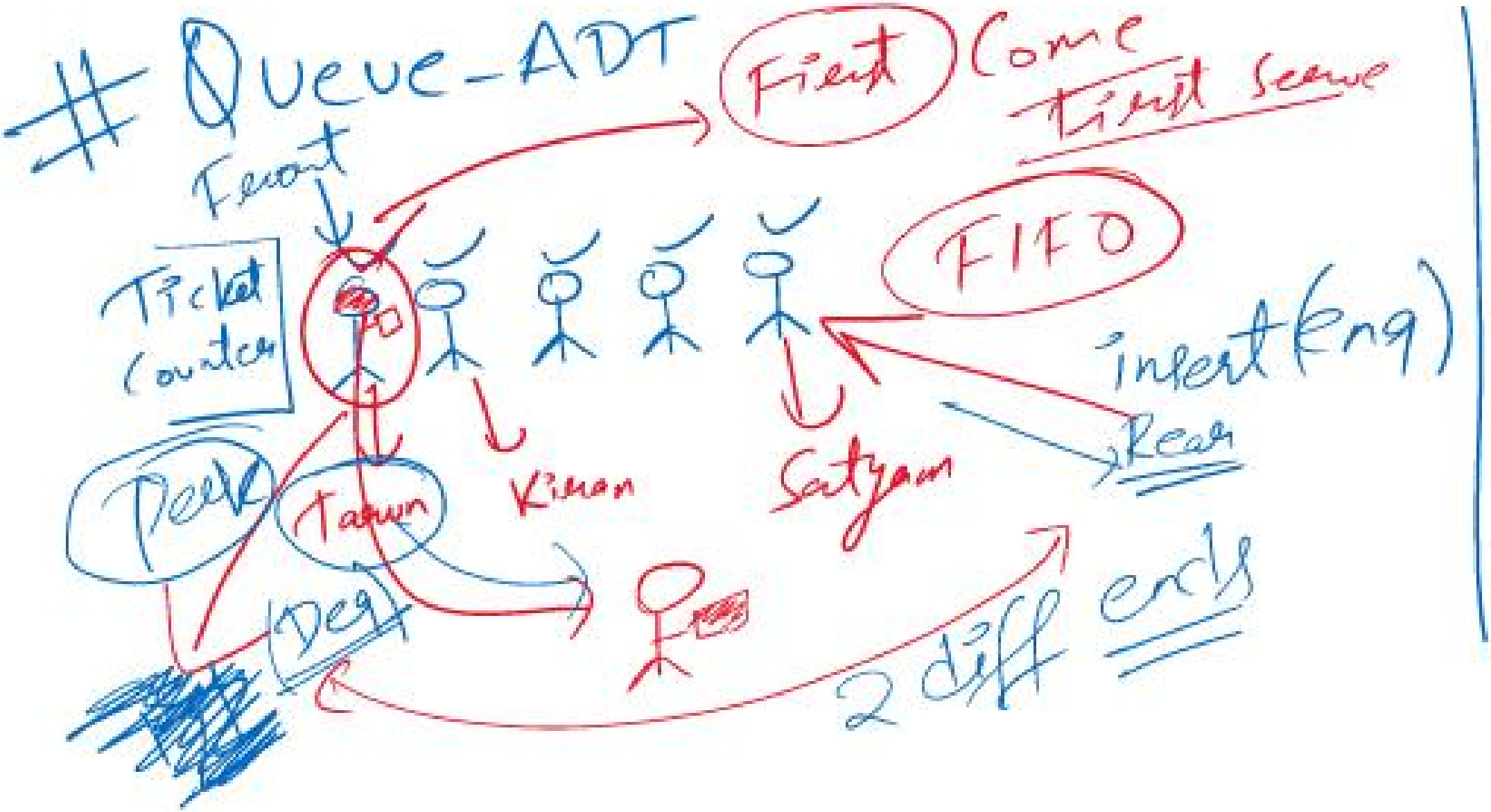
