Greedy algorithms:-

P - NP - NPH
NPC.

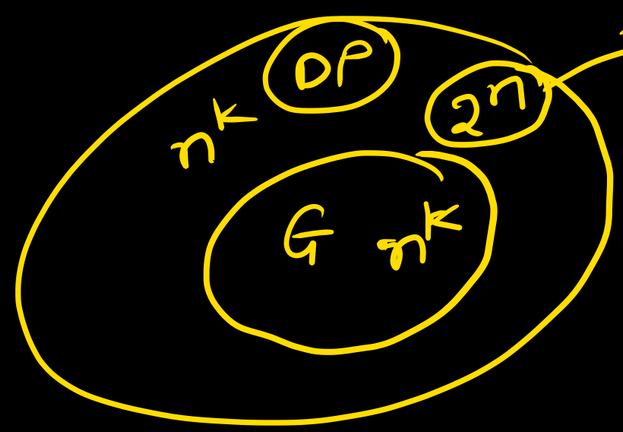
↓
not in syllabus.

Optimization problems:

→ min cost
→ max profit
→ min delay
→ max reliability
→ min risk.

Greedy

Dynamic programming

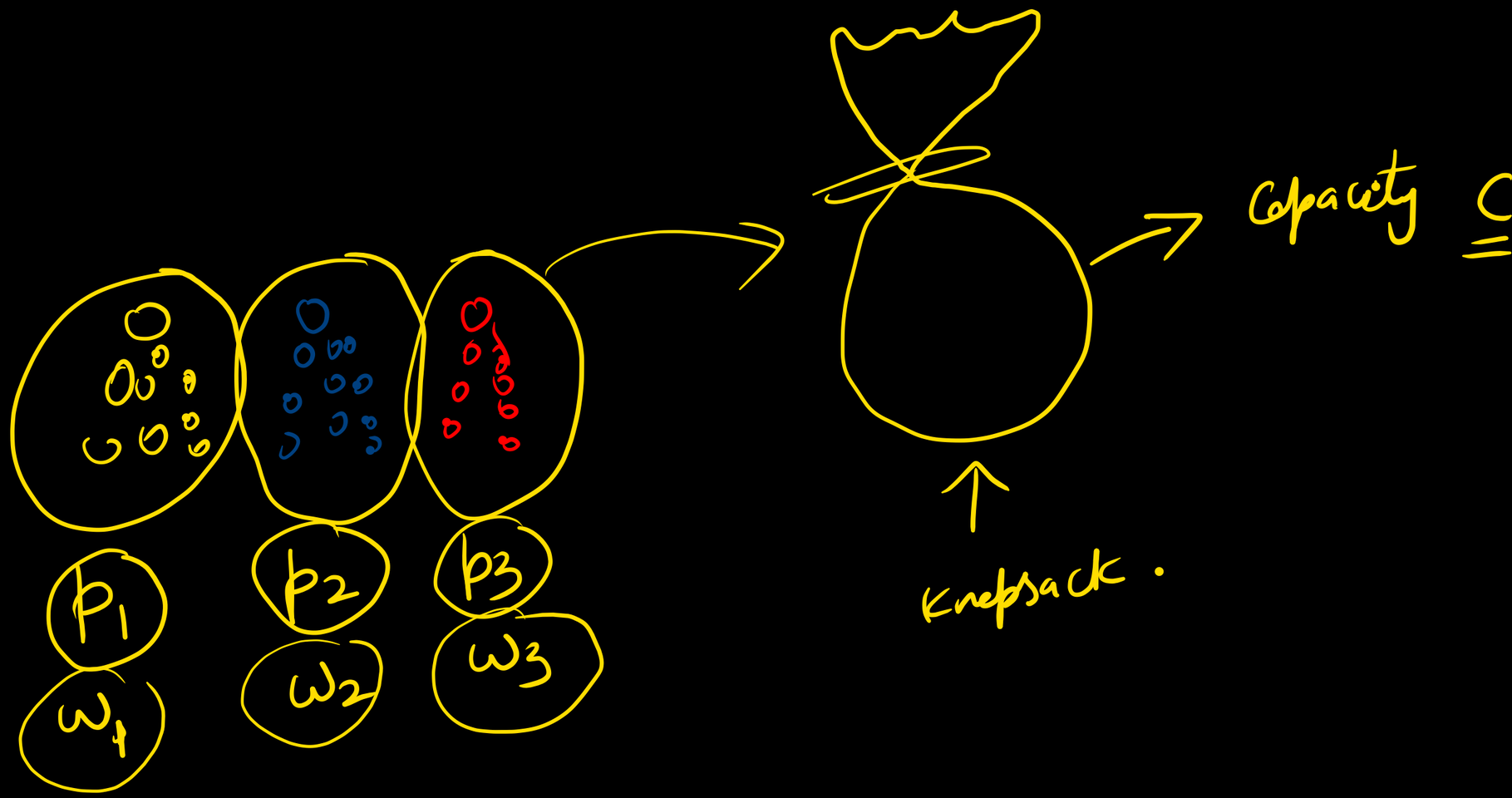


→ NP hard problem

↓

longest path b/w two nodes

Fractional knapsack:

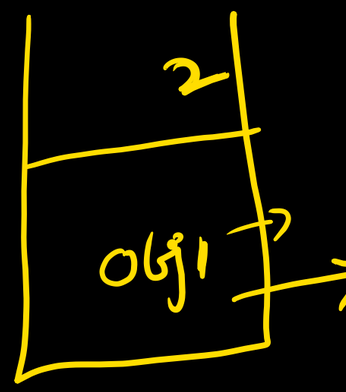


	obj1	obj2	obj3
profit	25	24	15
weight	18	15	10

Greedy about profit

obj1

n = 3 m = 20



2/0 2.

