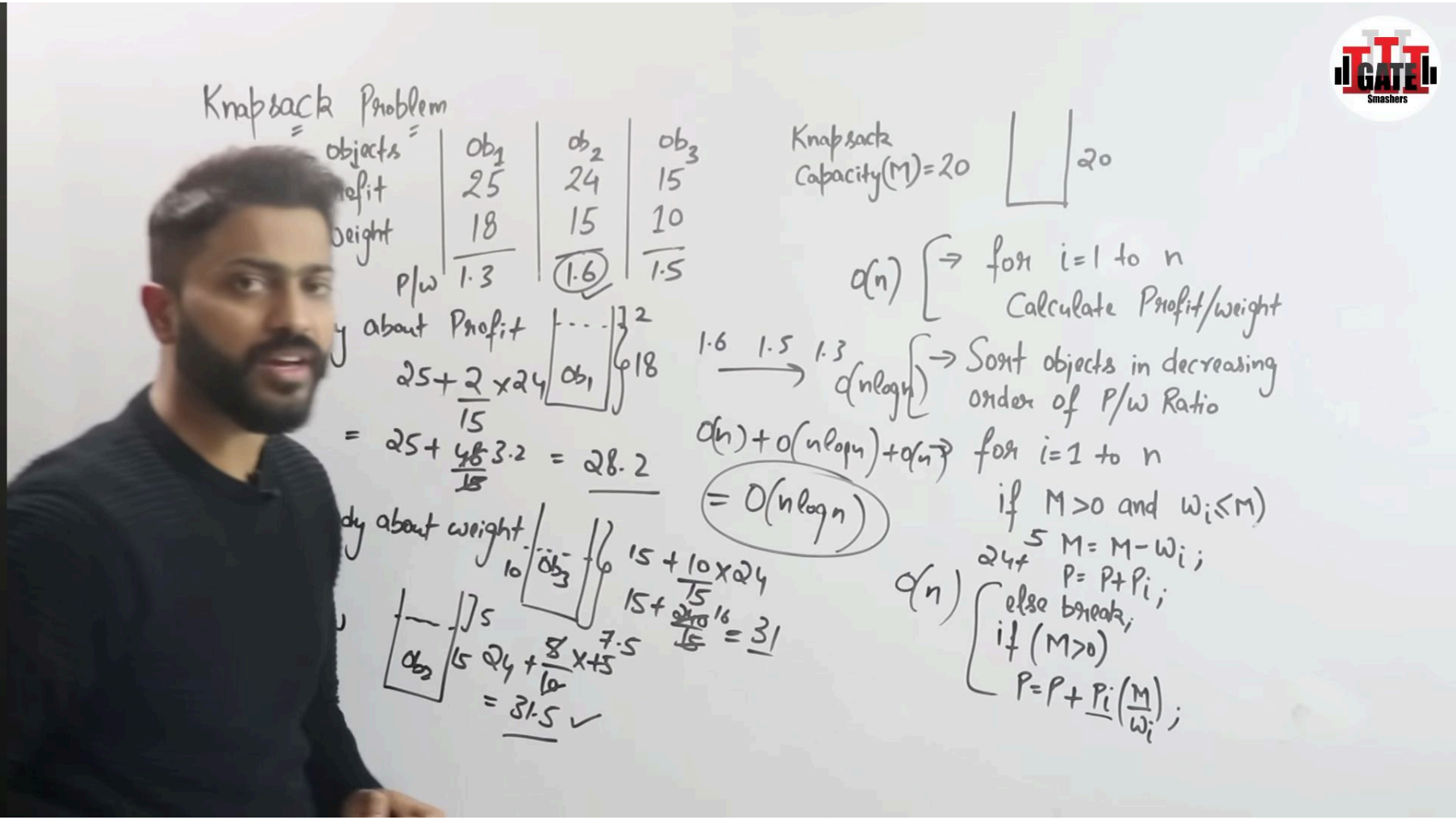
Knapsack Capacity(M) = 20

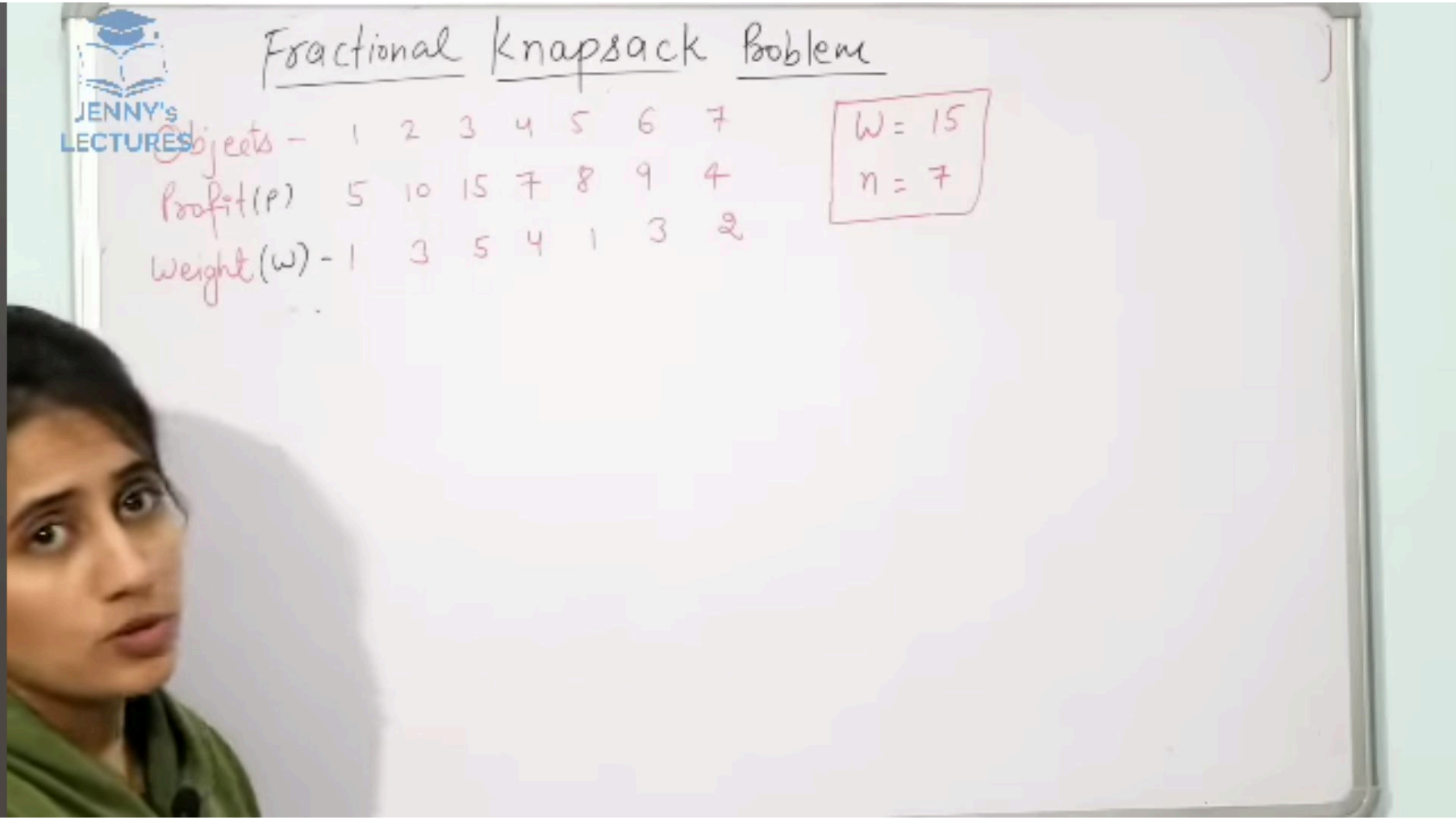
→ for i=1 to n
  Calculate Profit/weight

