Longest subarray with equal freq of 0's, 1's & 2's.

arr[] = { 1, 1, 0, 0, 1, 0, 1, 2, 1, 2, 2, 0, 1 }

freq 0 → 1
freq 1 → 1
freq 2 → 1

Brute force:

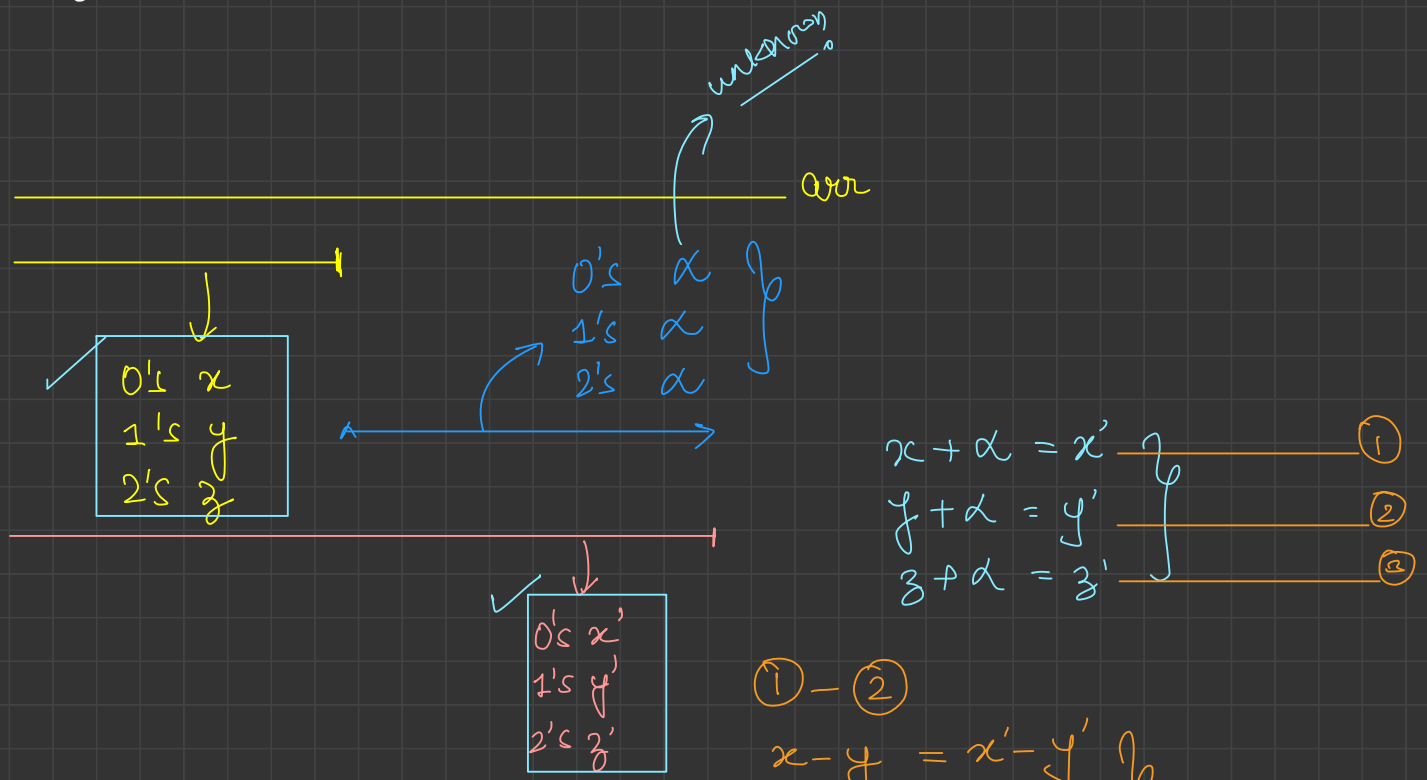
→ Calc. all subarray,

int cnt0
int cnt1
int cnt2

TC: $O(N^2)$

SC: $O(1)$

arr[] = { 1, 1, 0, 0, 1, 0, 1, 2, 1, 2, 2, 0, 1 }



$x + \alpha = x'$ ①

$y + \alpha = y'$ ②

$z + \alpha = z'$ ③

① − ②

$x - y = x' - y'$

② − ①

$y - z = y' - z'$

arr[] = { 1, 1, 0, 0, 1, 0, 1, 2, 1, 2, 2, 0, 1 }

(x)	0	0	0	0	1	2	2	3	3	3	3	3	3
(y)	1	0	1	2	2	2	3	3	4	4	5	5	5
(z)	2	0	0	0	0	0	0	0	0	1	1	2	3
(x-y)		0	-1	-2	-1	0	-1	0	-1	-1	-2	-2	-2
(y-z)		0	1	2	2	2	3	3	4	3	4	3	2