25

28.2 ✓ ✓

knapsack

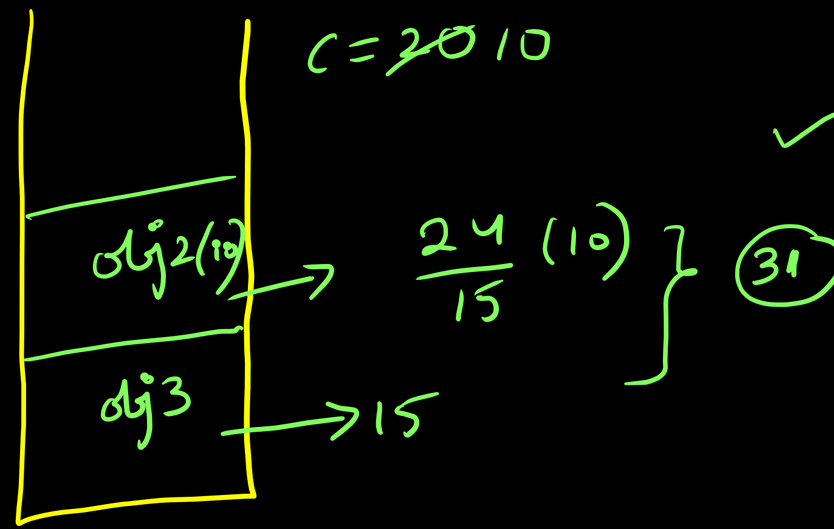
15 U → 24

20 → $\frac{24}{15} * 2$ ✓ = 3.2 ✓

	obj1	obj2	obj3
profit	25	24	15
weight	18	15	10

Greedy about weight

max profit
min weight.



	obj1	obj2	obj3
profit	25	24	15
weight	18	15	10
P/W	1.4	1.6	1.5

m = 20 5

obj3(5) → 15 x5/10

obj2 → 24

Profit (31.5)

Gate:

$n = 15$ $n = 7$

objects	1	2	3	4	5	6	7
profits	10	5	15	7	6	18	3
weights	2	(3)	5	7	1	4	1
p/w	5	1.6	3	1	6	4.5	3

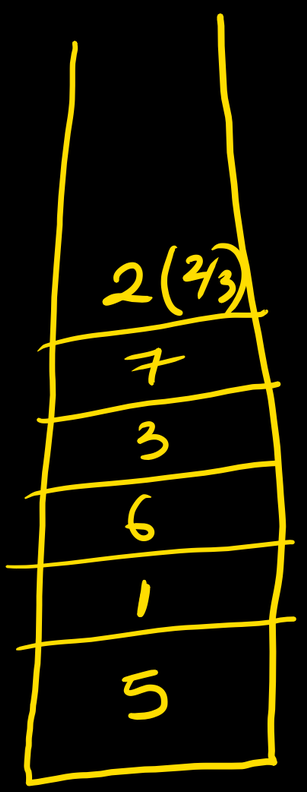
$(\cancel{5}, \cancel{1}, \cancel{6}, \cancel{3}, \cancel{7}, \cancel{2}, 4)$

$m = \cancel{15}\ \cancel{14}\ \cancel{12}\ \cancel{8}\ \cancel{3}$ (2)

$P = 6 + 10 + 18 + 15 + 3 + 5(2/3)$

$= 55.3$

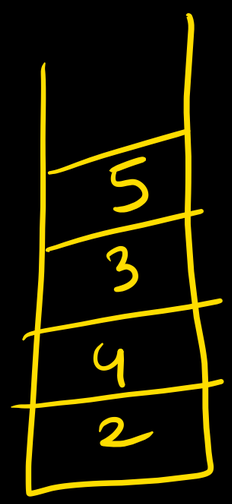
max profit = ?