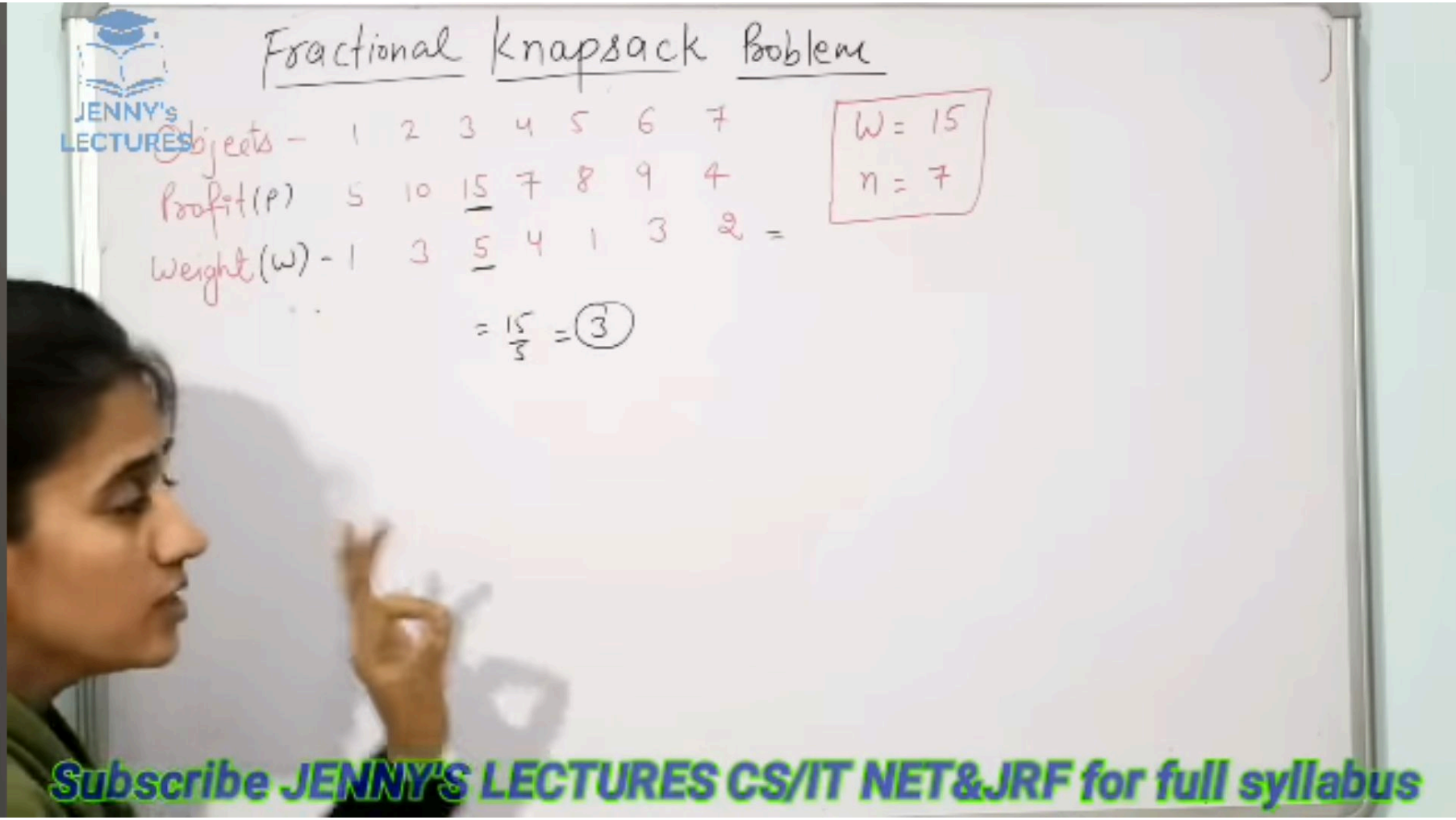
→ Sort objects in decreasing order of P/w Ratio

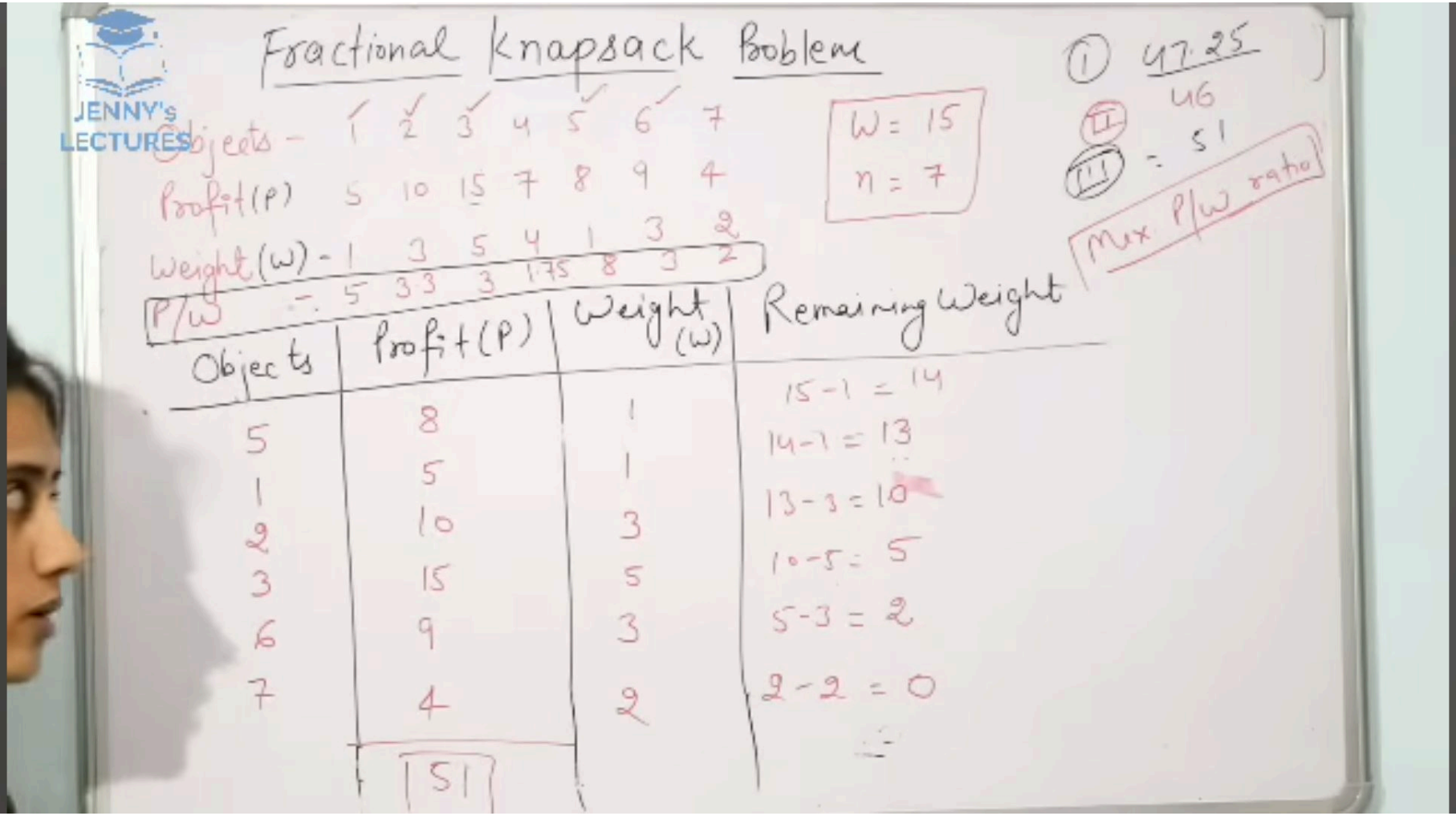
→ for i=1 to n
  if M>0 and $w_i \leq M$)
    M = M − w_i;
    P = P + P_i;
  else break;
if (M>0)
  $P = P + P_i\left(\frac{M}{w_i}\right)$;


Knapsack Problem
objects | ob_1 | ob_2 | ob_3
Profit | 25 | 24 | 15
weight | 18 | 15 | 10
P/w | 1.3 | 1.6 | 1.5
1) Greedy about Profit
$25+\frac{2}{15}\times 24 = 25+\frac{48}{15} = 25+3.2 = 28.2$
2) Greedy about weight
$15+\frac{10}{15}\times 24 = 15+16 = 31$
3) P/w Both
$24+\frac{8}{10}\times 15 = 24+12 = 31.5$
Knapsack Capacity(M)=20
20
→ for i=1 to n
Calculate Profit/weight
→ Sort objects in decreasing order of P/w Ratio
→ for i=1 to n
if M>0 and $w_i \le M$)
$M = M - w_i$;
$P = P + P_i$;
else break;
if (M>0)
$P = P + P_i\left(\frac{M}{w_i}\right)$;


Knapsack Problem
objects | ob1 | ob2 | ob3
Profit | 25 | 24 | 15
Weight | 18 | 15 | 10
P/w | 1.3 | 1.6 | 1.5
Knapsack Capacity(M)=20
20
1.6 1.5 1.3
O(n) for i=1 to n
Calculate Profit/weight
O(nlogn) Sort objects in decreasing order of P/w Ratio
O(n)+O(nlogn)+O(n)
= O(nlogn)
for i=1 to n
if M>0 and wi≤M)
M= M-wi;
P= P+Pi;
else break;
if (M>0)
P=P+Pi(M/wi);
O(n)
y about Profit
25+2/15 x24
= 25+48/15 3.2 = 28.2
dy about weight
15+10/15 x24
15+24 = 31
24+8/10 x15
= 31.5 ✓


Fractional knapsack Poblem
JENNY's LECTURES
Objects - 1 2 3 4 5 6 7
Profit(P) 5 10 15 7 8 9 4
Weight(W) - 1 3 5 4 1 3 2
W = 15
n = 7


