Data Structures

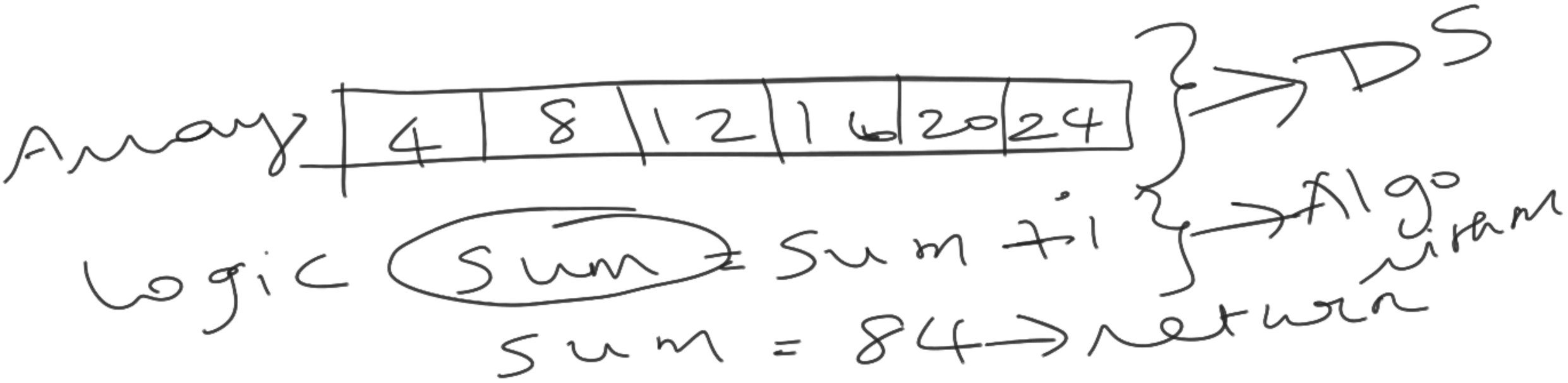
&

Algorithms

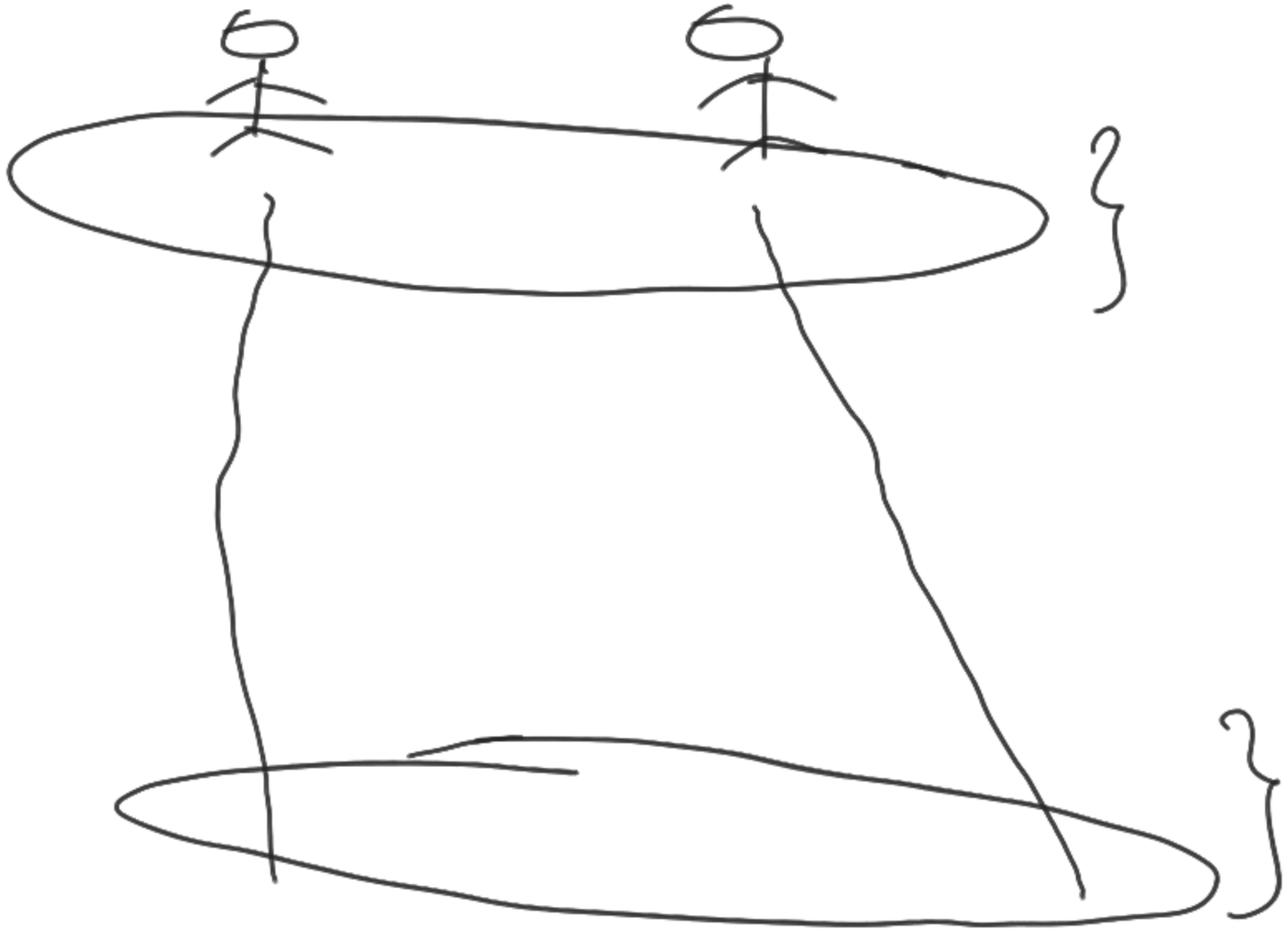
Big O

O(n)