Gate: $m=15$ $n=5$

objects	1	2	3	4	5
P	2	28	25	18	9
W	1	4	5	3	3
P/W	2	7	5	6	3

(1, 1, 1, 1, 1)



$m = 15 - 4 - 3 - 5 = 3$

$P = 28 + 18 + 25 + 9$

$= 80.$

Algo

→ Find P/W

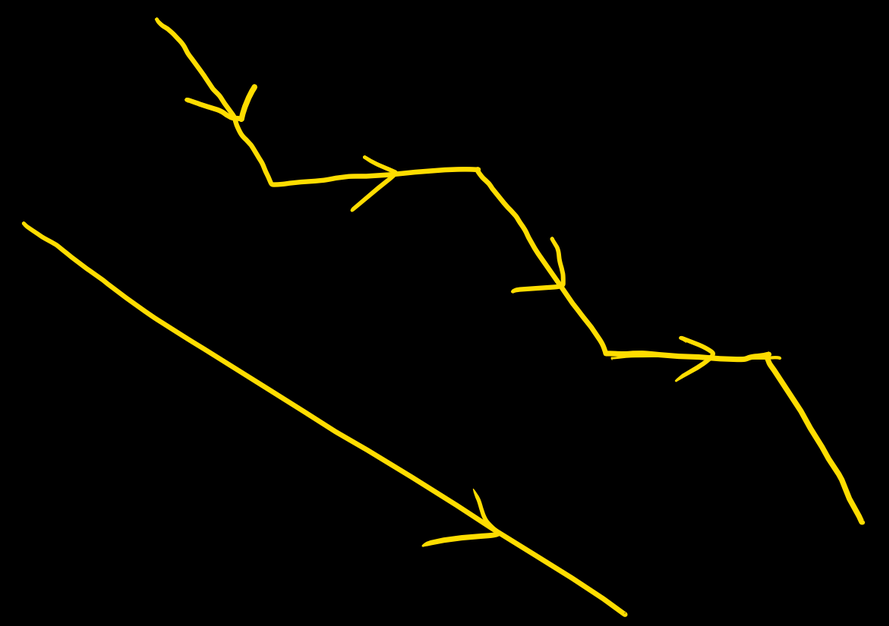
→ Sort them

→ add elements one by one till the knapsack is full

→ If only a part of an object can be put, then put that fraction.

Greedy knapsack:

```
{
    for (i = 1 to n);
        Compute pi/wi;
    Sort objects in non increasing order of p/w
    for (i = 1 to n) from sorted list
    if ( m >= 0 && wi <= m)
        m = m - wi;
        p = p + pi;
    else break;
    if (m >= 0)
        p = p + pi (m/wi);
```



Huffman Coding :-

To Compress a file

a b c d
a a b b c c .
d a b c a a .
d d c a b c .

2 bits
100 c - 200 bits
Variable encode
100 c - 160 bits.

a - 00
b - 01
c - 10
d - 11

100 character → 100 × 2 = 200 characters.

if a, b, c, d distribution is uneven

a - 50. → 0 (1)
b - 40. → 10 (2)
c - 5. → 110 (3)
d - 5. → 111 (3)