Fractional Knapsack Problem
JENNY's LECTURES
Objects - 1 2 3 4 5 6 7
Profit(P) 5 10 15 7 8 9 4
Weight(W) - 1 3 5 4 1 3 2
W = 15
n = 7
= 15/5 = 3
Subscribe JENNY'S LECTURES CS/IT NET&JRF for full syllabus


JENNY's LECTURES
Fractional Knapsack Problem
Objects - 1 2 3 4 5 6 7
Profit(P) 5 10 15 7 8 9 4
Weight(W) - 1 3 5 4 1 3 2
P/W = 5 3.3 3 1.75 8 3 2
W = 15
n = 7
Objects | Profit(P) | Weight (W) | Remaining Weight
5 | 8 | 1 | 15-1 = 14
1 | 5 | 1 | 14-1 = 13
2 | 10 | 3 | 13-3 = 10
3 | 15 | 5 | 10-5 = 5
6 | 9 | 3 | 5-3 = 2
7 | 4 | 2 | 2-2 = 0
51
(I) 47.25
46
(II)
(III) = 51
Max P/W ratio

Job sequencing with deadline

Job Sequencing (Greedy Technique)

	J_1	J_2	J_3	J_4
Profit	50	15	10	25
Deadline	2	1	2	1

Algorithm:

1) Arrange All jobs in decreasing order of Profit.
2) For each job(m_i), do linear search to find particular slot in array of size (n), where n = maximum deadline
   m = total jobs

GATE Smashers

SUBSCRIBE

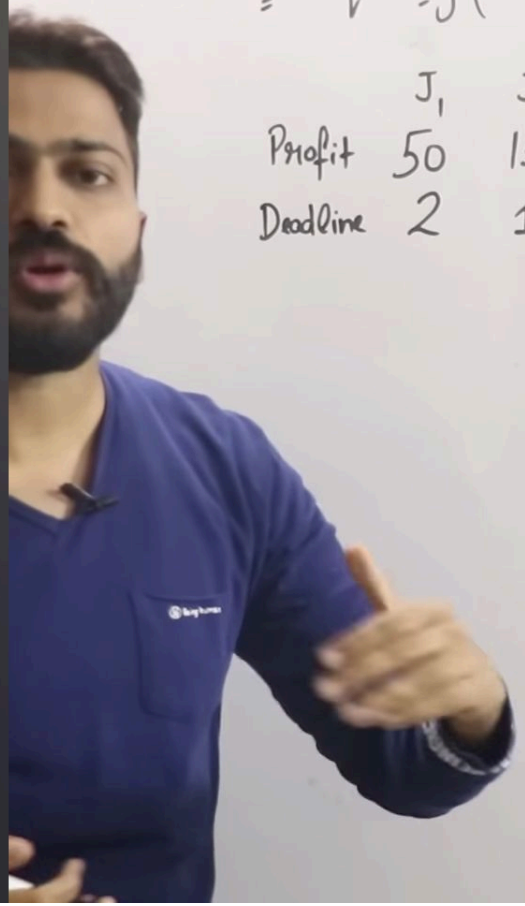
Job Sequencing (Greedy Technique)

	J_1	J_2	J_3	J_4
Profit	50	15	10	25
Deadline	2	1	2	1

(Uniprocessor, No Preempt!, 1 Unit of time)

Algorithm:

1) Arrange All jobs in decreasing order of Profit.
2) For each job(m_i), do linear search to find particular slot in array of size (n), where n = maximum deadline, m = total jobs



Job Sequencing (Greedy Technique)

	J_1	J_2	J_3	J_4
Profit	50	15	10	25
Deadline	2	1	2	1

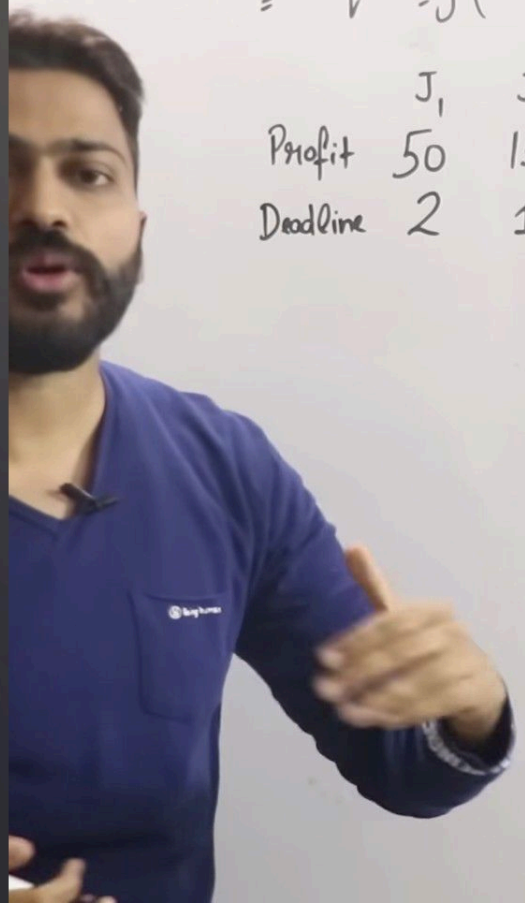
(Uniprocessor, No Preempt!, 1 Unit of time)

Algorithm:

1) Arrange All jobs in decreasing order of Profit.
2) For each job(m_i), do linear search to find particular slot in array of size (n), where n = maximum deadline, m = total jobs

The deadline here does not mean that, the job will take 2 months to complete, rather it means, that we can schedule the job in either 1st month or 2nd month.

Because as mentioned each job takes only 1 unit(month) of time



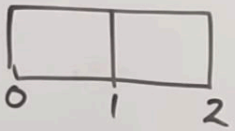
Job Sequencing (Greedy Technique)

	J_1	J_2	J_3	J_4
Profit	50	15	10	25
Deadline	2	1	2	1

(Uniprocessor, No Preempt!, 1 Unit of time)

Algorithm:

1) Arrange All jobs in decreasing order of Profit.
2) For each job(m_i), do linear search to find particular slot in array of size (n), where n = maximum deadline, m = total jobs



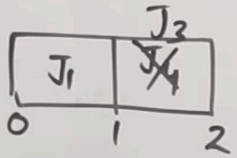
Job Sequencing (Greedy Technique)

	J_1	J_2	J_3	J_4
Profit	50	15	10	25
Deadline	2	1	2	1

(Uniprocessor, No Preempt! 1 Unit of time)

Algorithm:

1) Arrange All jobs in decreasing order of Profit.
2) For each job(m_i), do linear search to find particular slot in array of size (n), where n = maximum deadline
   m = total Jobs



50 + 10 = 60

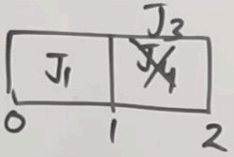
Job Sequencing (Greedy Technique)

	J_1	J_2	J_3	J_4
Profit	50	15	10	25
Deadline	2	1	2	1

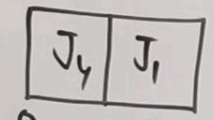
(Uniprocessor, No Preempt! 1 Unit of time)

Algorithm:

1) Arrange All jobs in decreasing order of Profit.
2) For each job(m_i), do linear search to find particular slot in array of size (n), where n = maximum deadline, m = total jobs



50 + 10 = 60



25 + 50 = 75 ✓

To get optimal solution, select the maximum profit and put it as far as its deadline allows, and repeat

Job Sequencing (Greedy Technique)

	J_1	J_2	J_3	J_4
Profit	50	15	10	25
Deadline	2	1	2	1

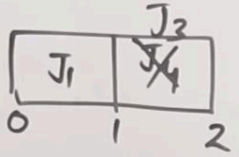
(Uniprocessor, No Preempt! 1 Unit of time)

Algorithm:

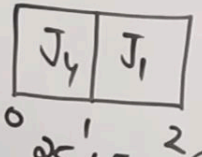
1) Arrange All jobs in decreasing order of Profit. $(n \log n)$
2) For each job(m_i), do linear search to find particular slot in array of size (n), where n = maximum deadline
   m = total Jobs