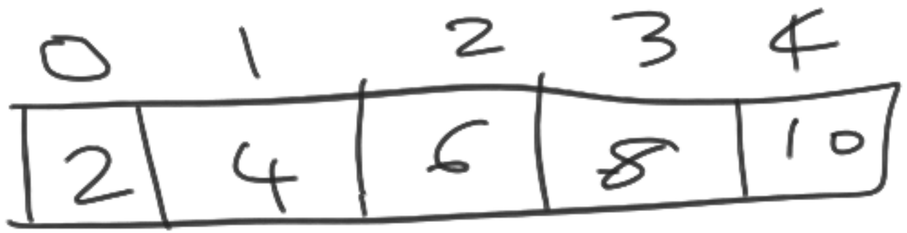
time & Space



Time Complexity



Array



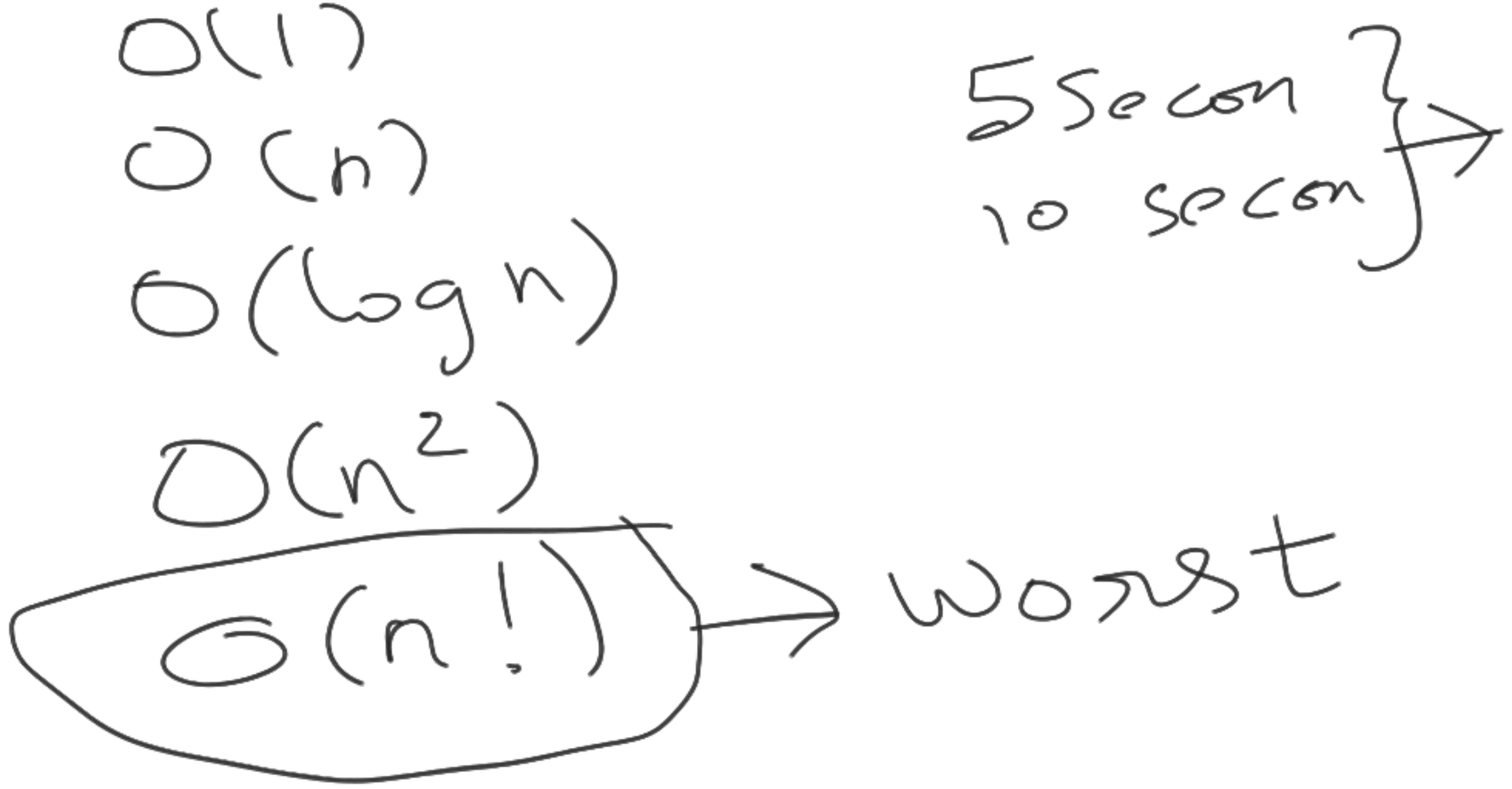
Linked List

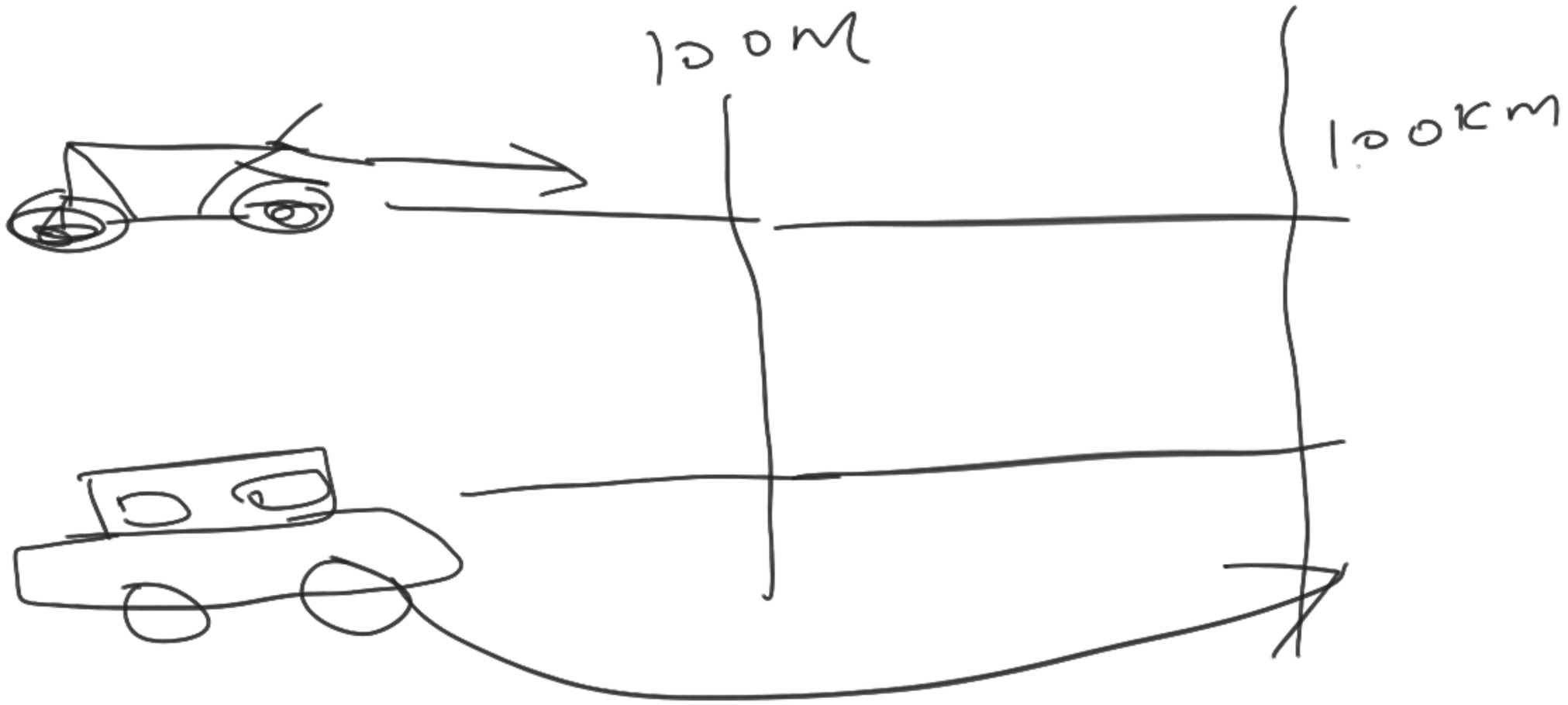
Access → 10 data[4] → 10 O(1)