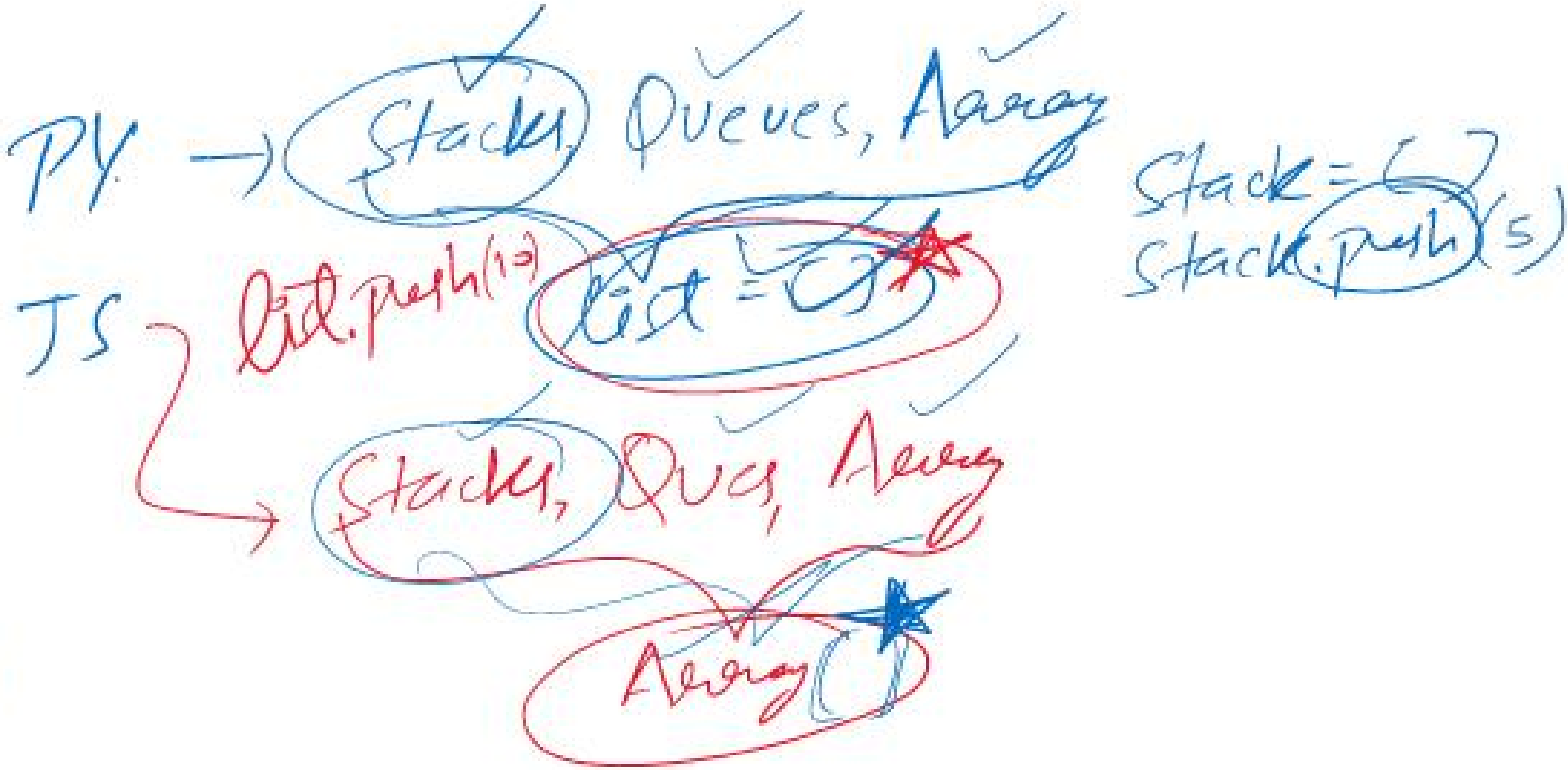