$m \; n$

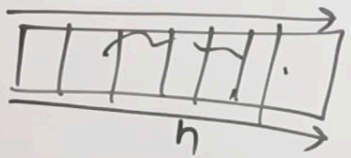
n^2 $O(n^2)$

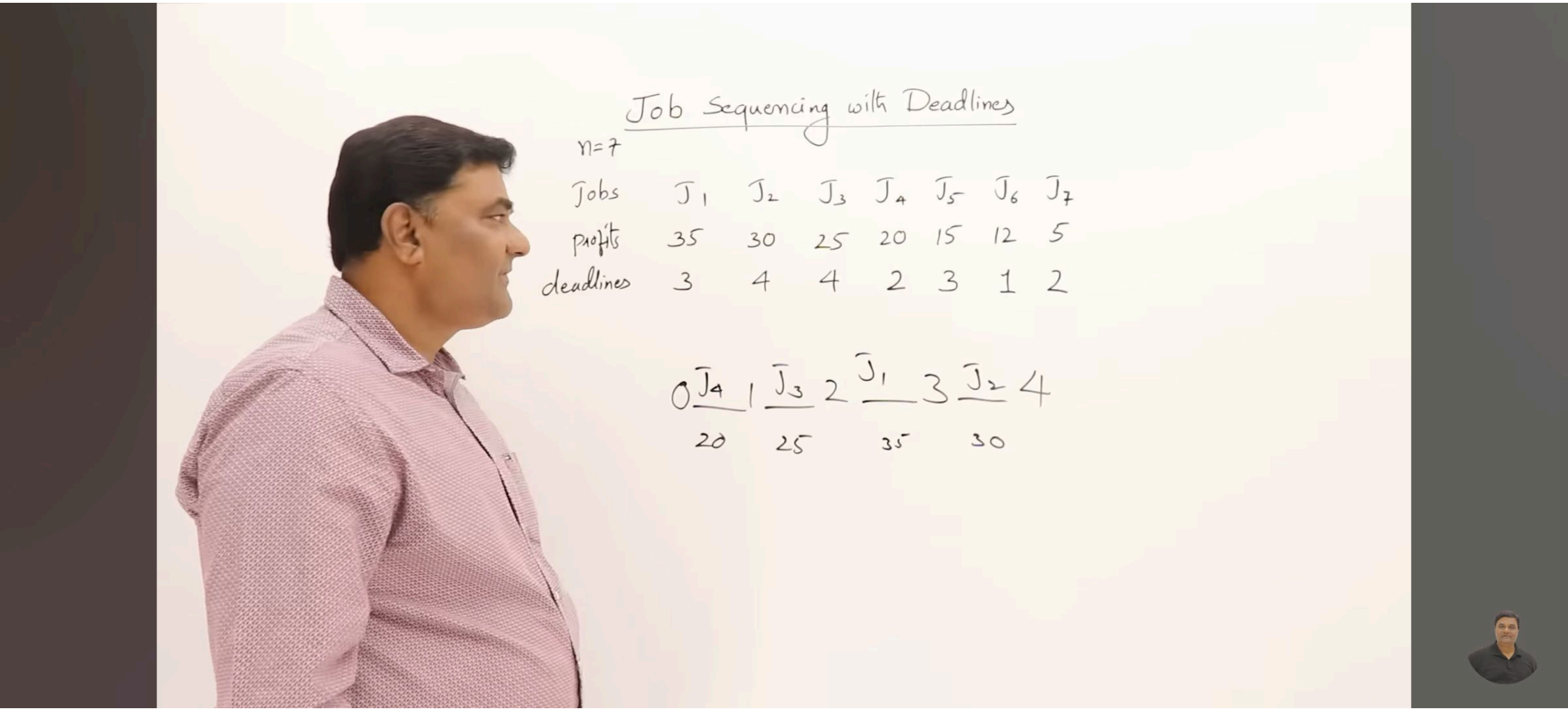


50 + 10 = 60



25 + 50 = 75 ✓

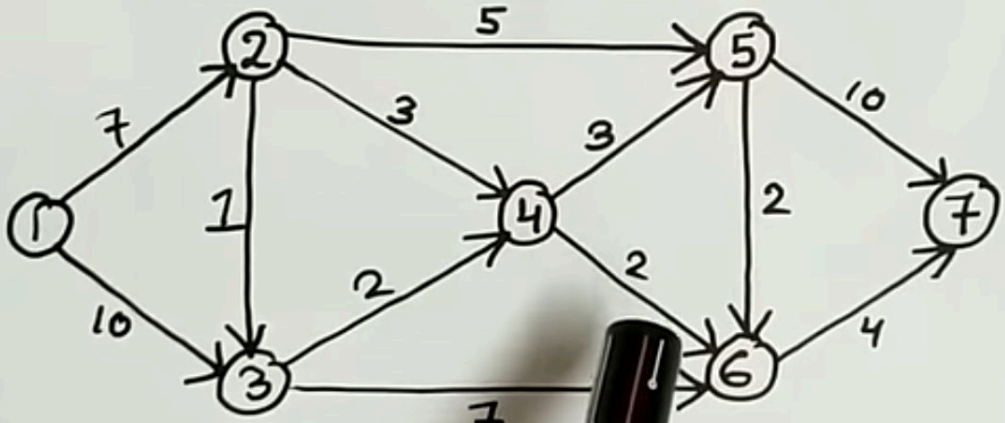



Job Sequencing with Deadlines
n=7
Jobs J1 J2 J3 J4 J5 J6 J7
profits 35 30 25 20 15 12 5
deadlines 3 4 4 2 3 1 2
0 J4 1 J3 2 J1 3 J2 4
20 25 35 30

Ford fulkerson method for max flow

Maximal Flow Problem (Ford Fulkerson Rule)

Q: Find the maximum flow through the given network using Ford Fulkerson algorithm.

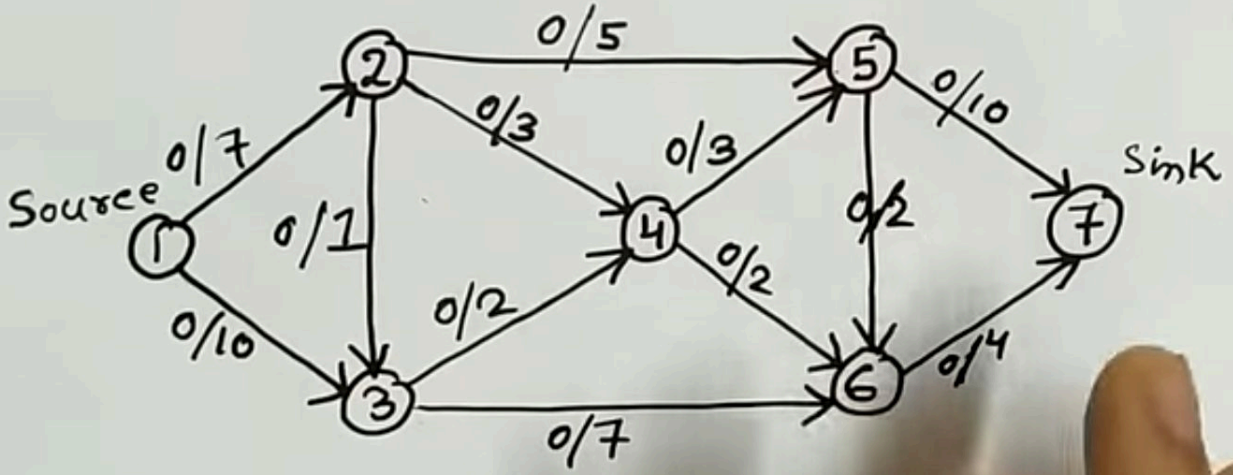


Please L

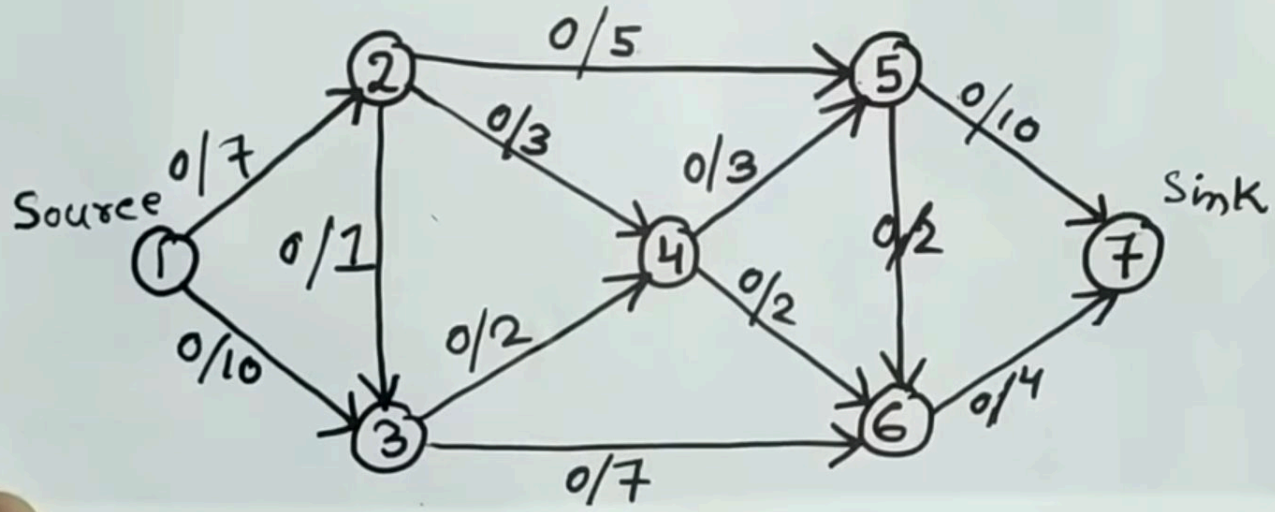
SUBSCRIBE

Maximal Flow Problem (Ford Fulkerson Rule)