length = 5

n = 5

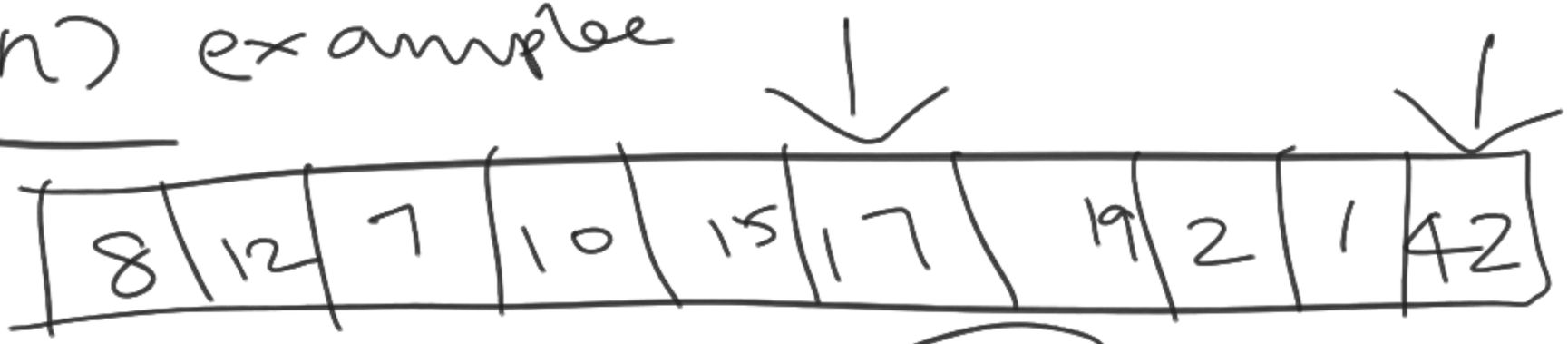
O(n)


O(1)
O(n)
O(log n)
O(n²)
O(n!)
worst
5 secon
10 secon



notations

O(n) examples



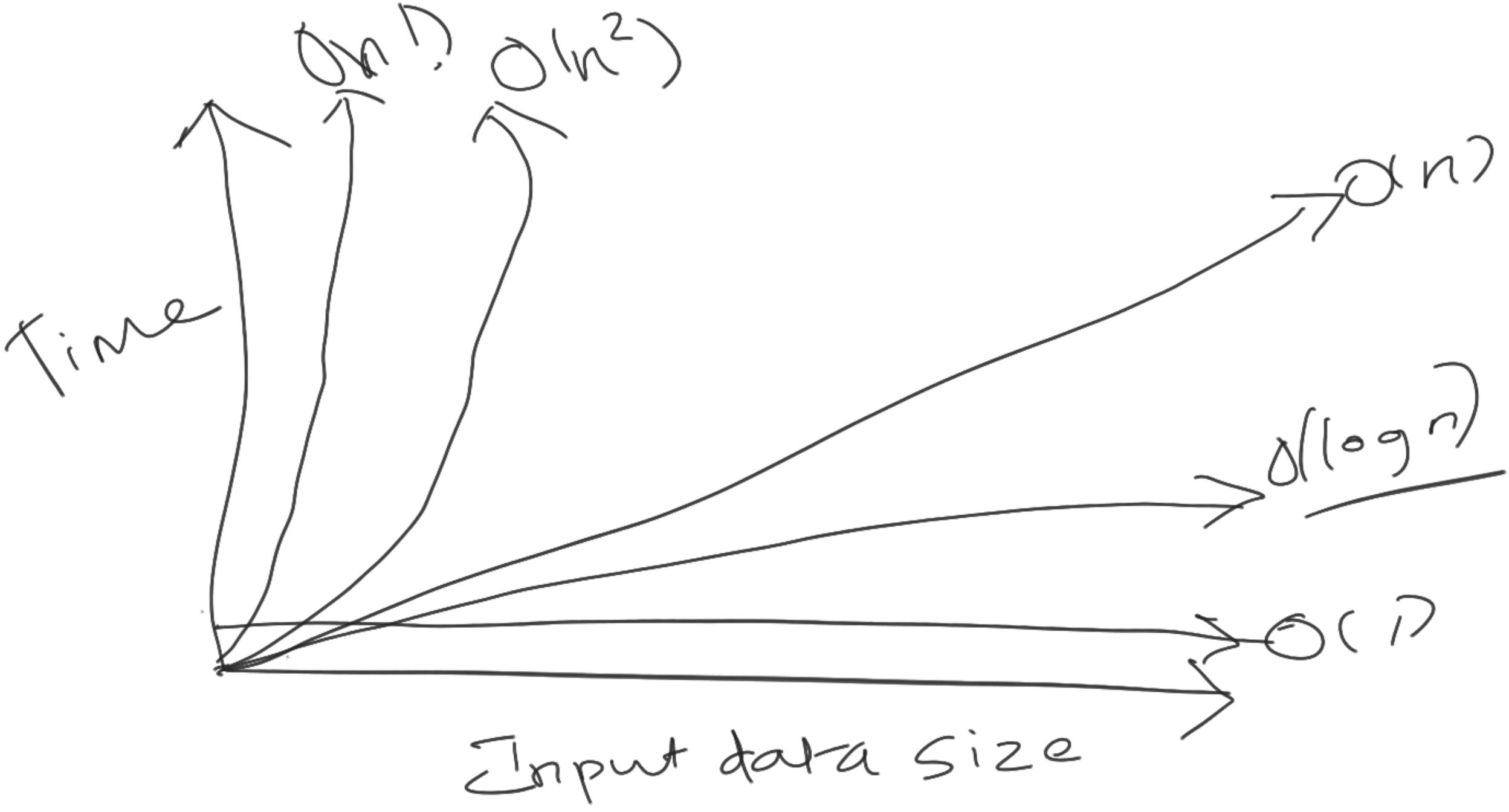
Find 42

Access → data[0] ..... [length-1]

