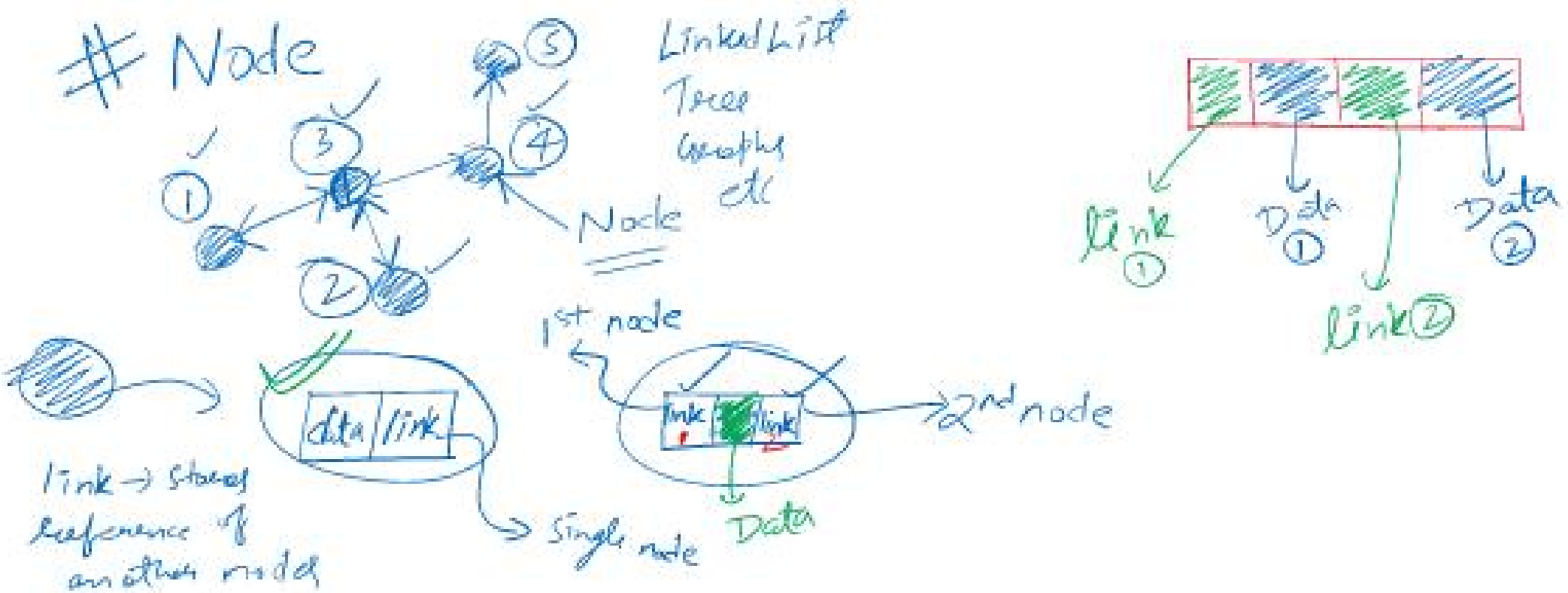