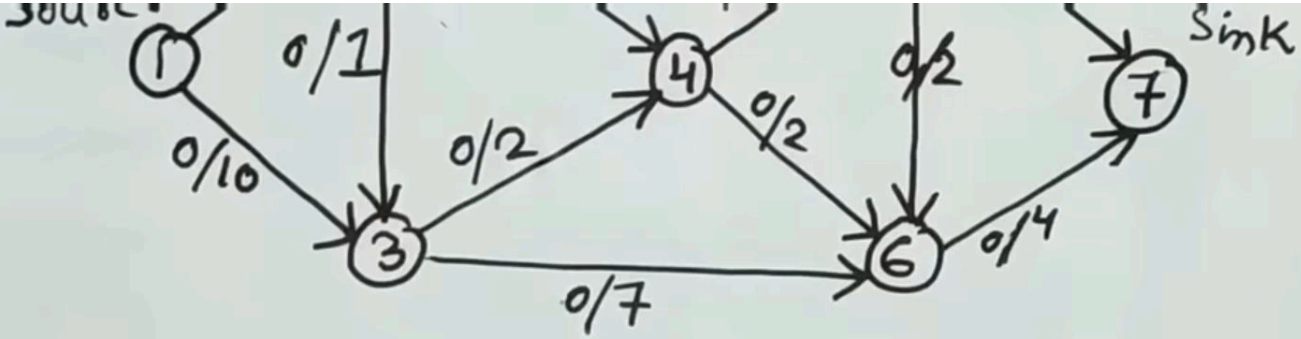
Q: Find the maximum flow through the given network using Ford Fulkerson algorithm.



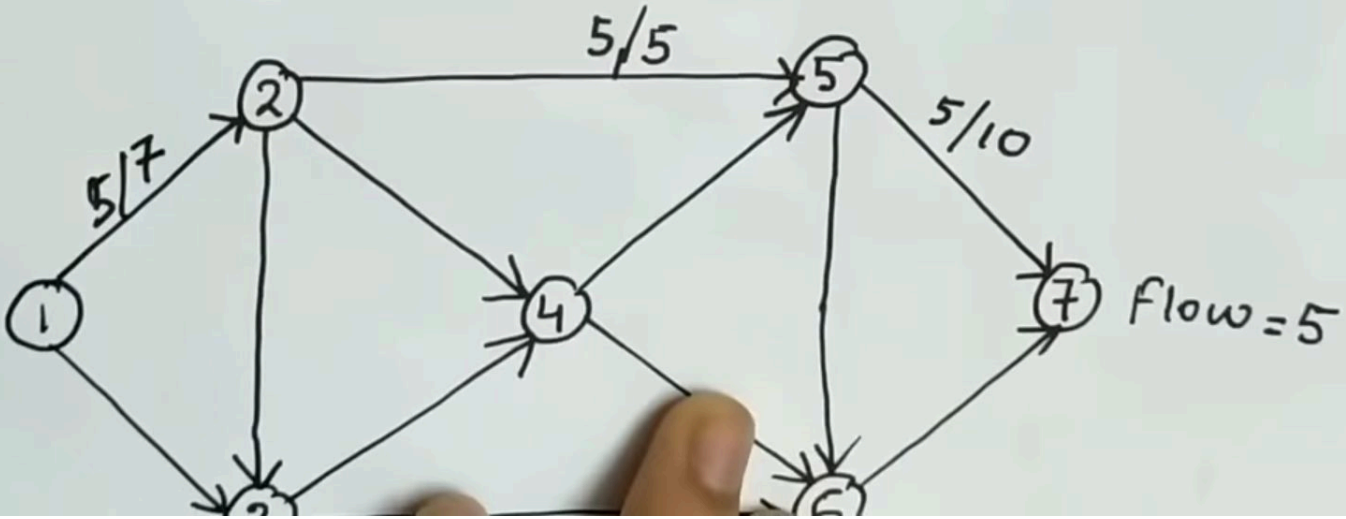
Solution :

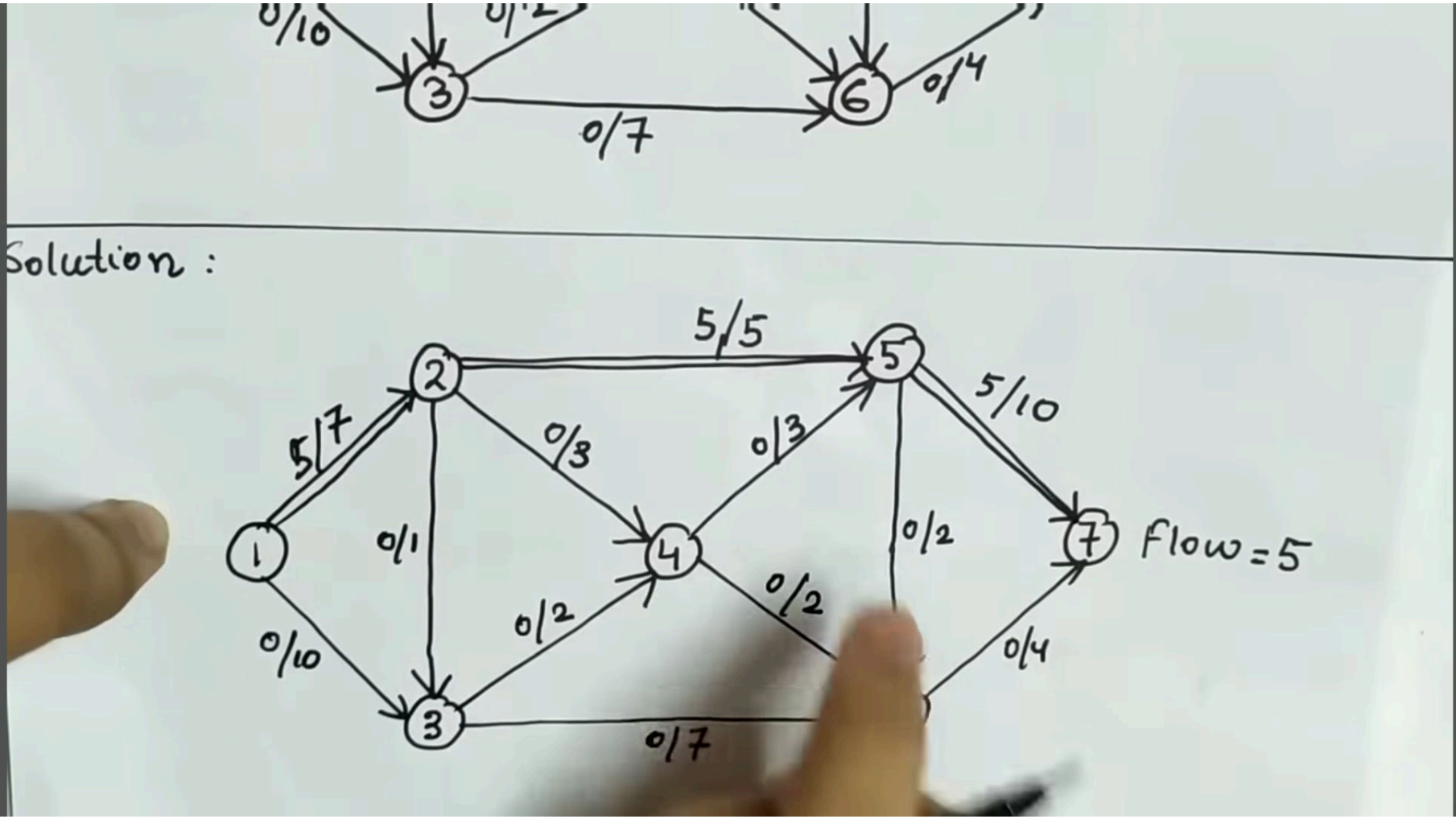


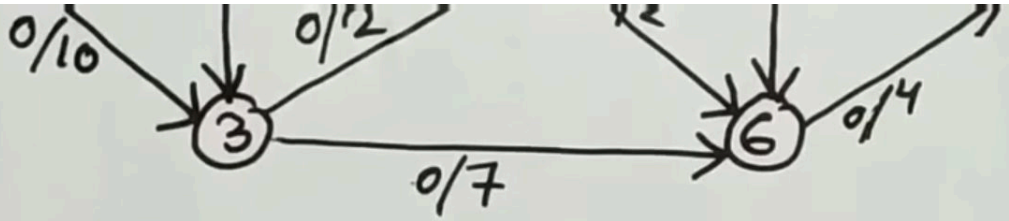
Augumenting path	Bottleneck capacity
1 - 2 - 5 - 7	5