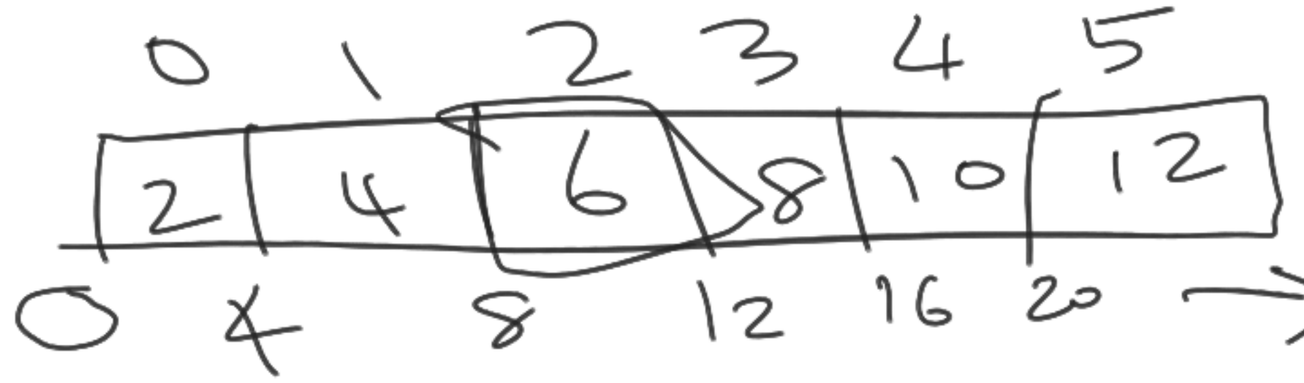
Search → 42 == data[i]

O(n)

Woohoo!


O(n!)
O(n²)
O(n)
O(log n)
O(1)
Time
Input data size

Q(1)



32 bit machine integer → 4 bytes

Find (10) → index → 4

data(4) = 10

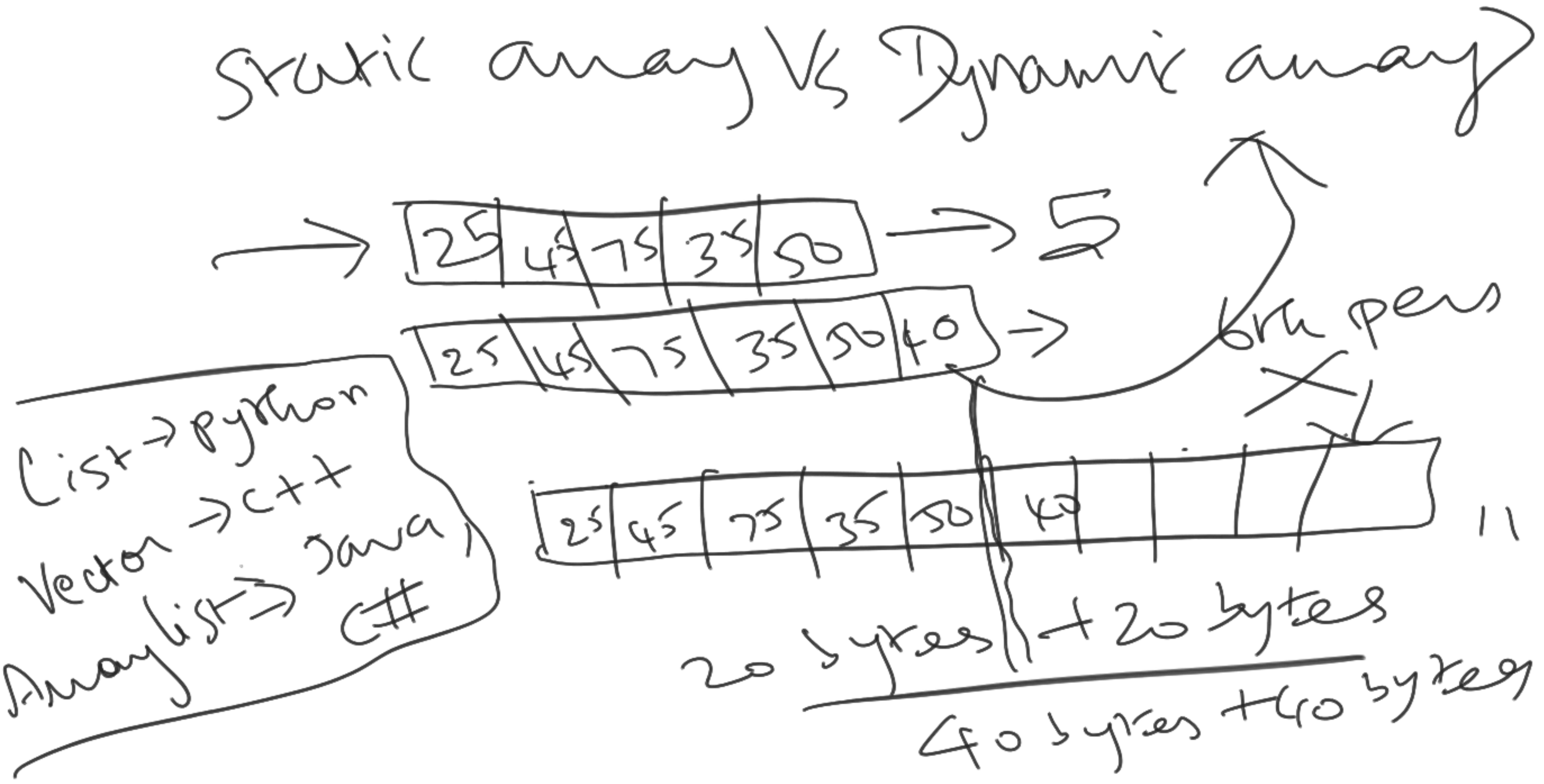
Ans (Contiguous)

data[4] = (4×4) = 16

= 10

5 × 4 bytes = 20 bytes → (Continuous)