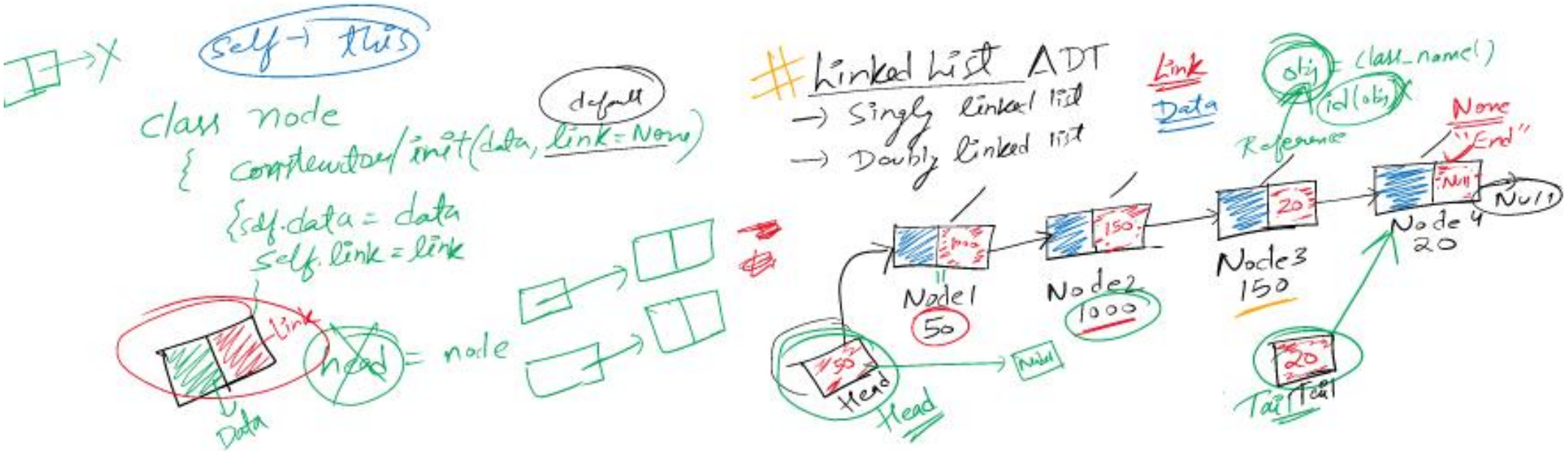