variable sized encoding. ✓

100 ✓

50 × 1 + 40 × 2 + 5 × 3 + 5 × 3

= 160

$n = 4$

charact	a	b	c	d
frequency	50	40	5	5

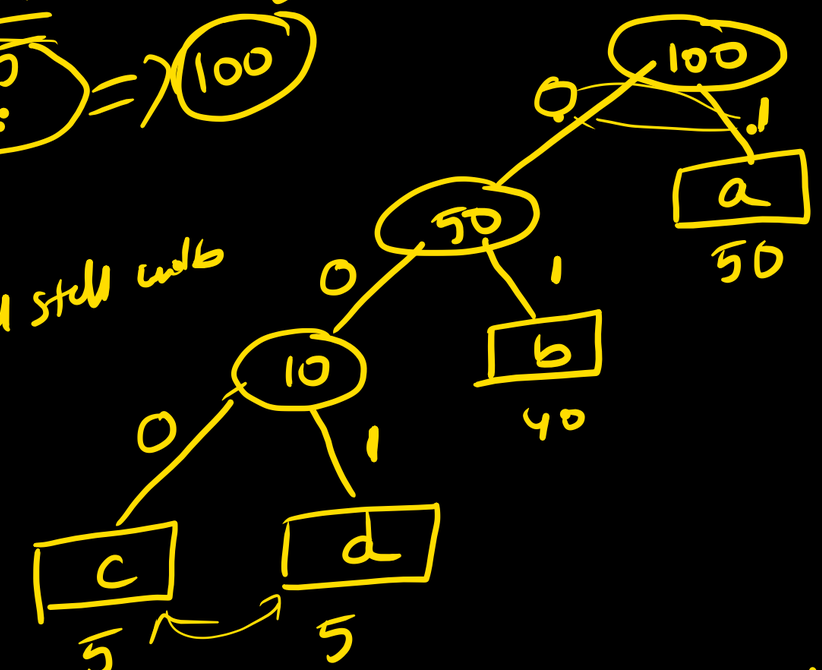
$\Rightarrow$ 100 ✓

a - 1
no other char will start with 1
b - 01 ✓

Total = $5 \times 3 + 5 \times 3 + 40 \times 2 + 50 \times 1$

$= 160$

avg bits/character $= \frac{160}{100} = 1.6$



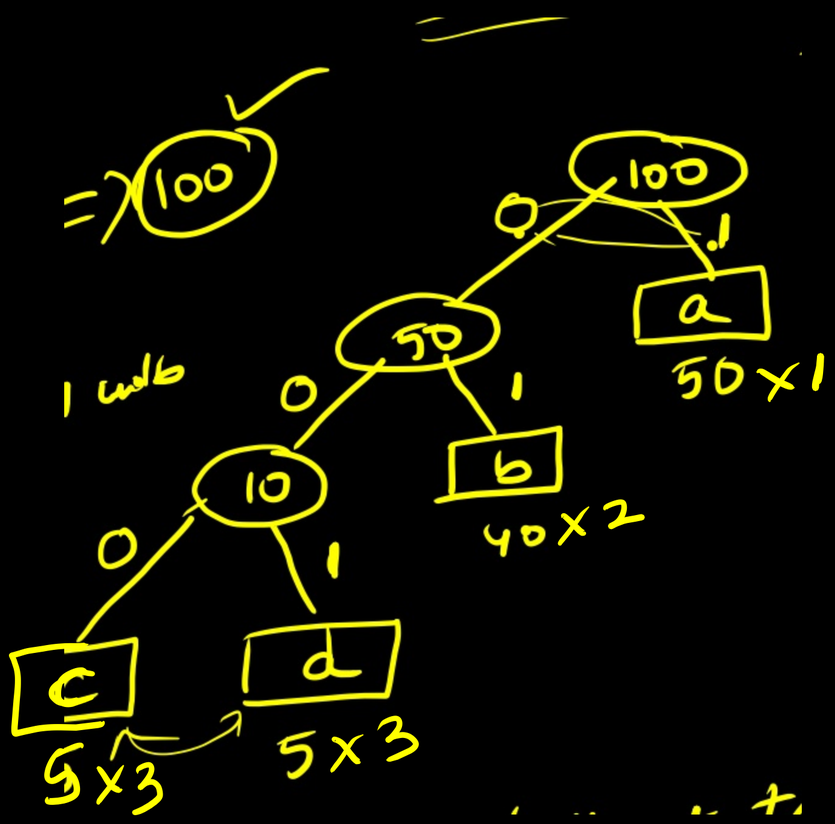
You may get different tree
encoding may also be different

Convention — LST should be smaller value

LST - 0 RST - 1

a - 1
b - 01
c - 000
d - 001

Prefix codes



weighted external path length :

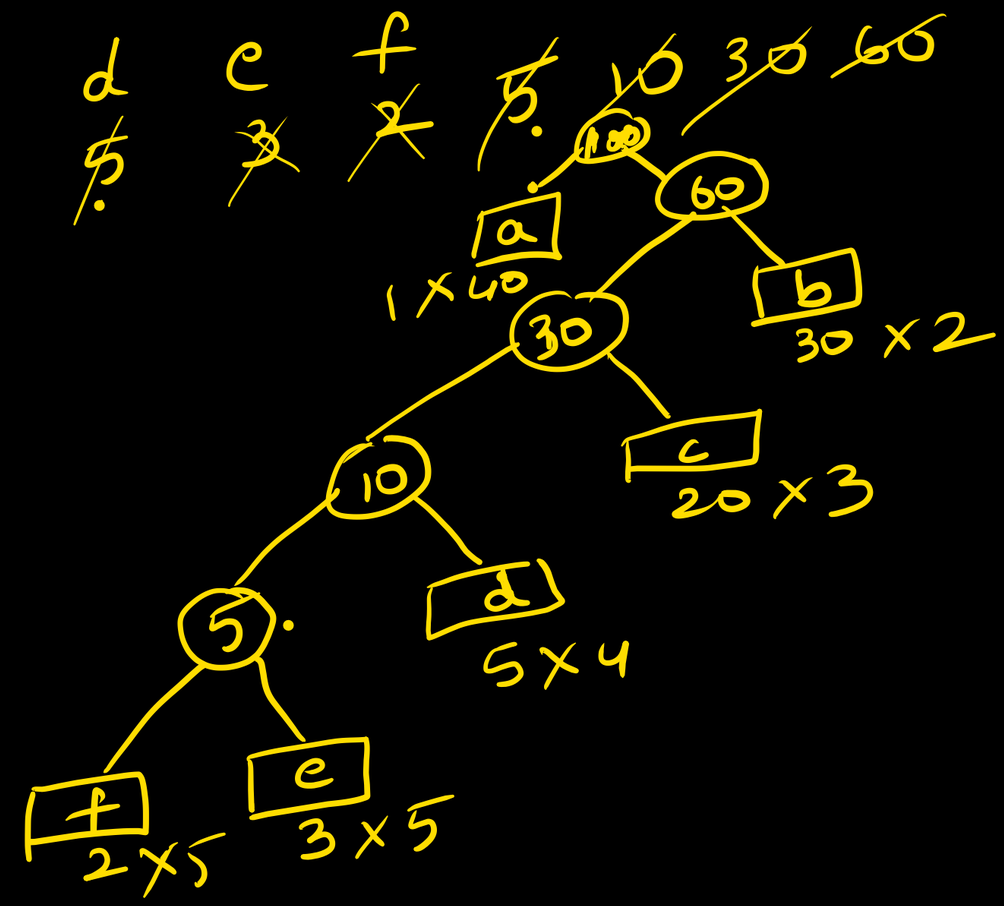
$= 5\times3 + 5\times3 + 40\times2 + 50\times1$