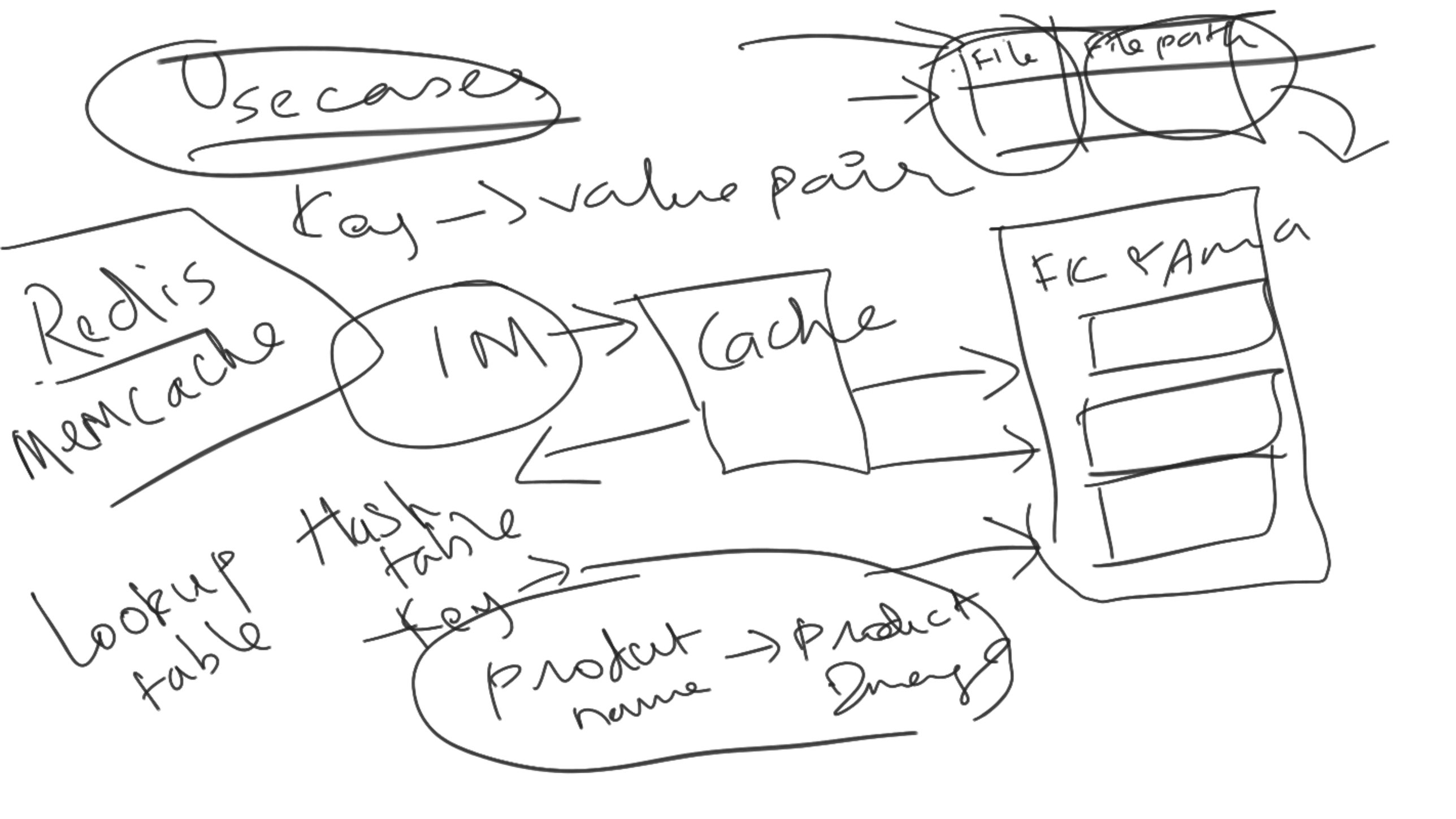
Static array Vs Dynamic array



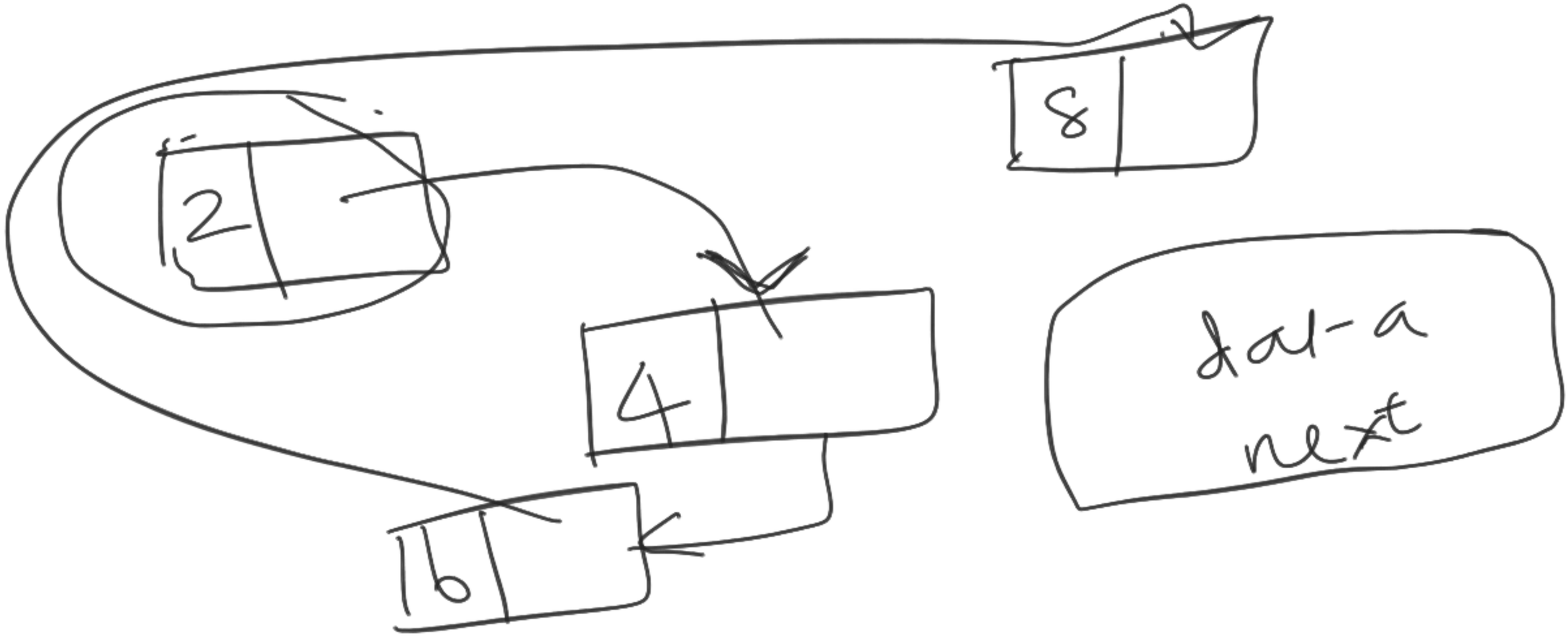
Use cases of Arrays

Array → list, hashtables, stack, queues

O(1) → Hash tables

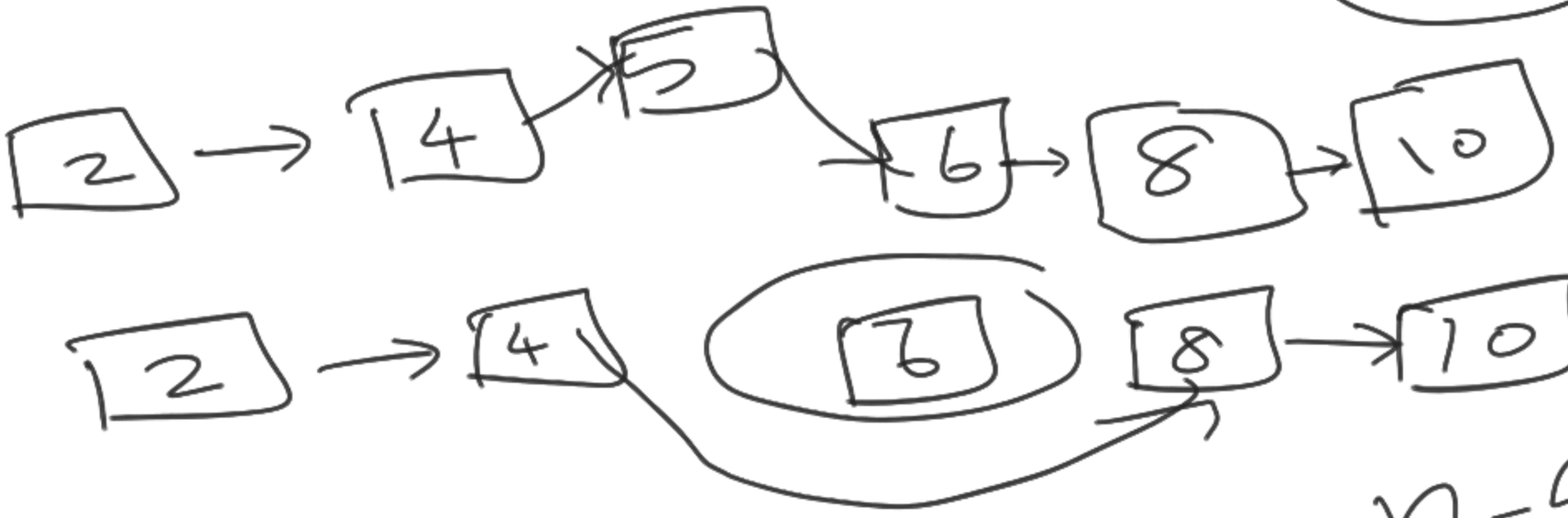

Usecase
File
File path
key -> value pair
Redis
Memcache
IM
Cache
FK RAM
Lookup table
Hash table
key
product name -> product Image

O(n) → linkedlist



O(n)