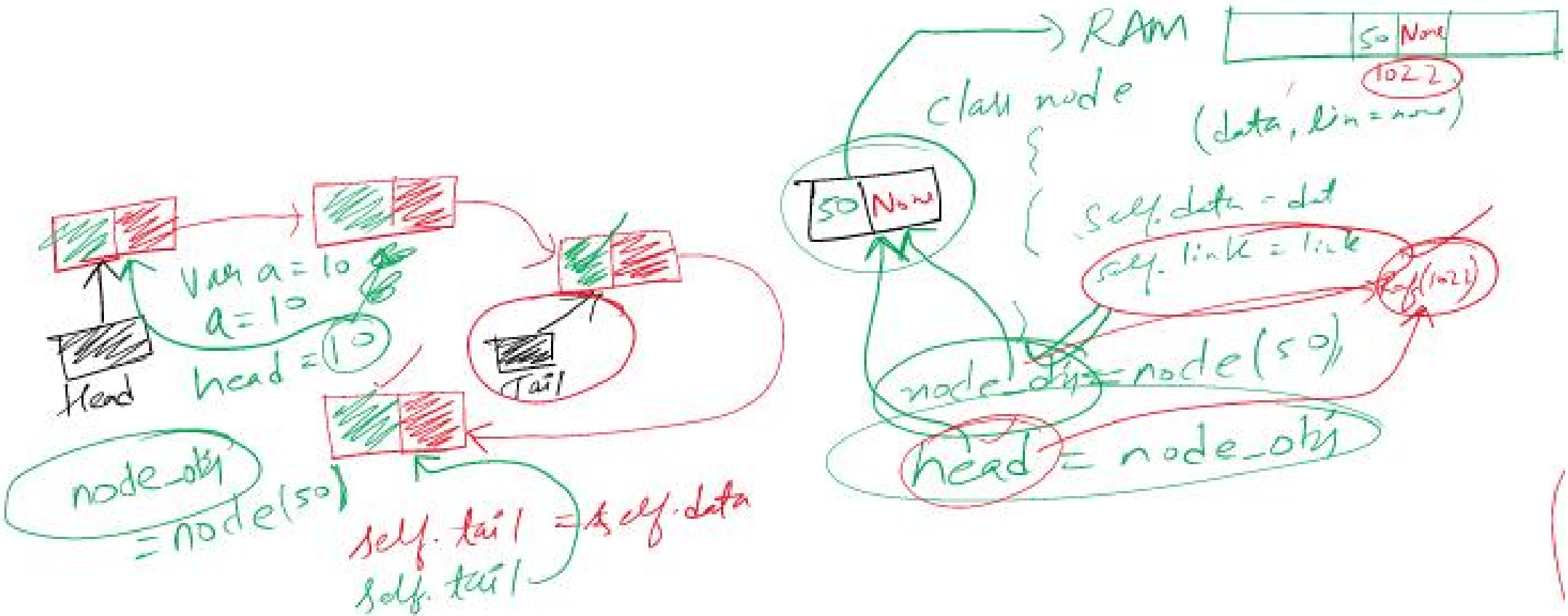
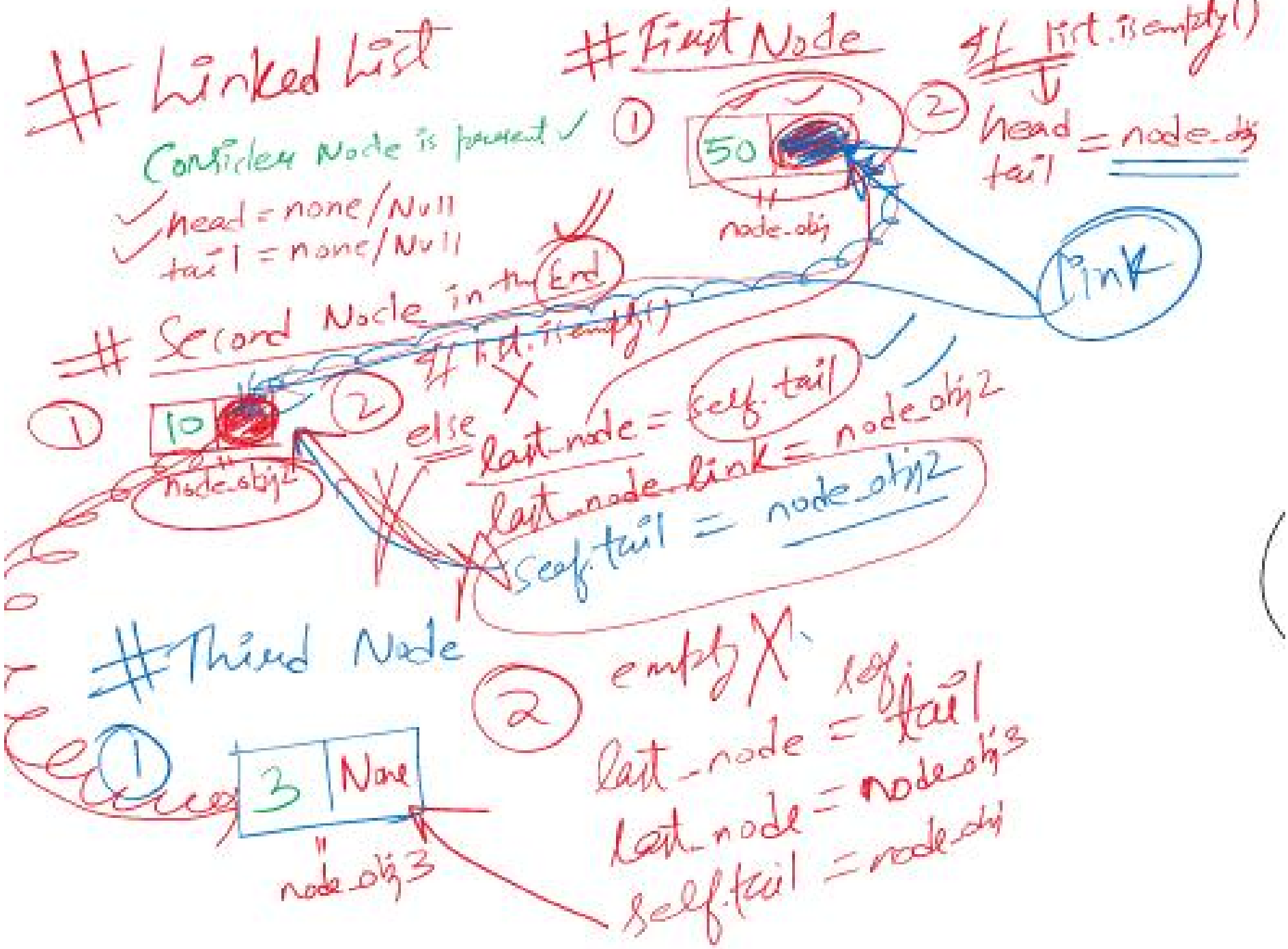