$=$

a - 40
b - 30
c - 20
d - 5
e - 3
f - 2

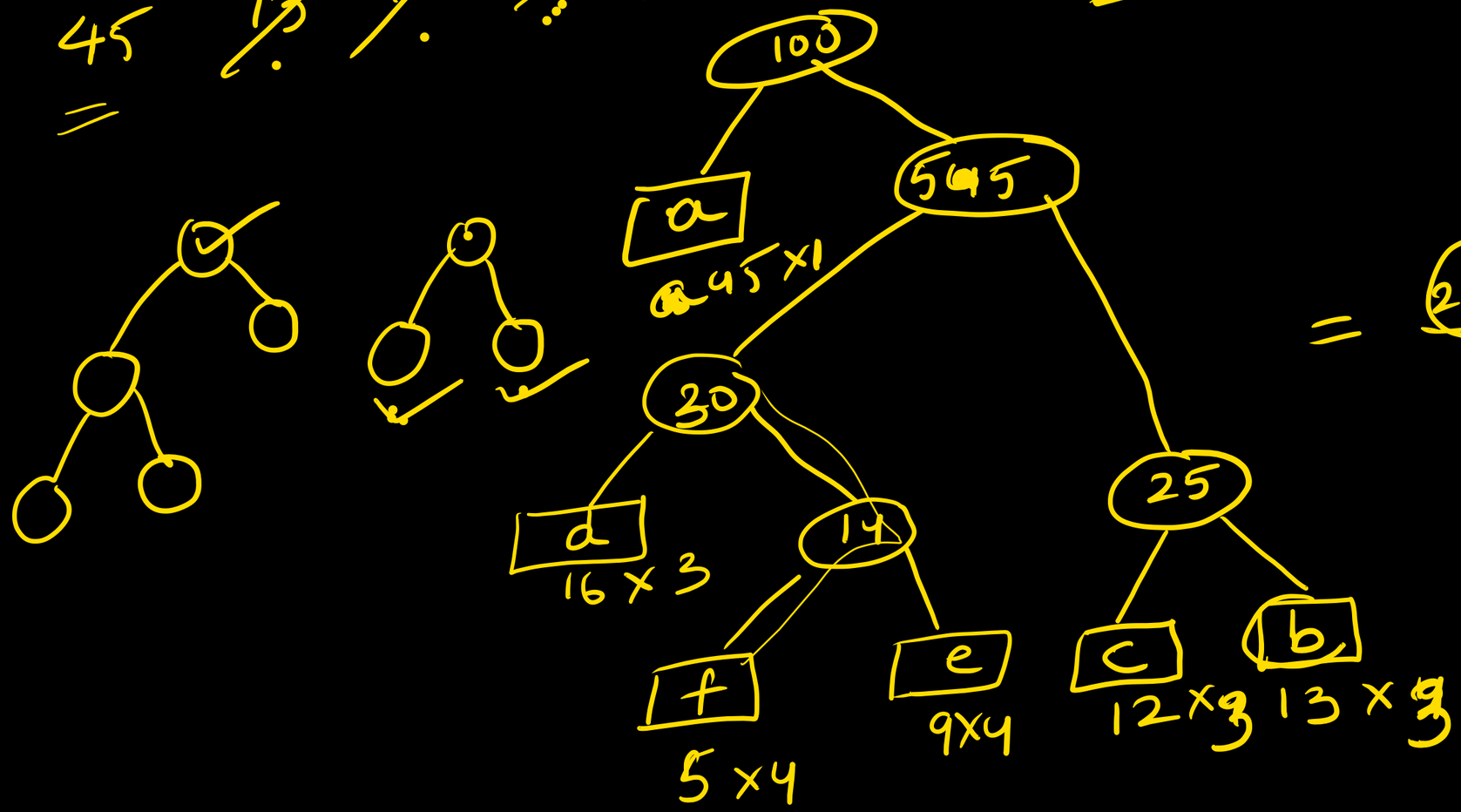
weighted external path length

a	b	c	d	e	f
~~40~~	~~30~~	~~20~~	~~5~~	~~3~~	~~2~~

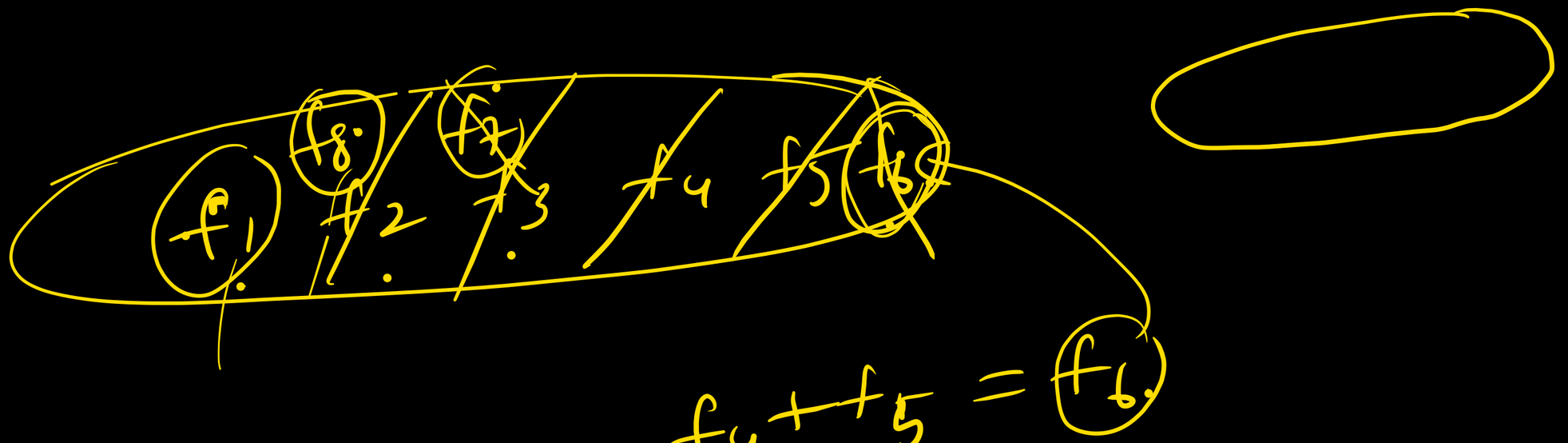


Gate:

a	b	c	d	e	f	14	25	30
45	13	12	16	9	5			



= 224.



$f_2 + f_3 = f_7$

$f_7 + f_1 = f_8$

Huffman (C)

SRT $n \log n$ ← $O(n)$ ✓ $O(n^2)$ $n \log n$ $S(=O(n)$

{ $n = |C|$

make a ~~min~~ min heap Q with C → $O(n)$

for (i = 1 to $n-1$)

allocate a new node z

$z.left = x =$ extract_min(Q) ✓ } $\log n \times 2$

$z.right = y =$ extract_min(Q) ✓

$z.freq = x.freq + y.freq$

Insert(Q, z) } → $\log n \times 1$

return (extract min Q))