Queue-ADT
First
Come First Serve
Front
Ticket counter
FIFO
Insert(Enq)
Rear
Deq
Tarun
Kiran
Satyam
2 diff ends


PY
Stacks, Queues, Arrays
Stack = []
Stack.push(5)
JS
list.push(1)
list = []
Stacks, Ques, Arrays
Array()

Node & Linked list

Classes

Node

PY & JS

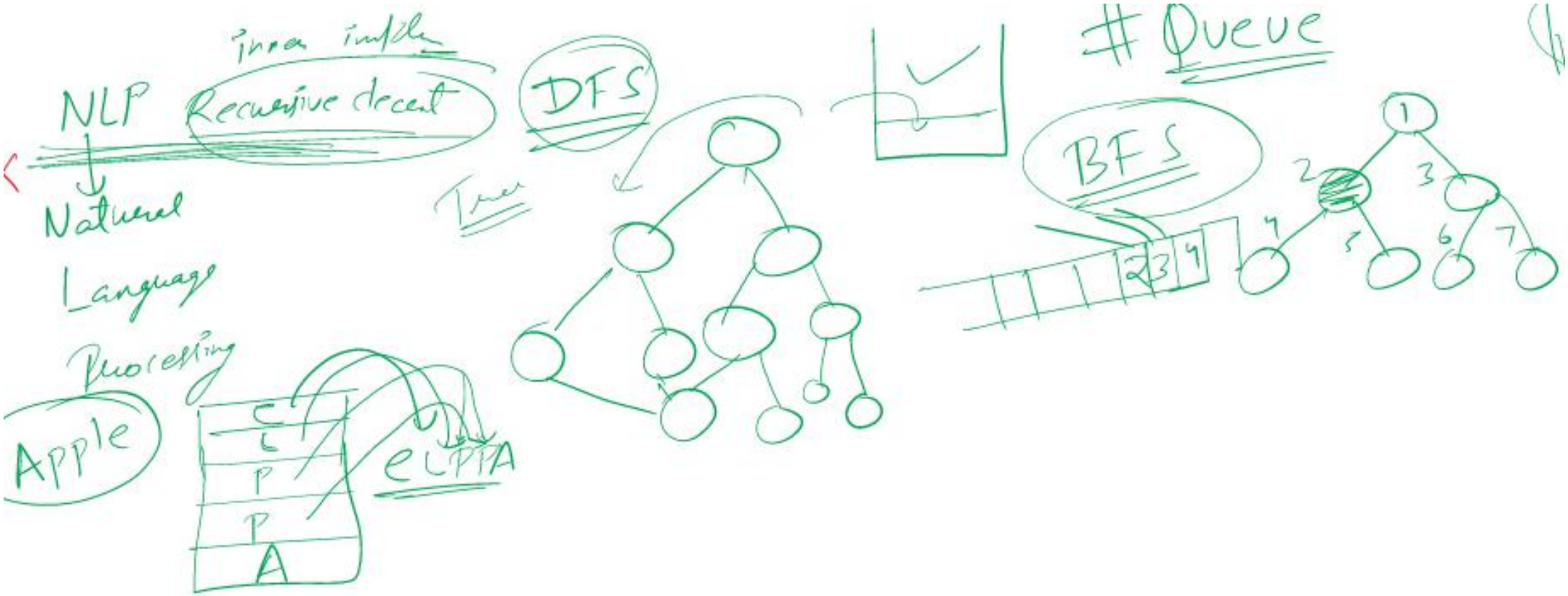
Uses & Applications of Linear DS

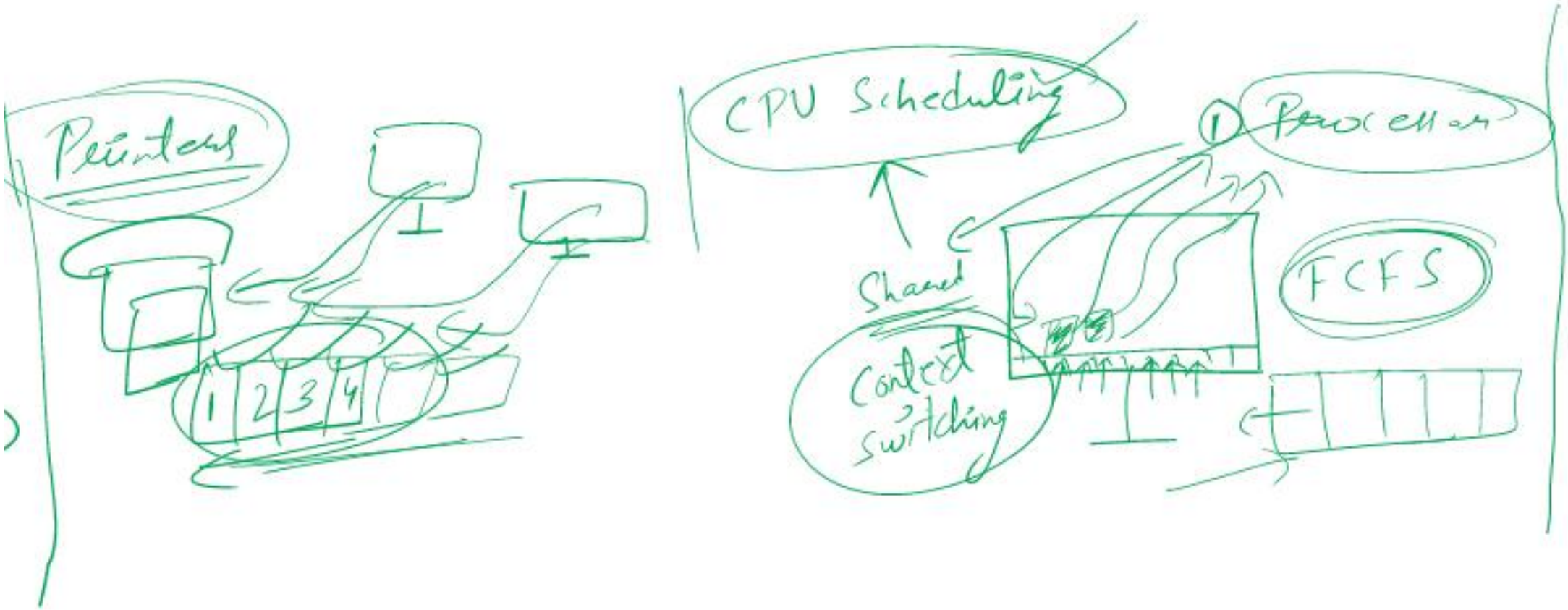
① Stack :