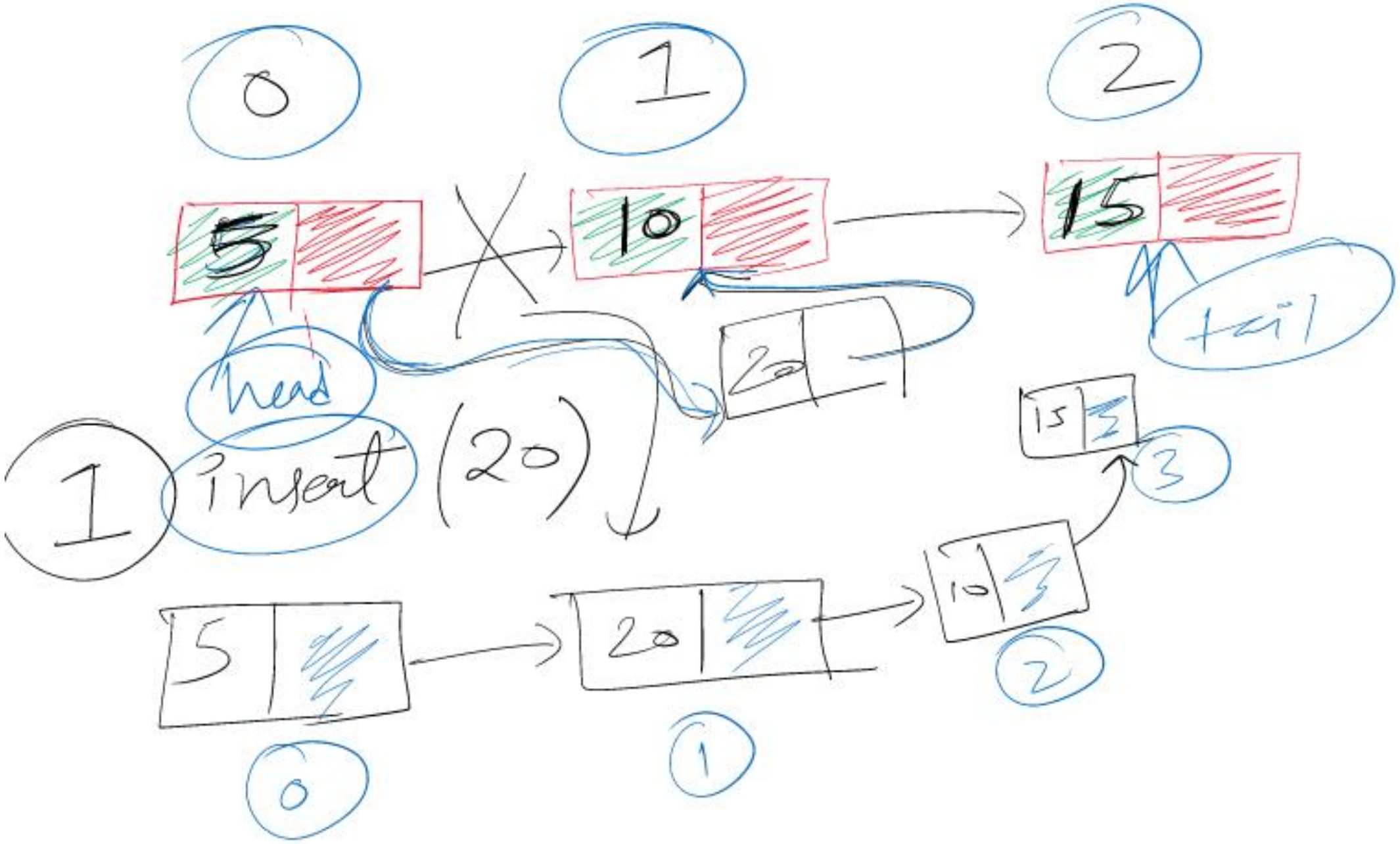