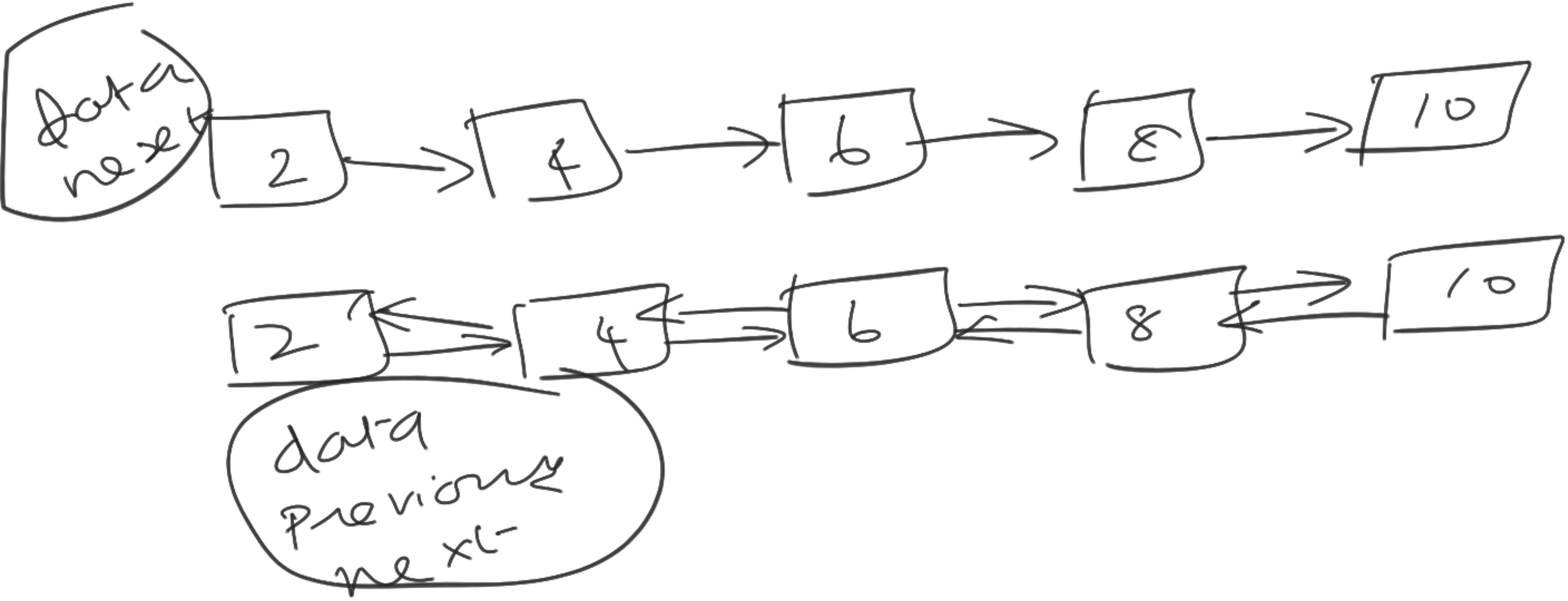
add
deletion



n=5

O(n)-

Singly Linked List Vs Doubly Linked List



Stack

ADT

push →
pop →
seek →

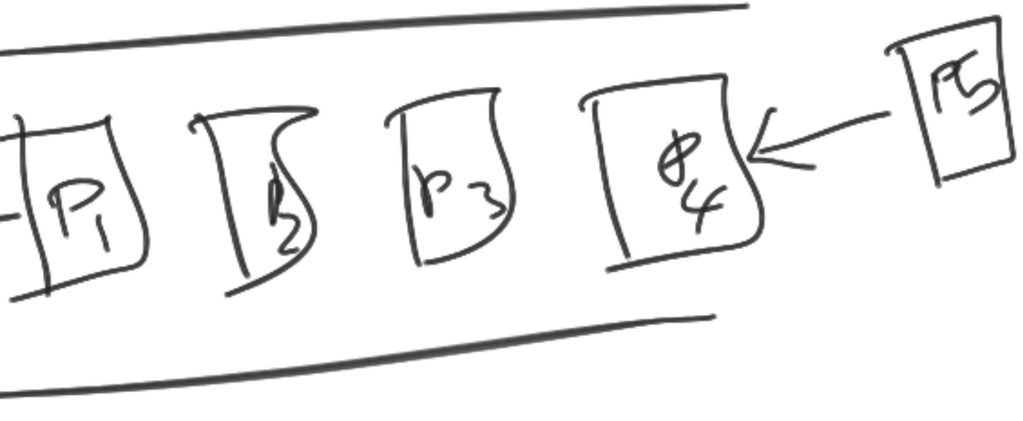
LIFO

Usecases

Undo
Syntax check