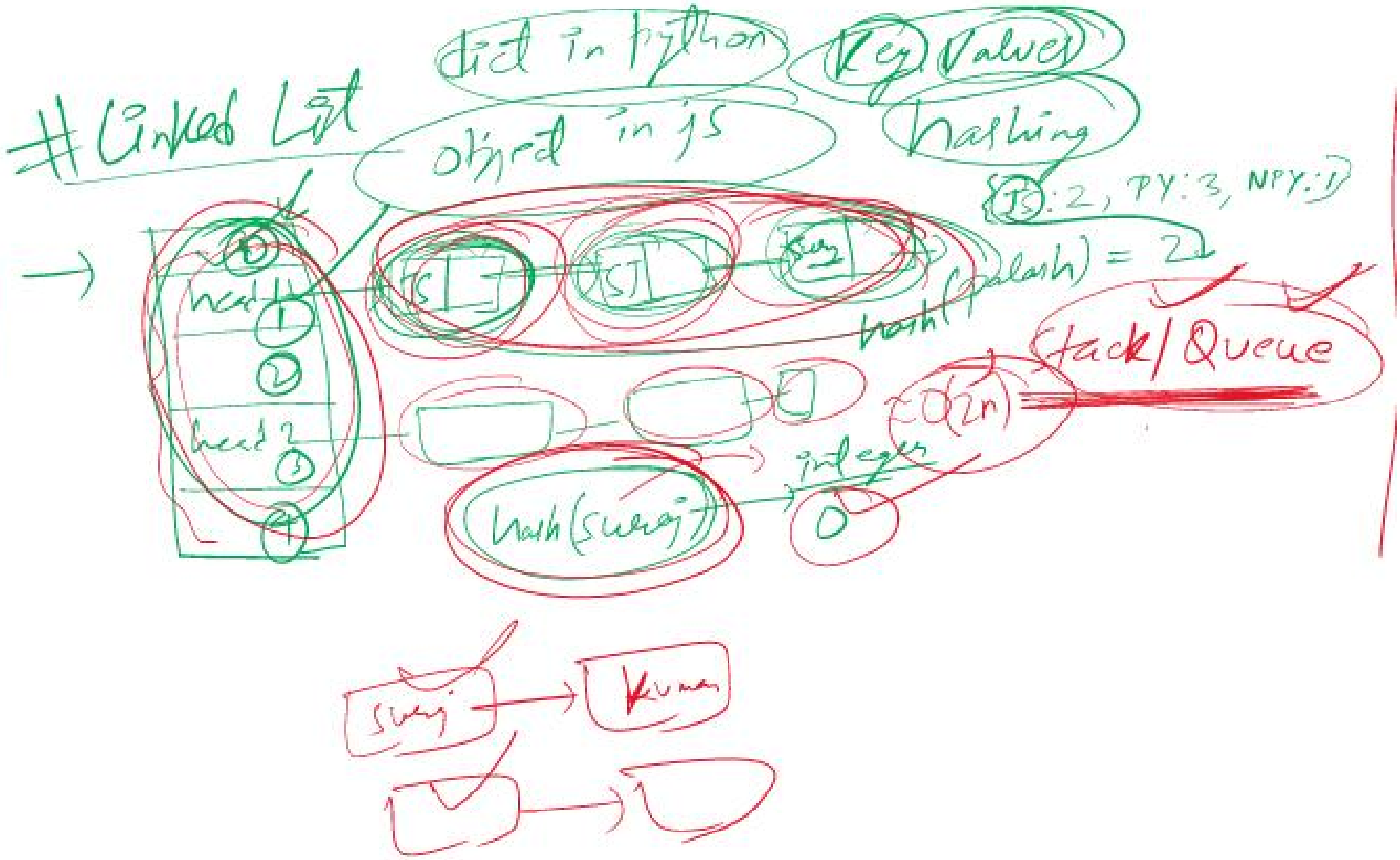
Expression Eval

Prefix, Postfix

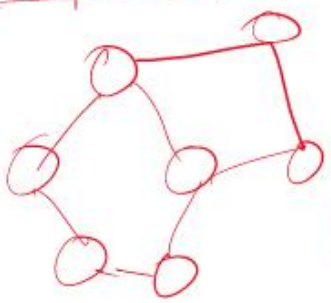
Backtracking

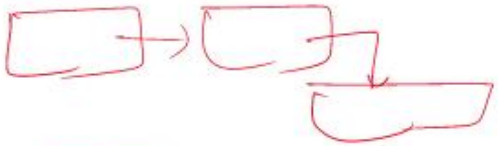

NLP
Natural
Language
Processing
Recursive descent
DFS
Tree
Apple
ELPPA
Queue
BFS

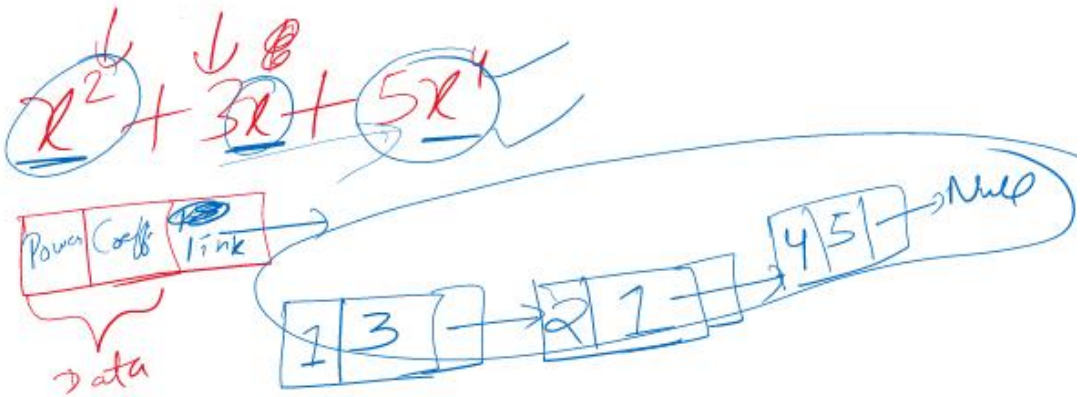