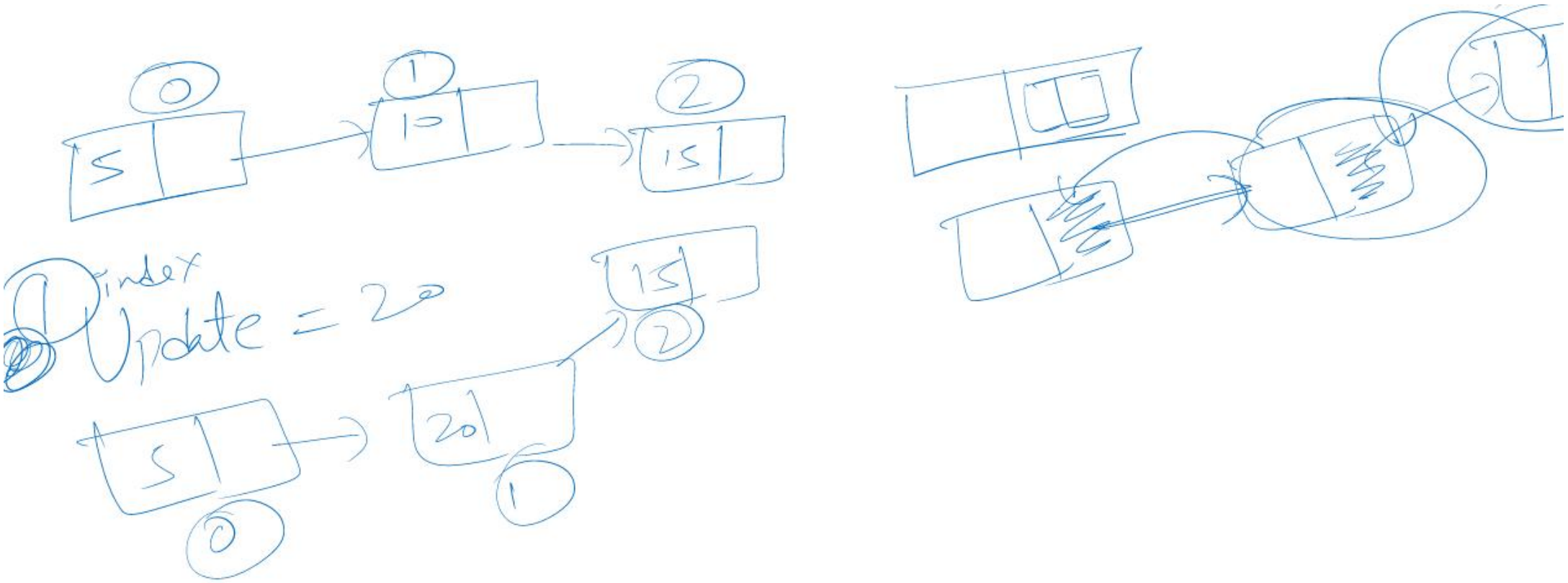