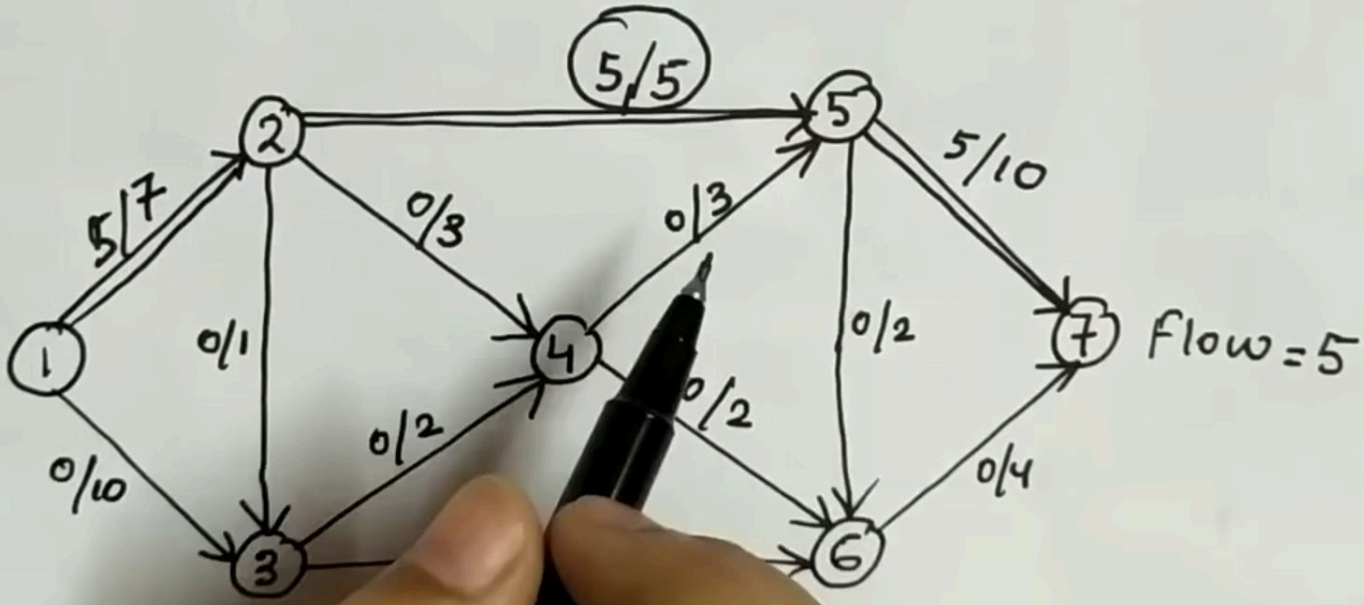
Solution :

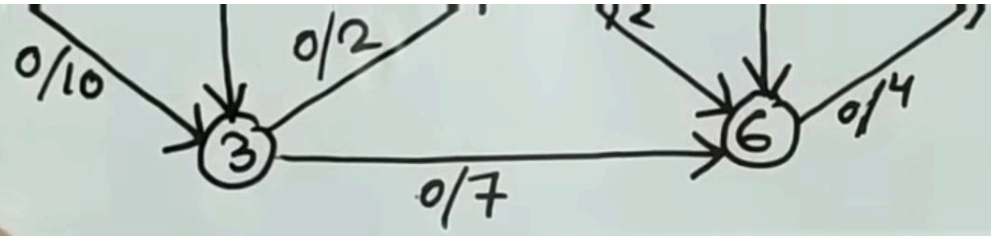



0/10
0/2
3
6
0/4
0/7
Solution :
5/5
2
5
5/7
5/10
0/3
0/3
1
0/1
4
0/2
7
Flow = 5
0/2
0/10
0/2
0/4
3
0/7
SUBSCRIBE