0#0 -1#1 -2#2 2#2

value

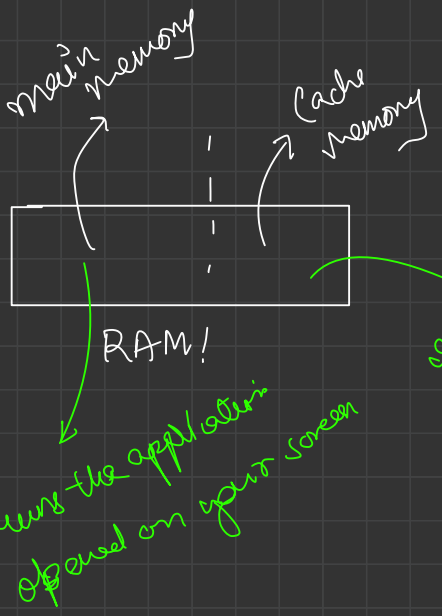
key = value

encoding = (x-y) # (y-z)

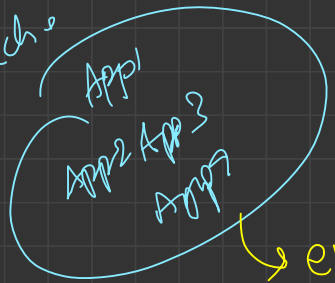
$x - y = x' - y'$ ④

$y - z = y' - z'$ ⑤

LRU Cache

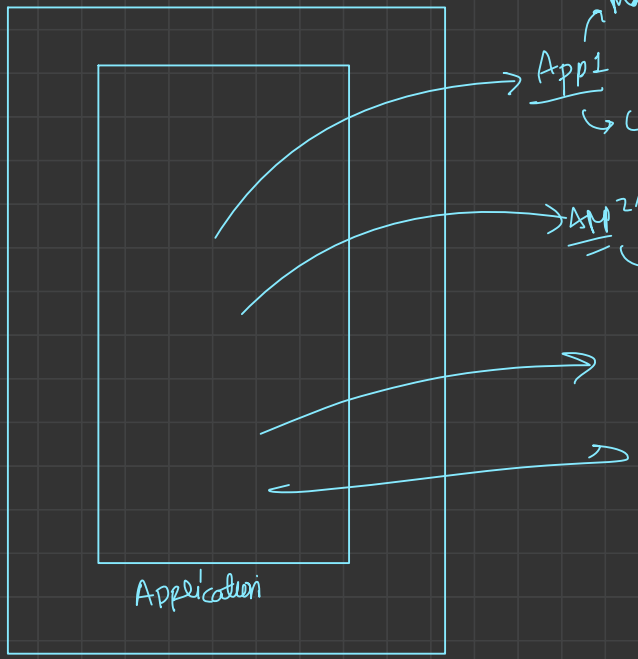


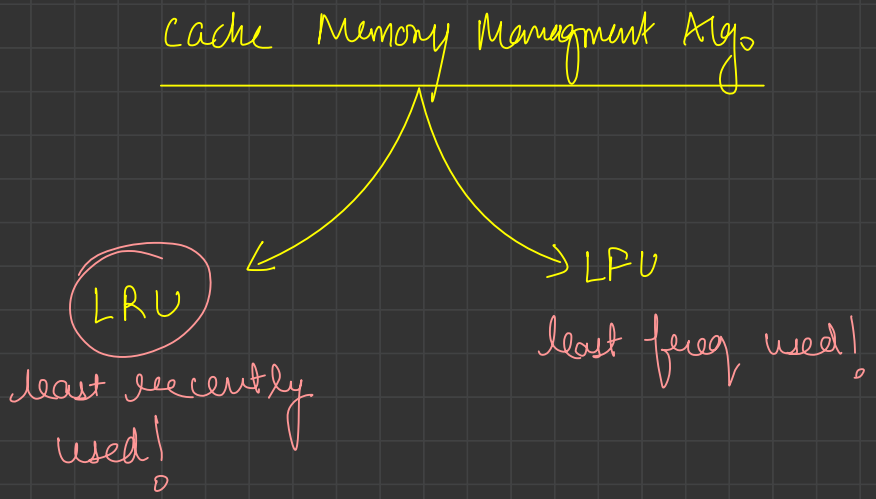
Cache:



↳ eventually all space will be taken by things running in back

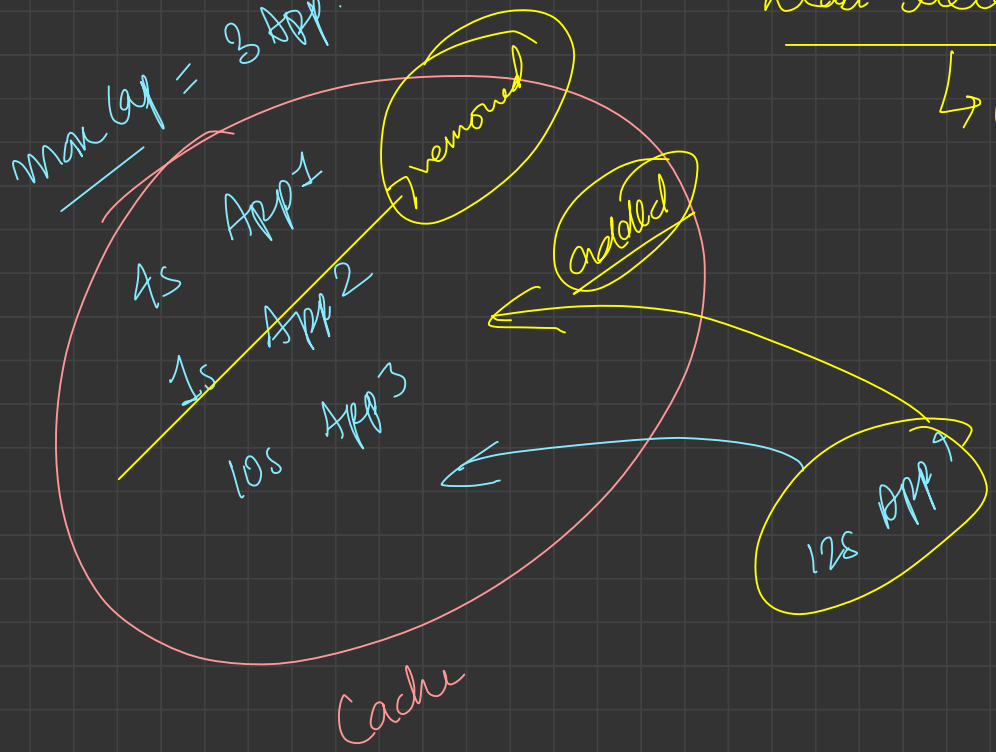
Mobile phone

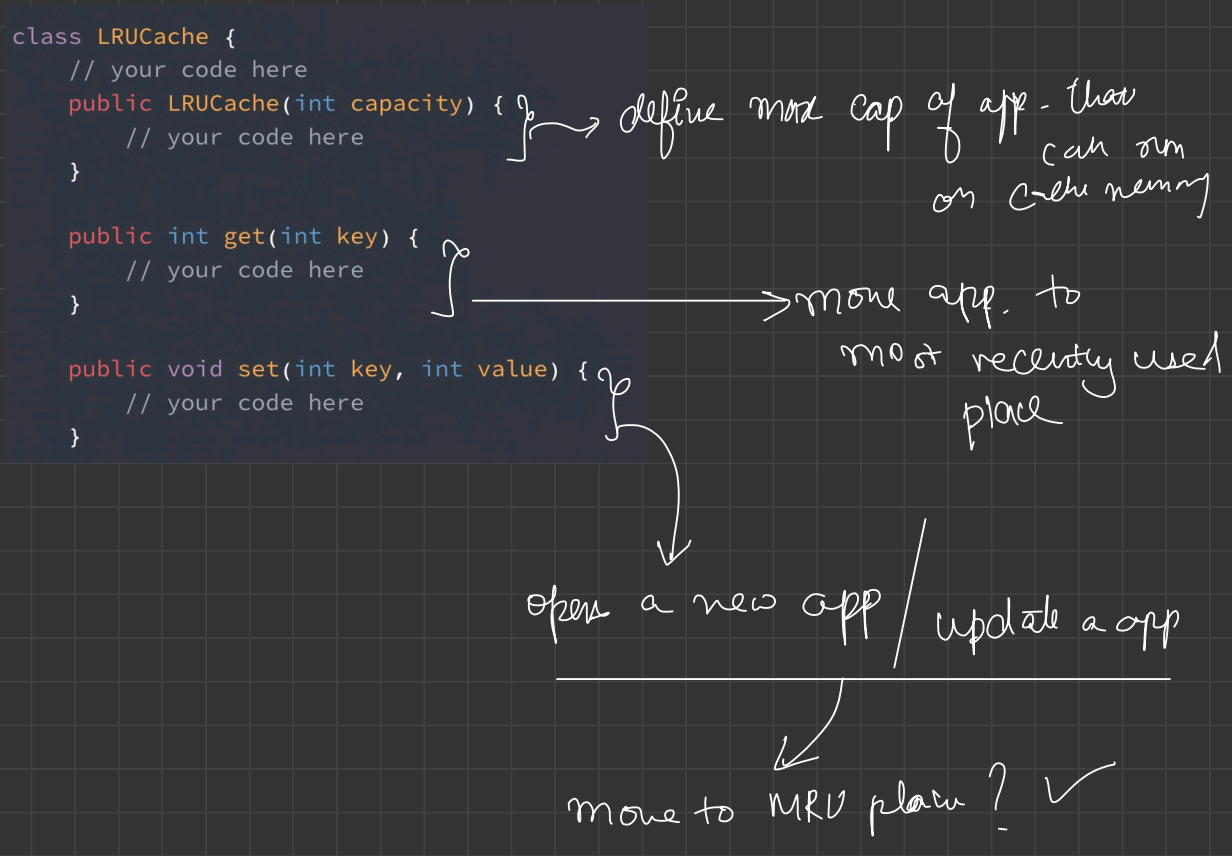


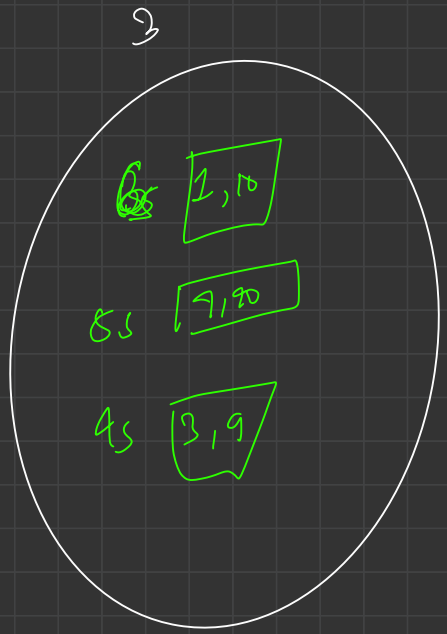

Cache Memory Management Alg.
LRU
least recently used!
LFU
least freq used!

Least recently used

↳ Cache Memory Mgmt.!




class LRUCache {
// your code here
public LRUCache(int capacity) {
// your code here
}
public int get(int key) {
// your code here
}
public void set(int key, int value) {
// your code here
}
define max cap of app - that can run on cache memory
move app. to most recently used place
open a new app / update a app
move to MRU place ?