}

Q = min heap

gate

Suppose the letters a, b, c, d, e, f have probabilities

1/2, 1/4, 1/8, 1/16, 1/32, 1/32 ✓ =

What is the huffman code for letters a, b, c, d, e, f

a) 0, 10, 110, 1110, 11110, 11111

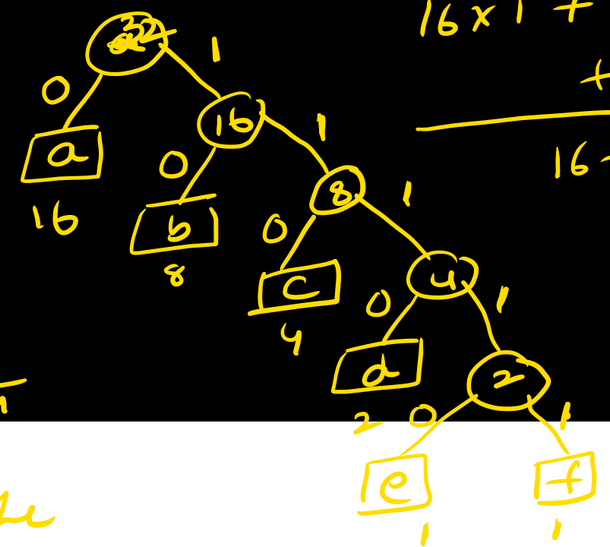
b) 11, 10, 011, 010, 001, 000

c) 11, 10, 01, 001, 0001, 0000

d) 110, 100, 010, 000, 001, 111

a b c d e f

16 8 4 2 1 1 ② 4 8 16



$16\times1 + 8\times2 + 4\times3 + 2\times4 + 1\times5 + 1\times5$

$16+8+4+2+1+1$

=