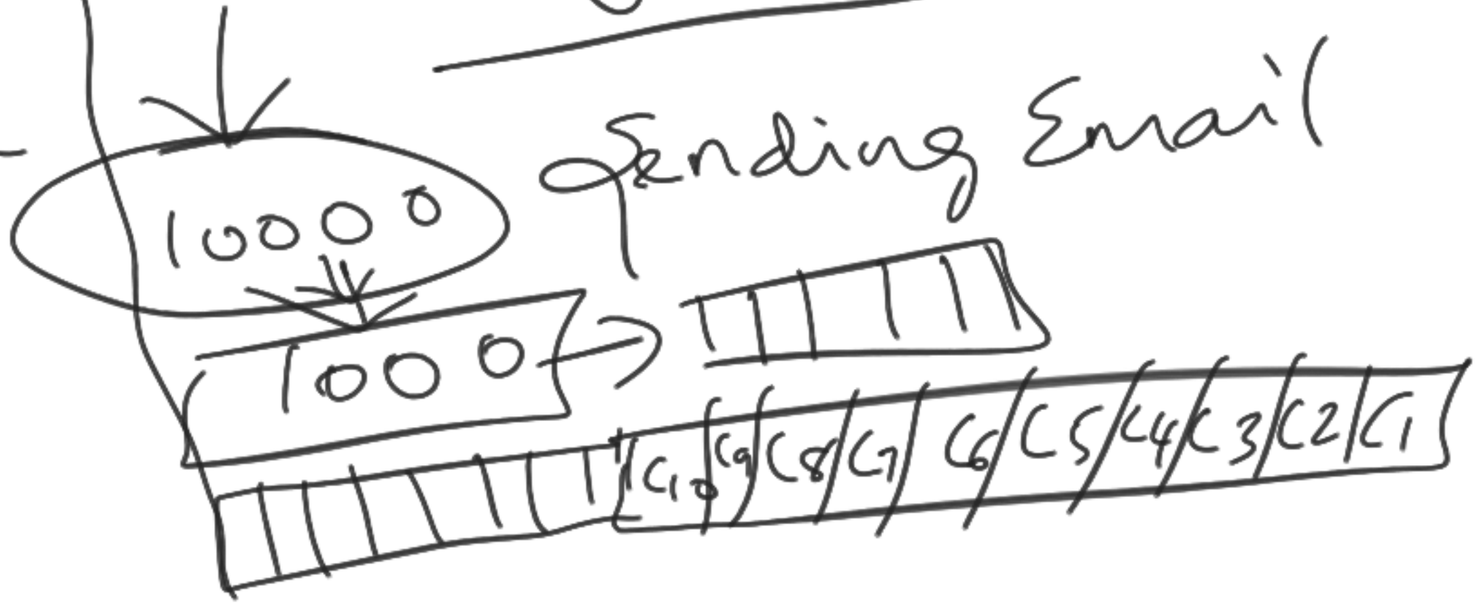
Queues

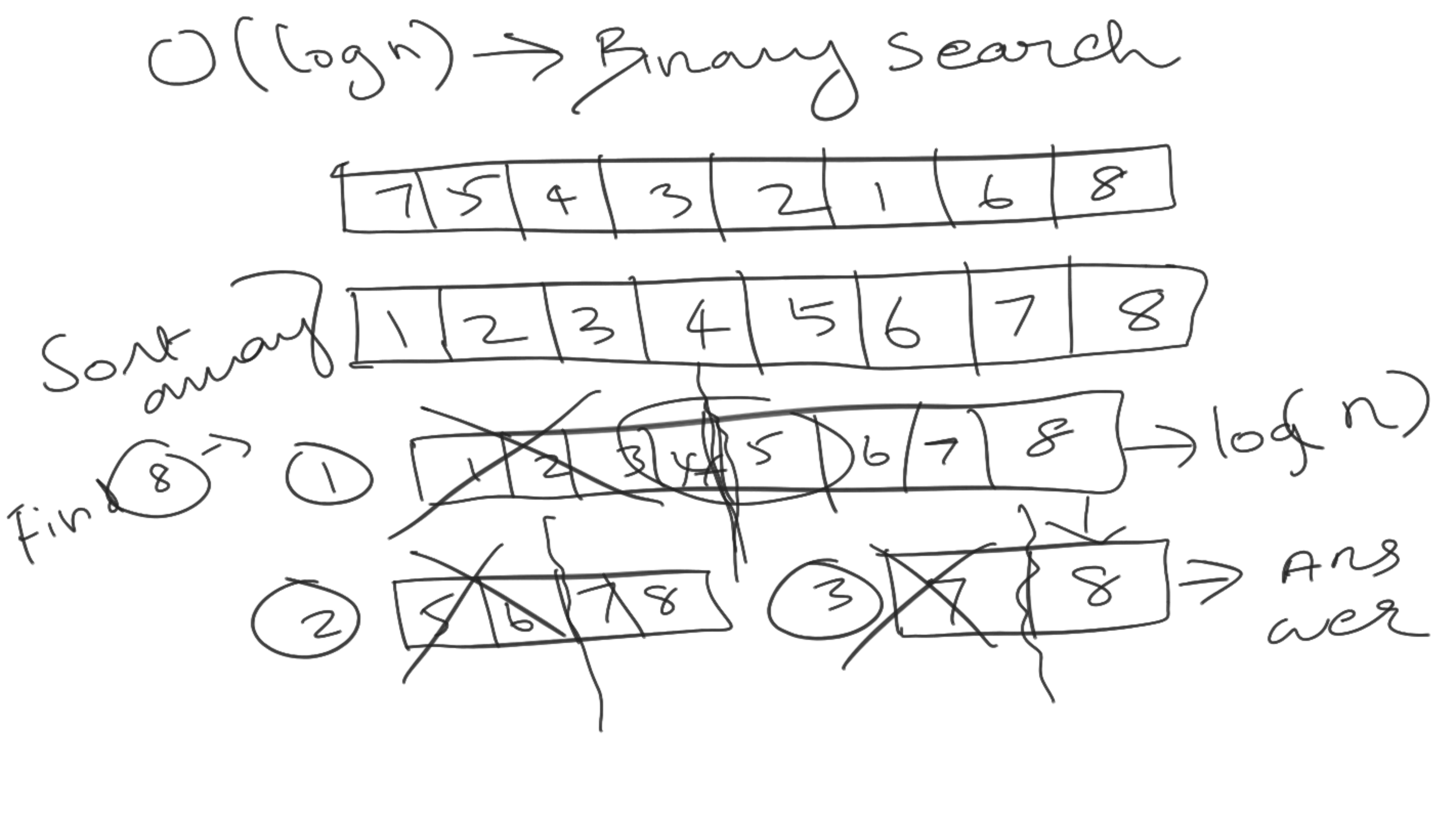
FIFO



counter

Usecases




O(logn) → Binary search
7 5 4 3 2 1 6 8
Sort array
1 2 3 4 5 6 7 8
Find 8 →
1
1 2 3 4 5 6 7 8
→ log(n)
2
5 6 7 8
3
7 8
→ Answer

O(n!)