Printers
1
2
3
4
CPU Scheduling
Shared
Context switching
FCFS

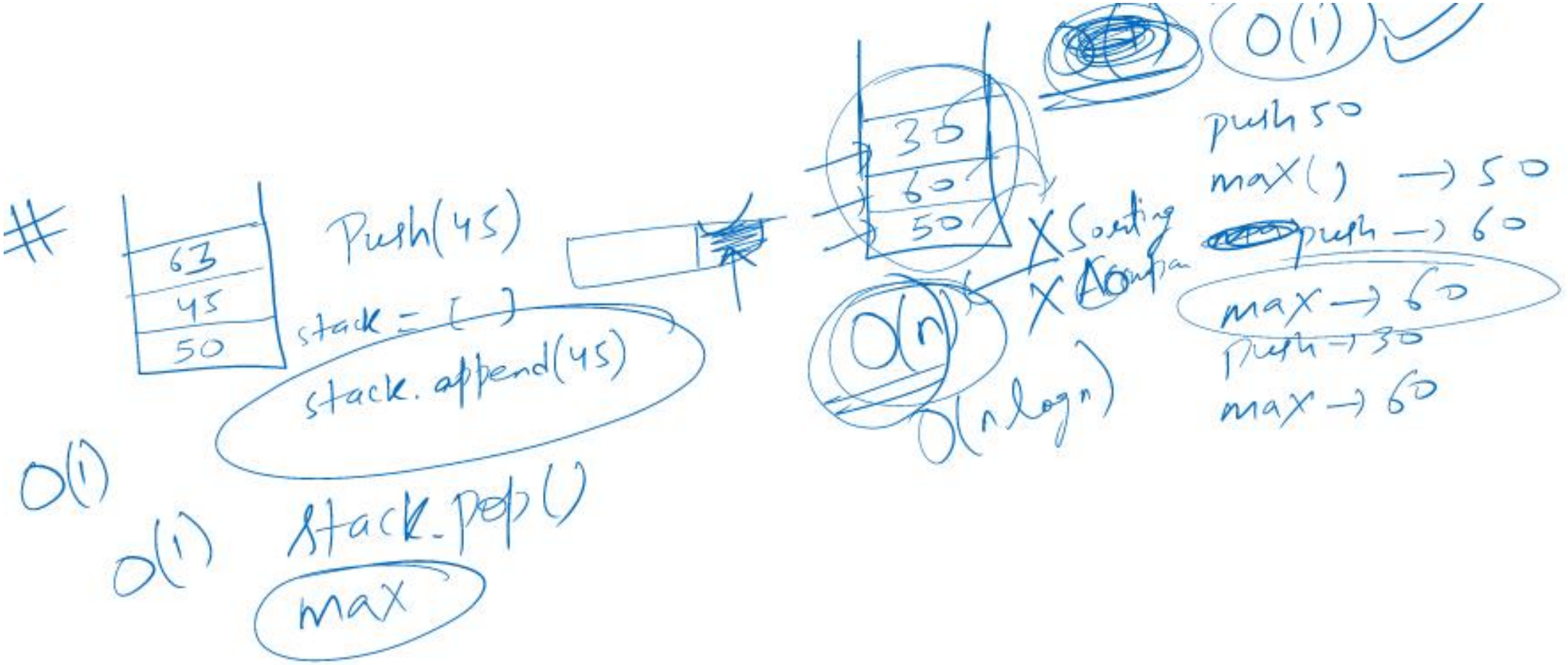

#Linked List
dict in Python
Key
Value
hashing
object in js
{JS: 2, PY: 3, NPY: 1}
hash(JS) = 2
head
head 2
hash(string)
integer
O(1)
Stack/Queue
string
Value