Node
LinkedList
Node
1st node
2nd node
data
link
link → Stores reference of another node
Single node
Data
Link
Data
Data
Link
Link

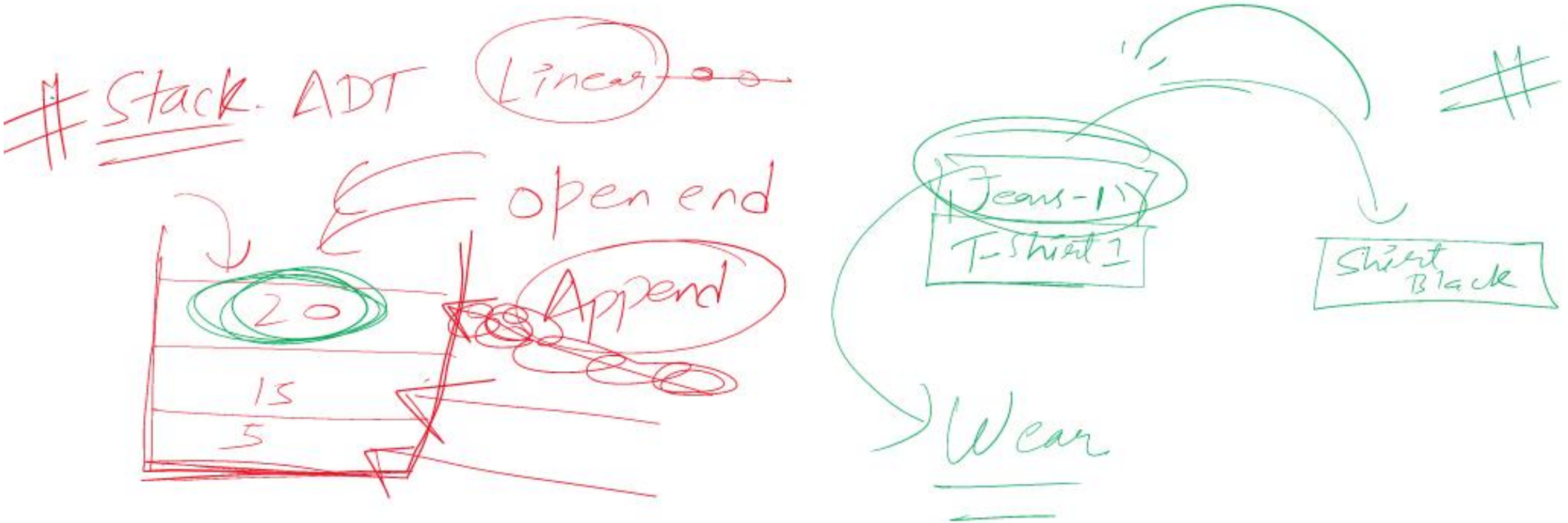

Self → this
class node
{ constructor/init(data, link=None)
default
{self.data = data
self.link = link
link
data
head = node
# Linked list ADT
→ Singly linked list
→ Doubly linked list
Link
Data
obj = class_name()
id(obj)
Reference
None
"End"
Node1
50
Node2
1000
Node3
150
Node4
20
Null
Head
Tail

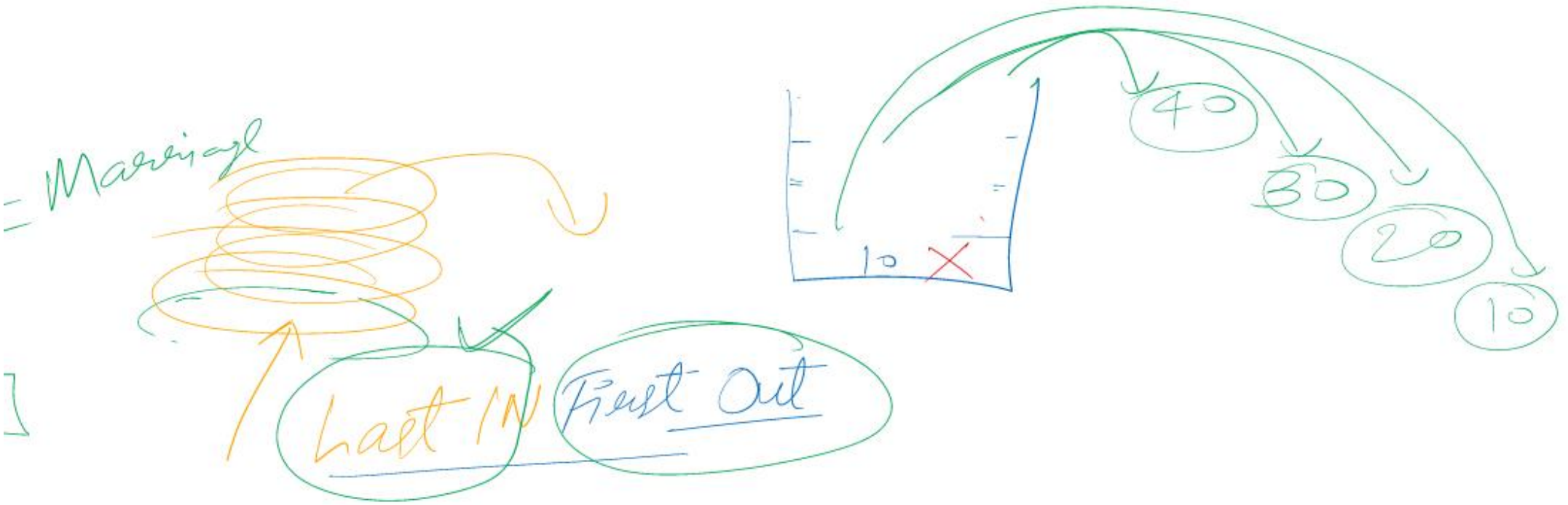

RAM
50
None
1022
class node
(data, link = none)
50
None
self.data = data
self.link = link
self(1022)
node_obj = node(50)
head = node_obj
Head
Tail
Var a = 10
a = 10
head = 10
node_obj
= node(50)
self.tail = self.data
self.tail