bit length/character

What is average length of the

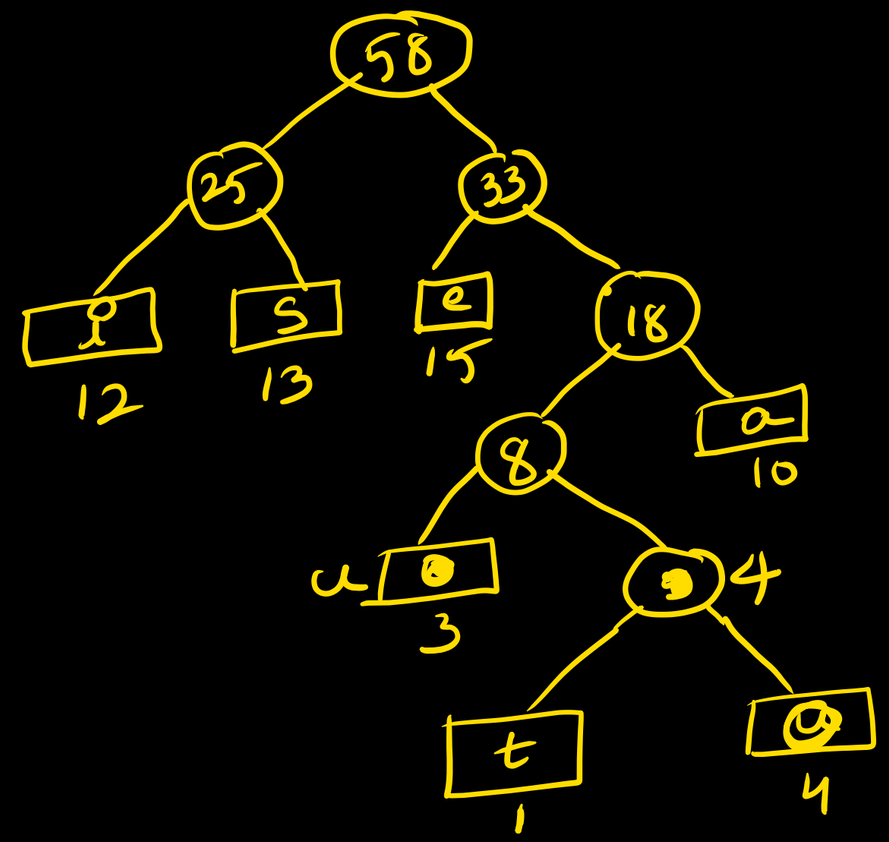
correct answer to the question ② ①

a) 3 b) 2.1875 c) 2.25 d) 1.9375

Code:

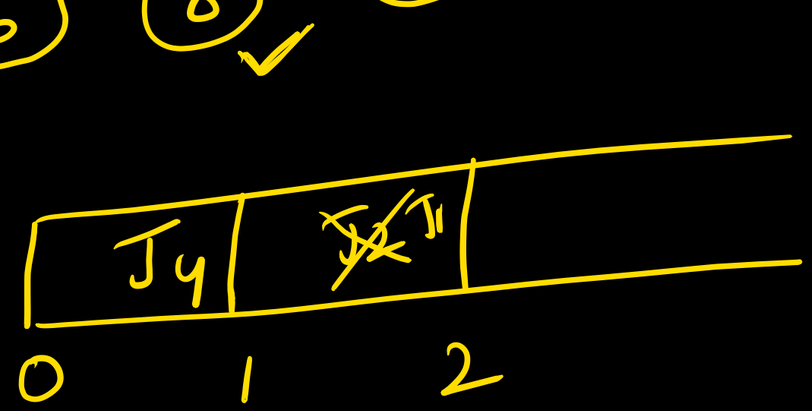
a c i o u s t

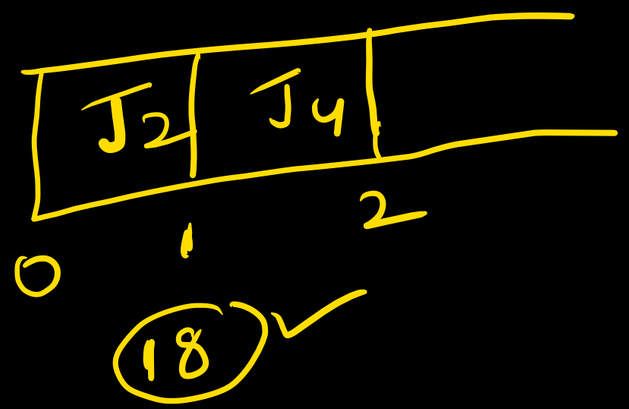
10 15 12 3 4 13 1 5 8 18 25 33



Job sequencing with deadlines: (greedy)

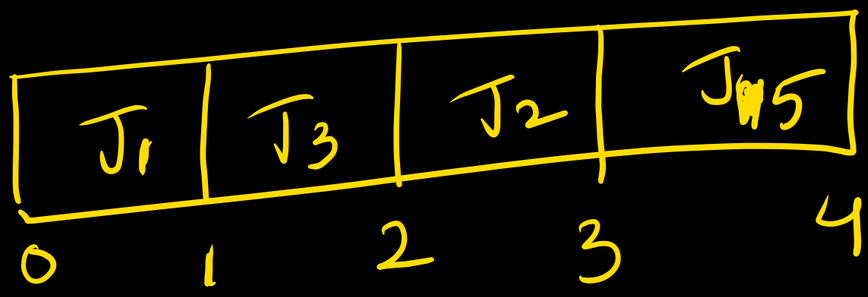
	J_1	J_2	J_3	J_4
D	2	1	1	2 ✓
P	6	8 ✓	5	10 ✓





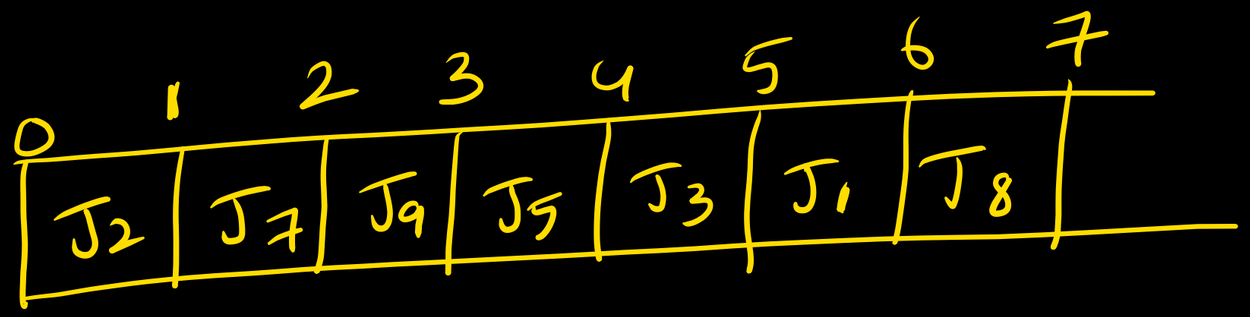
Gate:

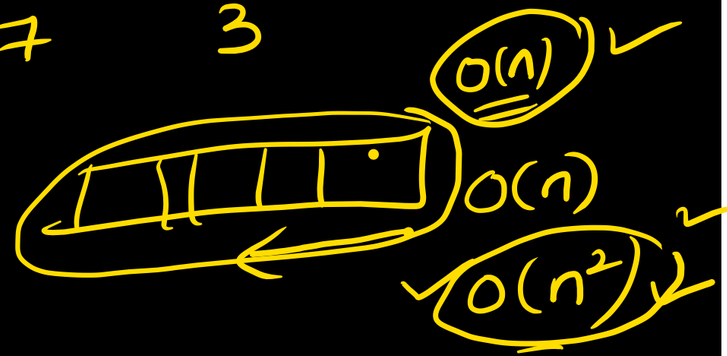
Jobs	J_1	J_2	J_3	J_4	J_5
Profits	2 ✓	4 ✓	3 ✓	1	10 ✓
deadline	3	3	3	4	4



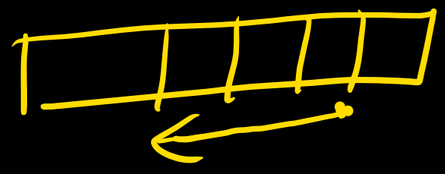
Gate:

	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9
P	15	20	30	18	18	10	23	16	25
D	7	2	5	3	4	5	2	7	3





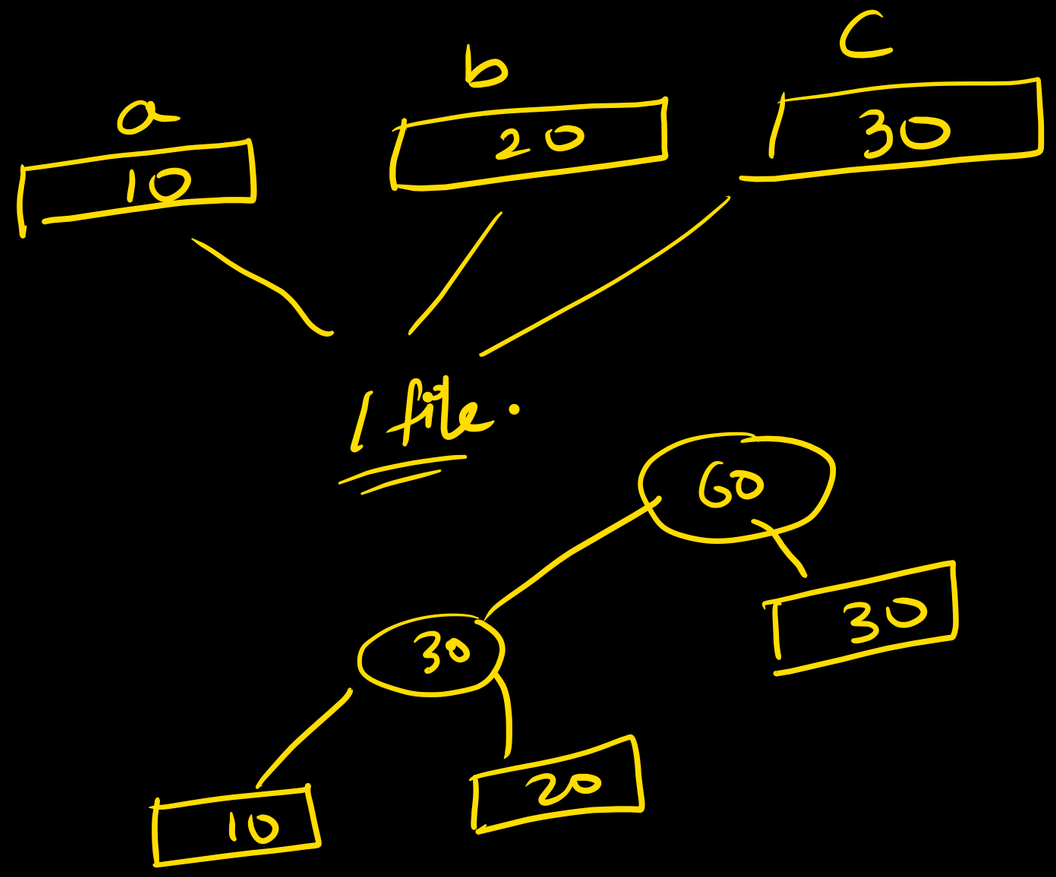
algo: → Sort Jobs according to profit = $O(n \log n)$



$O(n) \times n$

$O(n^2)$. $O(n^2)$

optimal merge pattern:



Extend weight path length
Huffman coding.

a	b	c	d	e	f
40	10	20	15	25	30