XYZ

XYZ
XZY
YXZ
YZX
ZXY
ZYX

3!
3×2×1 = 6

2! = 2×1 = 2
3! = 3×2×1 = 6
4! = 24
5! = 120
100! = ____

n = 4

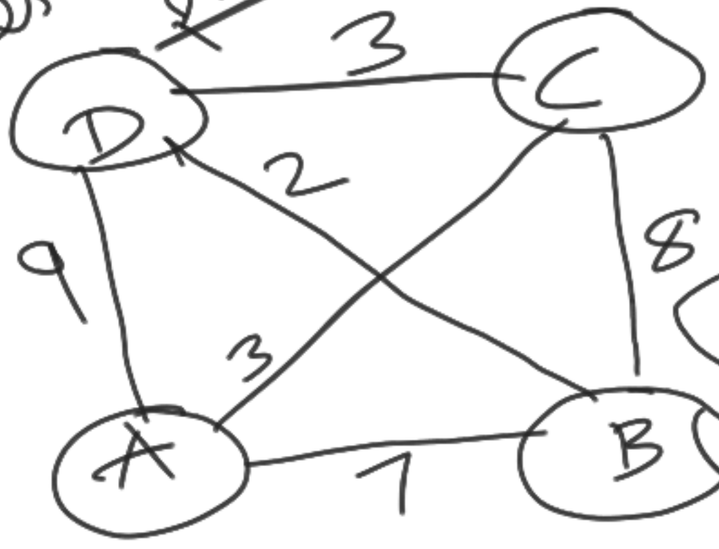
Travelling Salesman problem

3 possible paths

O(n!)/2

O(n-1)!

(4-1)!/2 = 3!/2 = 6/2 = 3

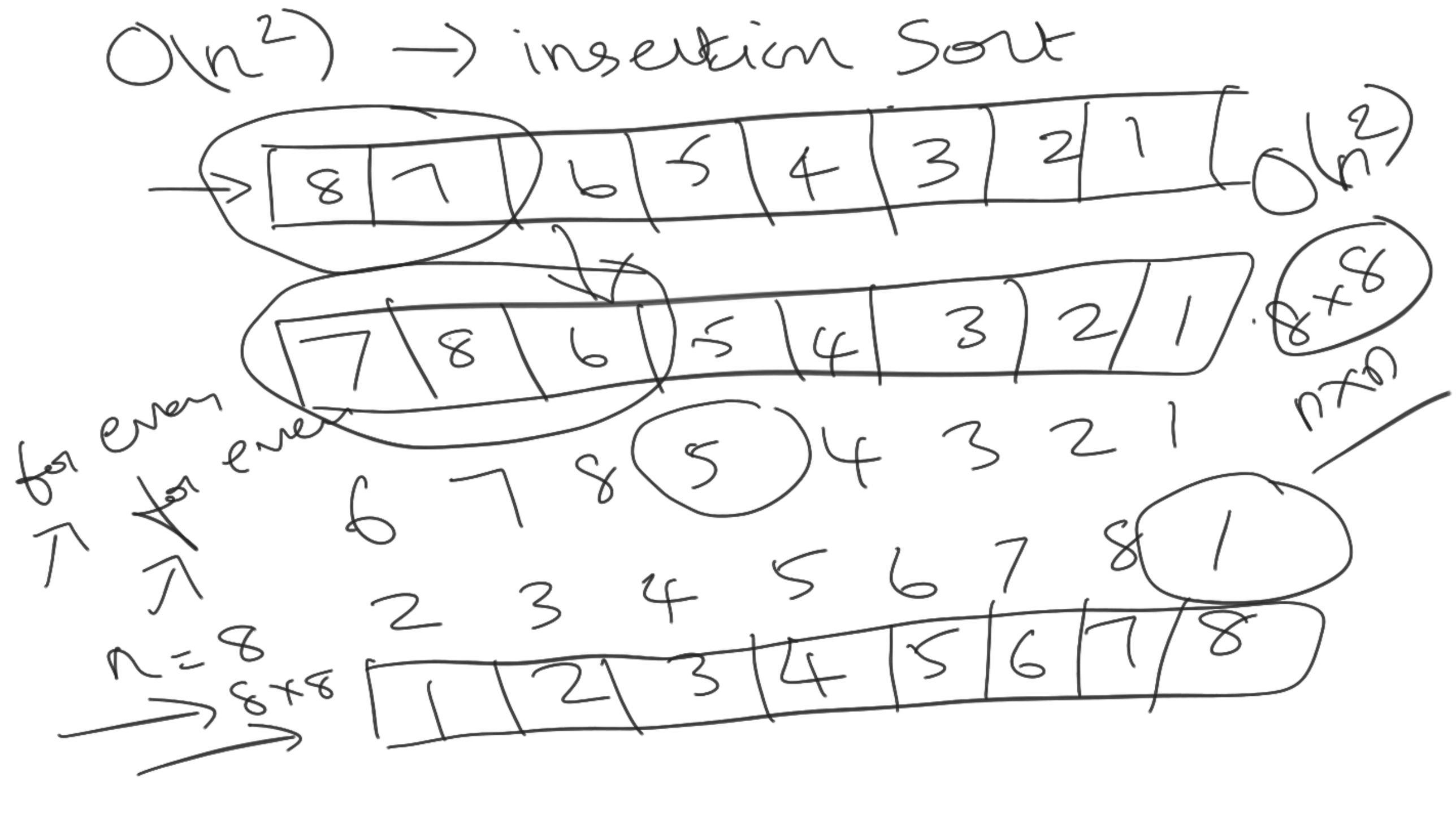


8 (A-B)(B-C)(C-D)(D-A) 27

(A-B)(B-D)(D-C)(C-A) 15

A-C)(C-B)(B-D)(D-A) 22

~~(A-D)(D-B)(B-C)(C-A)~~


O(n²) → insertion Sort
8 7 6 5 4 3 2 1
O(n²)
7 8 6 5 4 3 2 1
8x8
for even
for even
6 7 8 5 4 3 2 1
n x n
2 3 4 5 6 7 8 1
n = 8
8x8
1 2 3 4 5 6 7 8