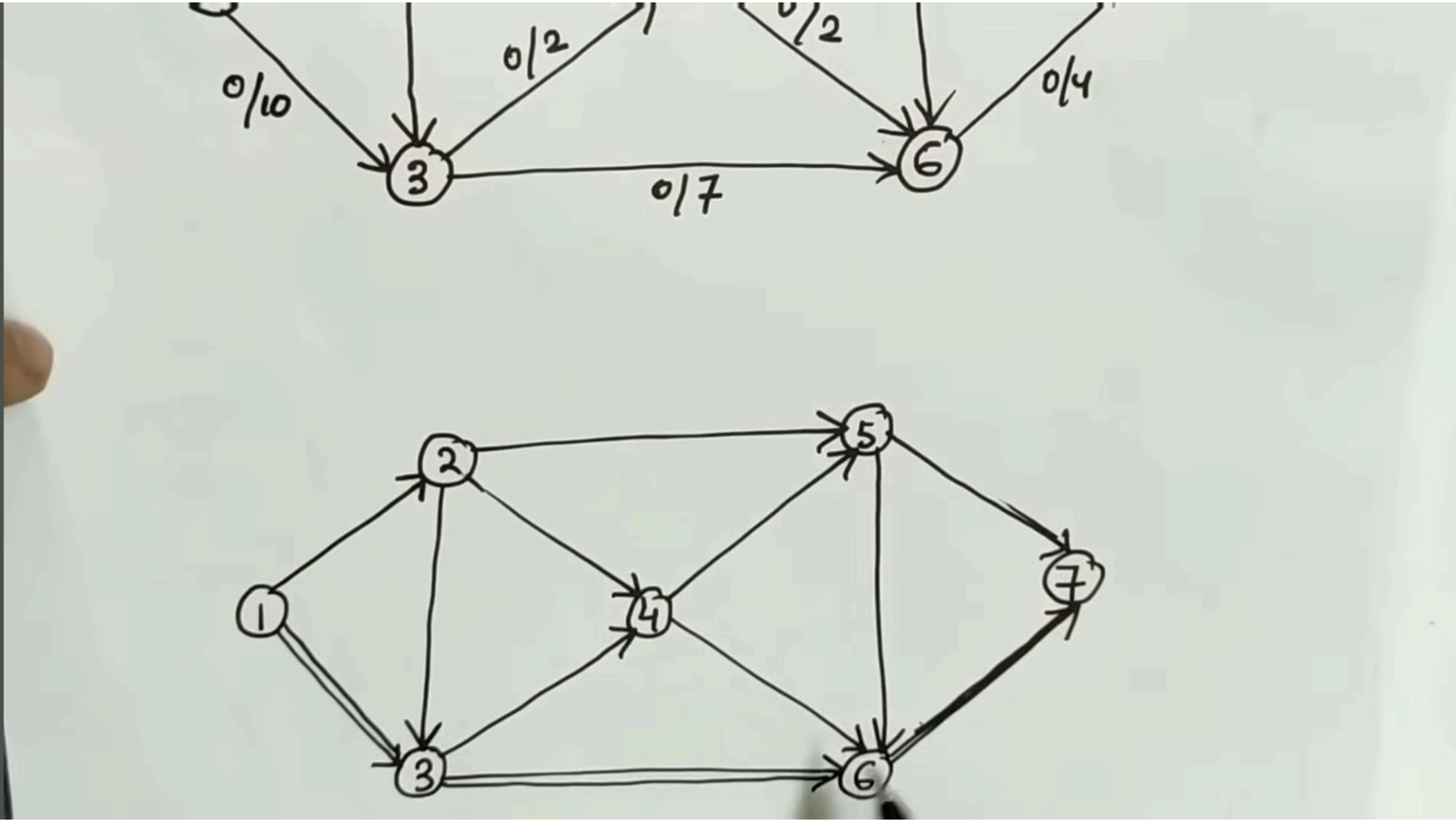


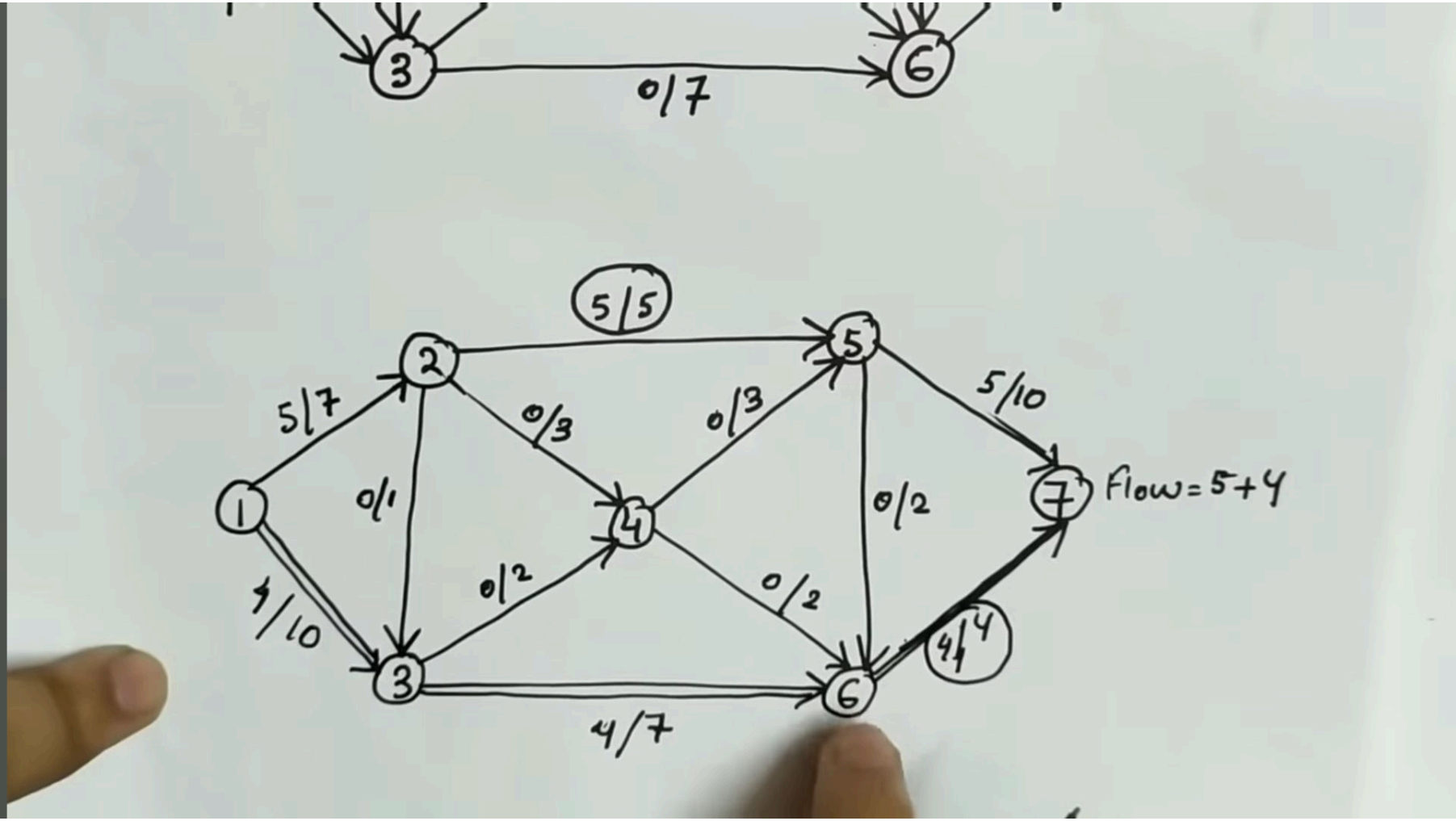
Solution :

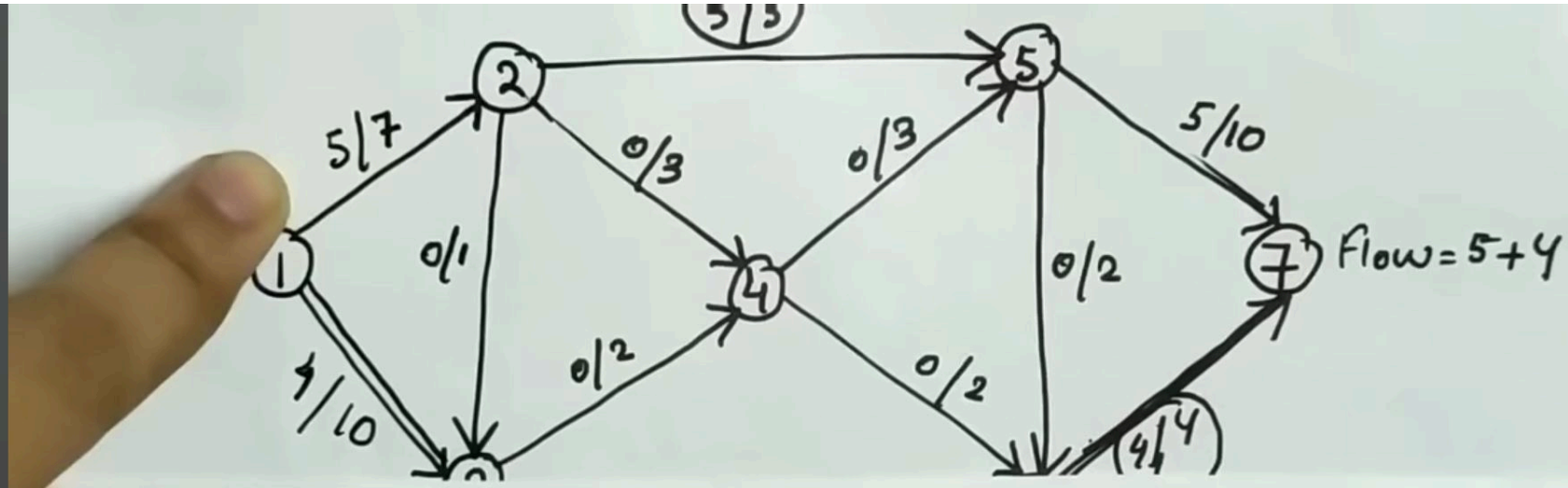




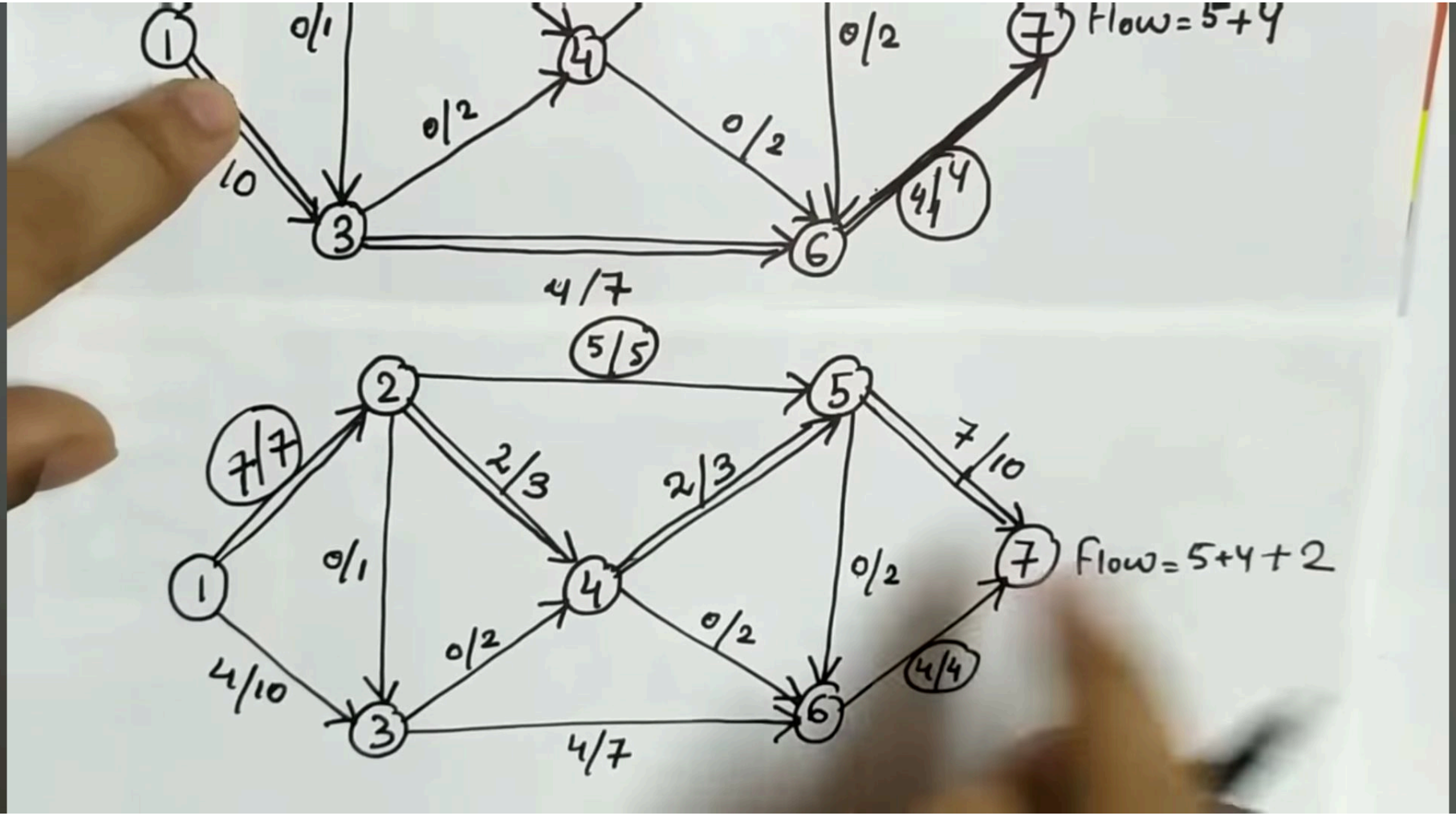
Augumenting path	Bottleneck capacity
1 - 2 - 5 - 7	5
1 - 3 - 6 - 7	4