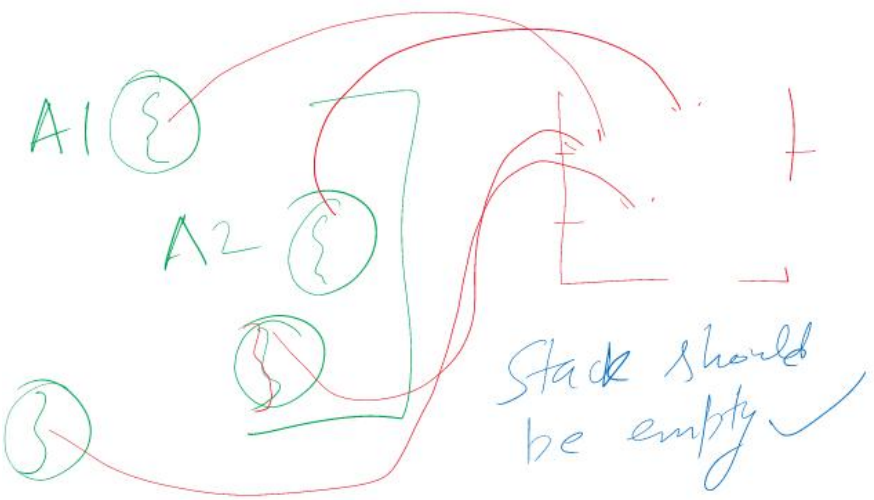

Linked list
Consider Node is passed
head = none/Null
tail = none/Null
# First Node
if list.isempty()
head = node_obj
tail
50
node_obj
Link
Second Node in the
End
if ll.isempty()
else
10
node_obj2
last_node = self.tail
last_node.link = node_obj2
self.tail = node_obj2
# Third Node
empty
last_node = self.tail
last_node = node_obj3
self.tail = node_obj3
3
None
node_obj3

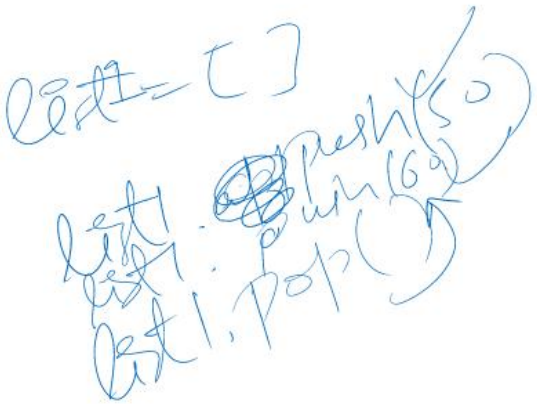

0
1
2
5
10
15
20
head
tail
1
insert
(20)
15
2
3
5
20
10
2
0
1

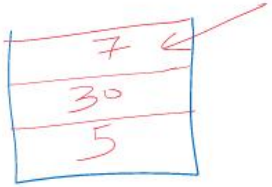

0
5
1
10
2
15
Index
Update = 20
15
2
5
20
0
1

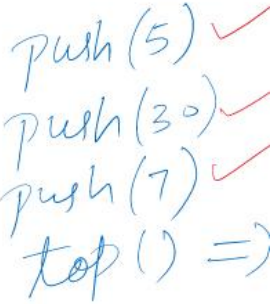

Stack. ADT
Linear
open end
Append
20
15
5
Jeans-1
T-Shirt 1
Shirt
Black
Wear


Marriage
Last IN
First out
40
30
20
10
10


A1
A2
Stack should be empty


left = [ ]
left.push(5)
left.push(6)
left.pop()
left.pop()


7
30
5


push(5)
push(30)
push(7)
top() =>