set (1, 10)

set (2, 300)

get (1) → 10

set (3, 9)

set (4, 90)

set (1, 100)

Cache Memory.

set (1, 10)

set (2, 300)

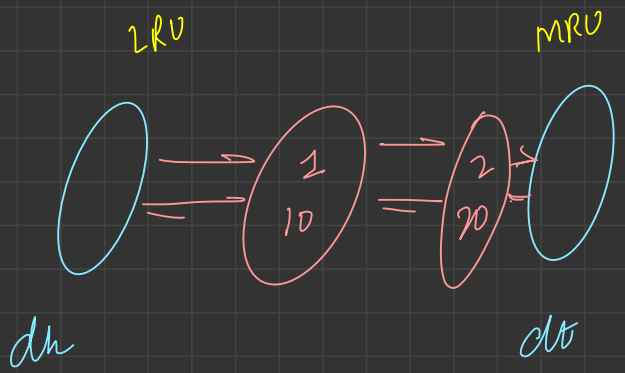
get (1) ⇝ 10

set (3, 9)

set (4, 70)

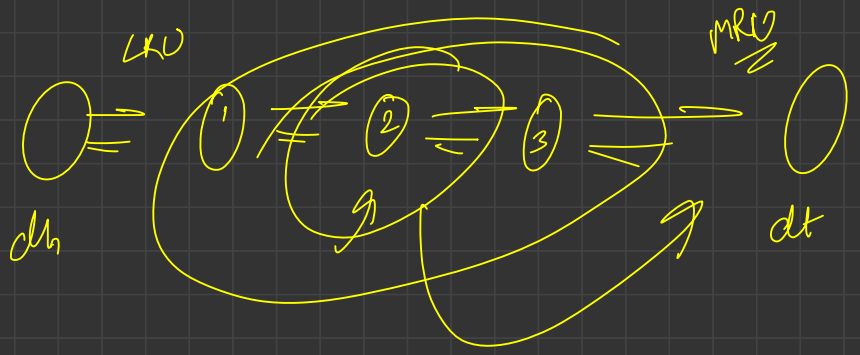
set (1, 100)

doubly linked list



HashMap

key ⟶ add



Capacity

hashmap

2 → dll

Snapshot Array

int[] arr = { 0, 0, 90, 40, 0, 0, 0 } (0 1 2 3 4 5 6)

snap_id = 0 1 2

set (2, 20)

set (3, 40)

snap()

set (2, 90)

snap()

get (3, 1) → 40

get (2, 0) → 20

snapId

0 { 0, 0, 20, 40, 0, 0, 0 } (0 1 2 3 4 5 6)

1 { 0, 0, 90, 40, 0, 0, 0 } (0 1 2 3 4 5 6)

0 1 2 3 4 5 6

int[] arr = { 0, 0, 0, 0, 0, 0, 0 }

snap_id = 0