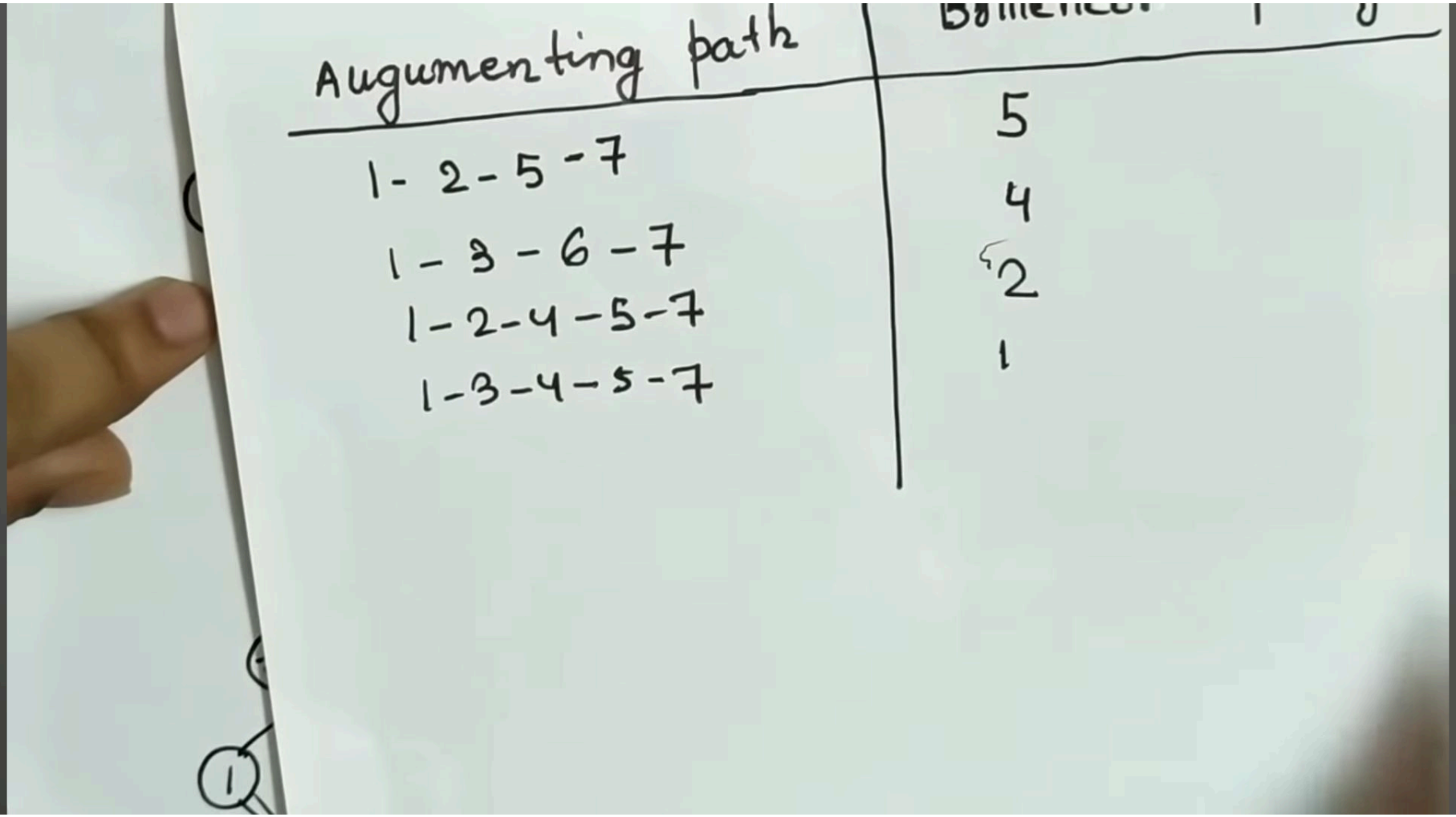

0/10
0/2
0/2
0/4
3
6
0/7
2
5
1
4
7
3
6

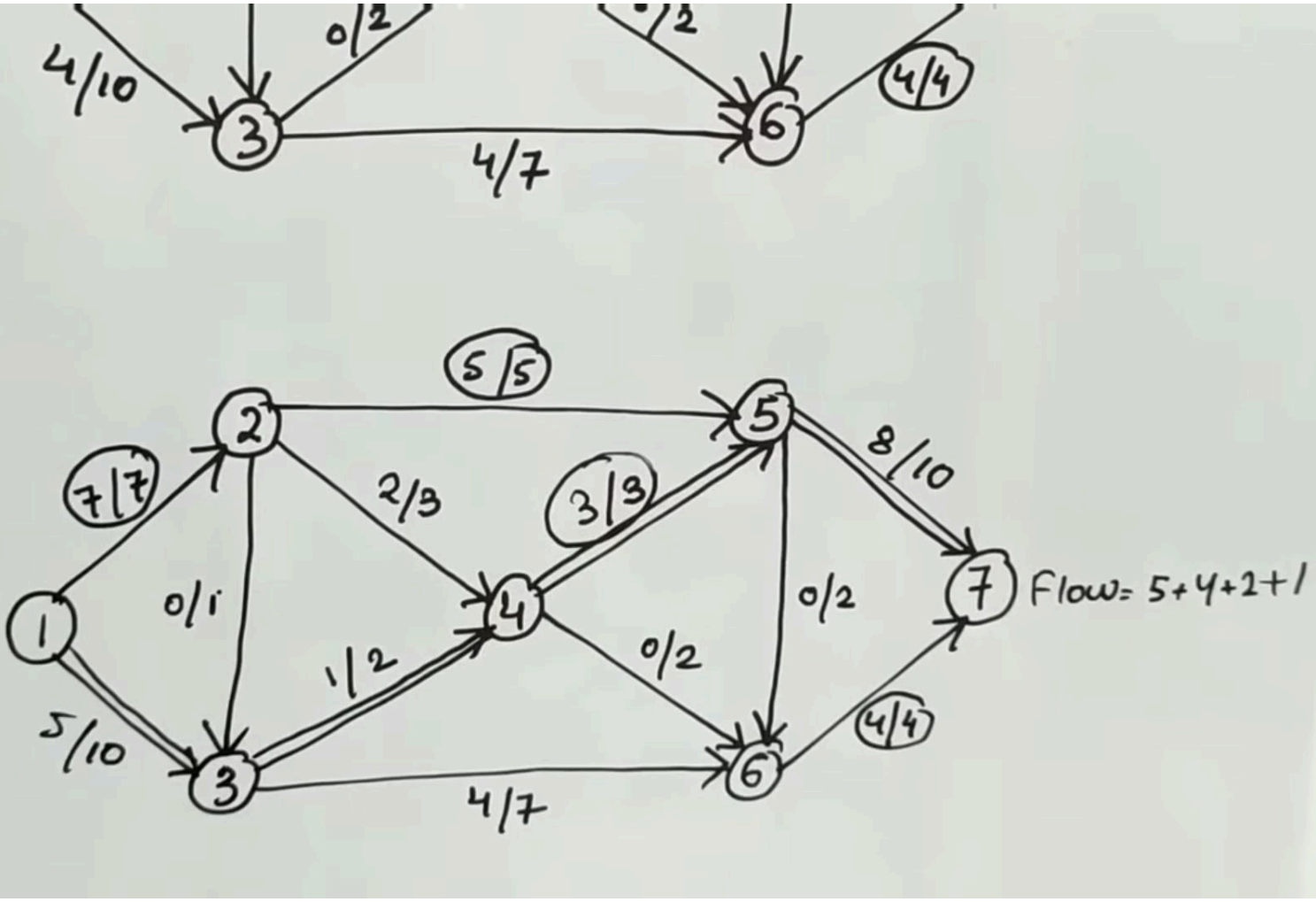

3
6
0/7
5/5
2
5
5/7
0/3
0/3
5/10
1
0/1
4
0/2
7
Flow=5+4
0/2
0/2
4/10
4/4
3
6
4/7



Augumenting path	Bottleneck capacity
1-2-5-7	5
1-3-6-7	4
1-2-4-5-7	2


1
0/1
4
0/2
7
Flow = 5+4
0/2
10
3
6
4/4
4/7
5/5
2
5
7/7
2/3
2/3
7/10
1
0/1
4
0/2
7
Flow = 5+4+2
0/2
4/10
3
6
4/4
4/7


Augumenting path
Bottleneck
1-2-5-7
5
1-3-6-7
4
1-2-4-5-7
2
1-3-4-5-7
1
SUBSCRIBE


4/10
0/2
4/7
4/4
5/5
7/7
2/3
3/3
8/10
0/1
0/2
1/2
0/2
5/10
4/7
4/4
Flow= 5+4+2+1

1- 2- 5 - 7	5
1 - 3 - 6 - 7	4
1 - 2 - 4 - 5 - 7	2
1 - 3 - 4 - 5 - 7	1

Maximum Flow through the given network using Ford Fulkerson method is 5+4+2+1