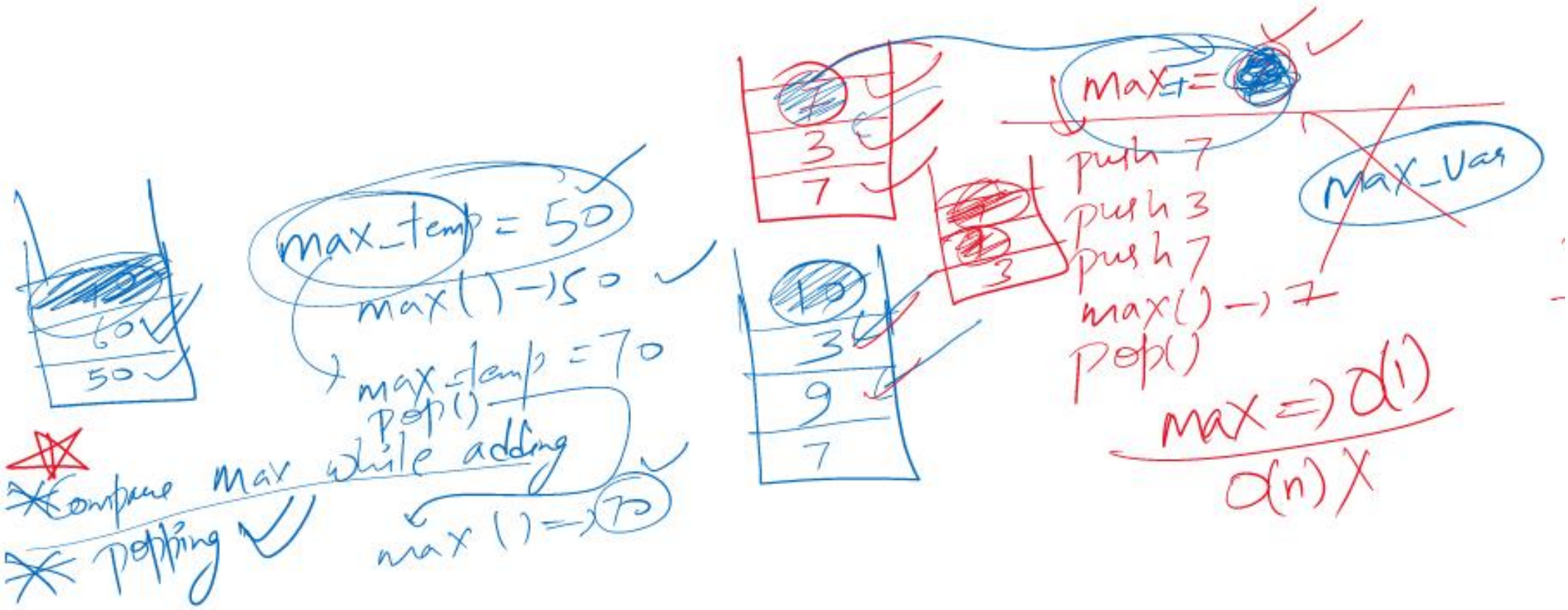
Graph Representation

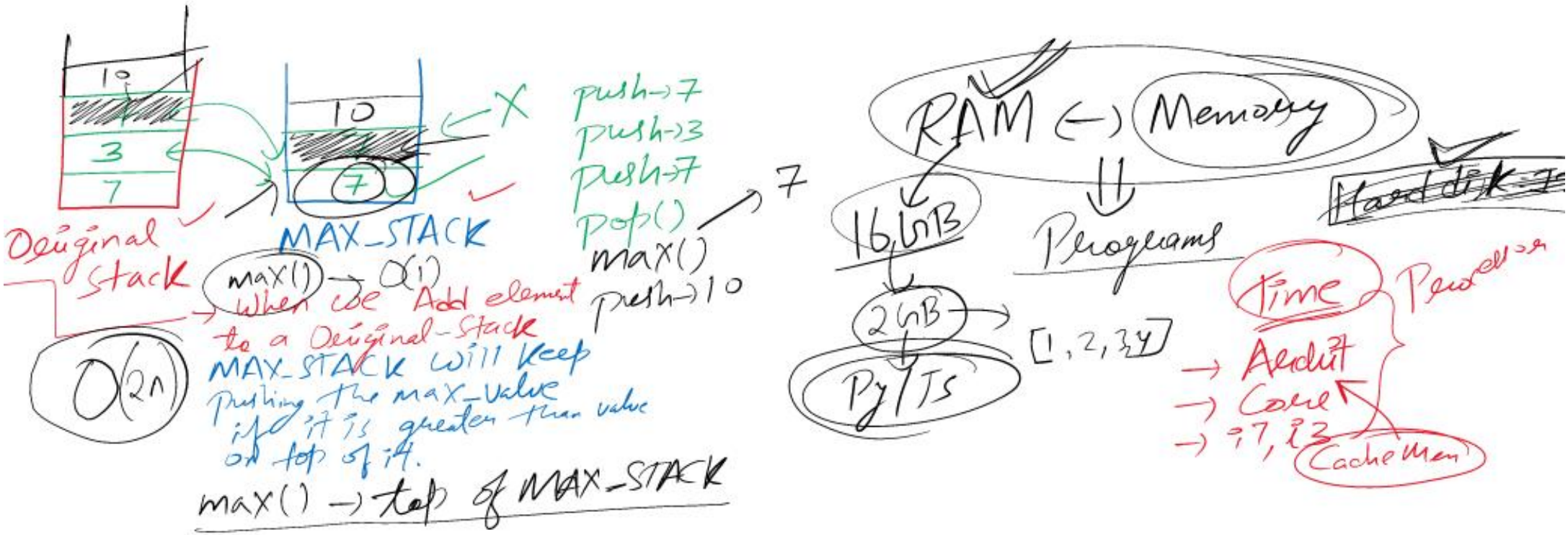







#
63
45
50
Push(45)
stack = [ ]
stack.append(45)
O(1)
O(1)
stack.pop()
max
30
60
50
X Sorting
X Compa
O(n)
O(nlogn)
O(1)
push 50
max() → 50
push → 60
max → 60
push → 30
max → 60


60
50
max_temp = 50
max() -> 50
max_temp = 70
pop()
max() -> 70
3
7
3
9
7
3
push 7
push 3
push 7
max() -> 7
pop()
max_var
max => O(1)
O(n) X
* Compare max while adding
* popping


10
3
7
Original Stack
O(2n)
10
7
MAX_STACK
max() → O(1)
When we Add element to a Original-Stack MAX_STACK will keep Pushing the max_value if it is greater than value on top of it.
max() → top of MAX_STACK
X
push→7
push→3
push→7
pop()
max()
push→10
7
RAM ← Memory
16GB
2GB
Py/Js
Programs
[1,2,3,4]
Hard disk
Time
Process
→ Kernel
→ Core
→ i7, i3
Cache Mem