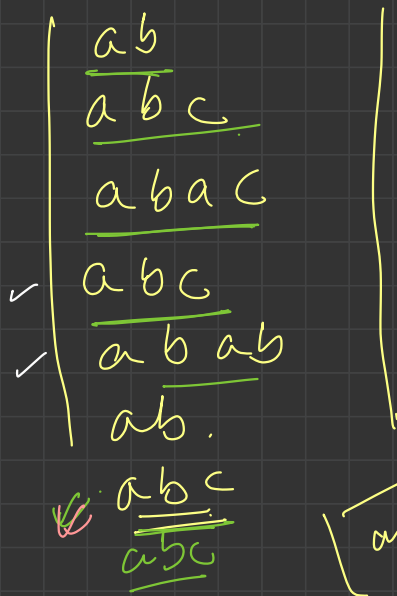
Registration

Andryusha and socks

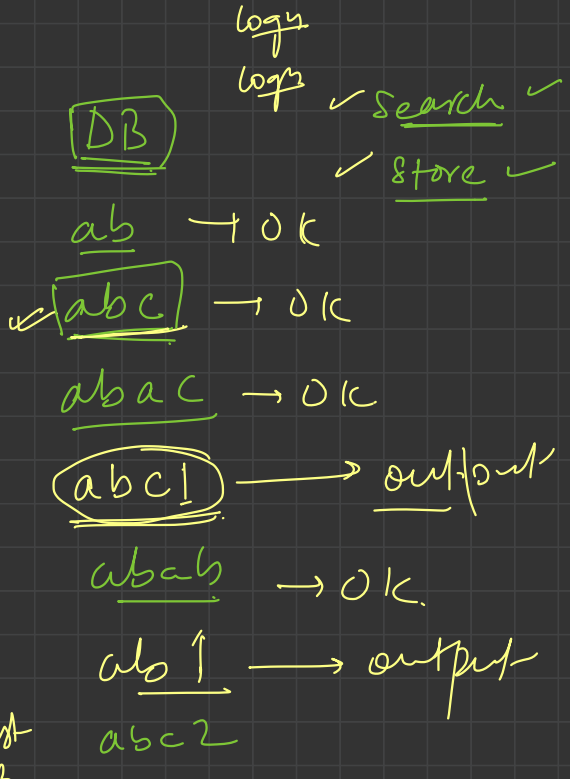
Second Order Statistics



abci

↳ non-negative integer

and i is min such that abci doesn't exist in DB



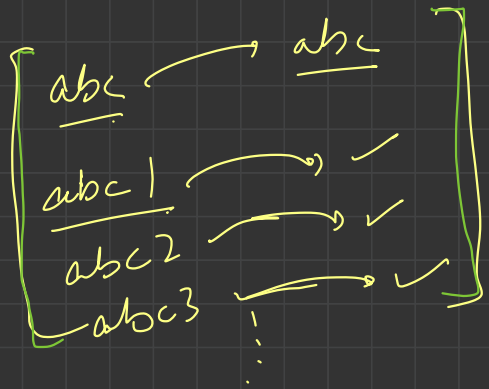
name → freq

ab → 2

abc → 1 2 3

abac

abab



abc + {freq} → count ⇒ abc1

(freq++)

ab + {1} ⇒ ab1

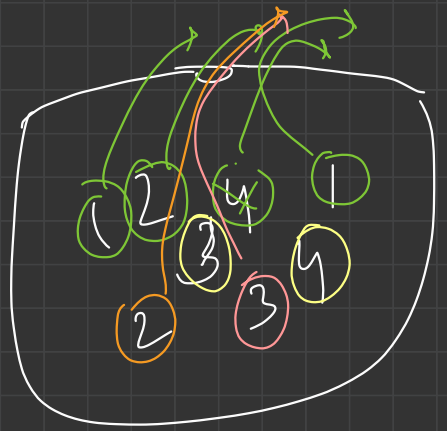
abc + {freq} ⇒ abc2

freq+1

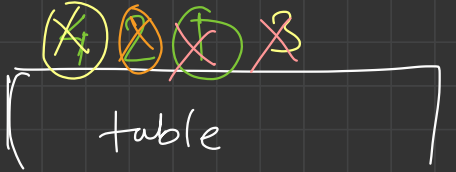
n=4

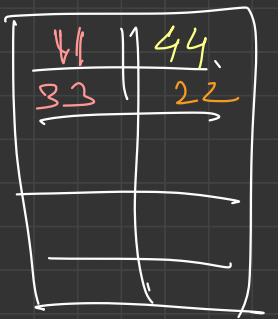
(1-n2)

2x



bag





wardrobe

- Merge sorted LLs
- Reorder
- ~~Design Linked List~~
- Deep copy

Add two numbers

cycle detection

~~linked list components~~

~~Odd even~~

~~Remove?~~

Lemon Change

Lab evaluation

~~Interview point~~ LL

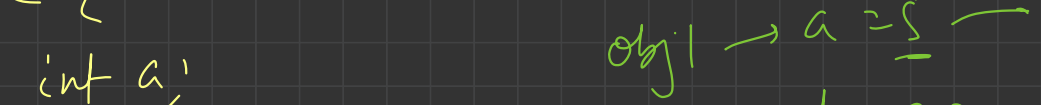
Shallow copy ⇒ bit-wise copy

Deep copy

C obj1;

shallow copy

C obj2 = obj1



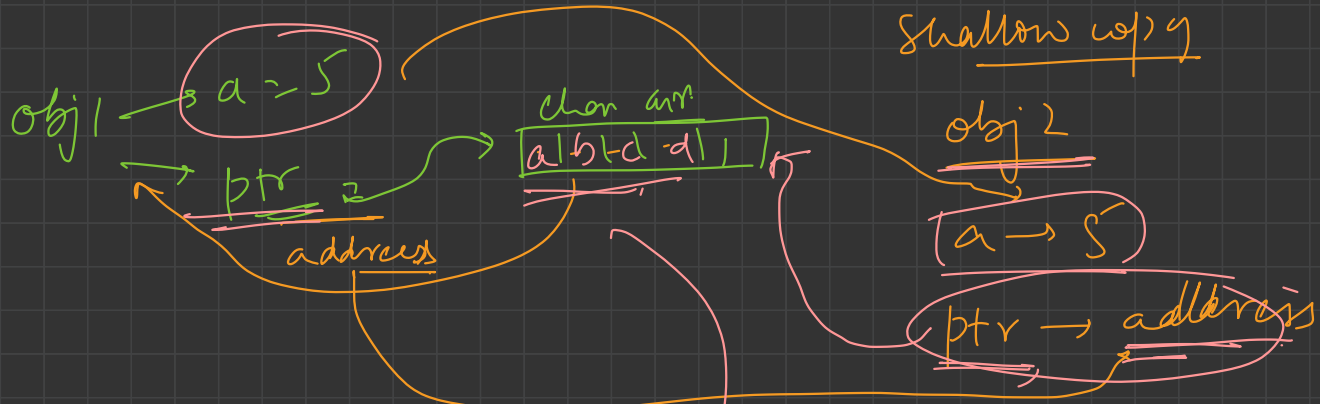

```
C {
int a;
double d;
}
```

obj2 → a = 5
→ d = 98

C{
int a;
char * ptr;
}

obj2 = obj1

Shallow copy



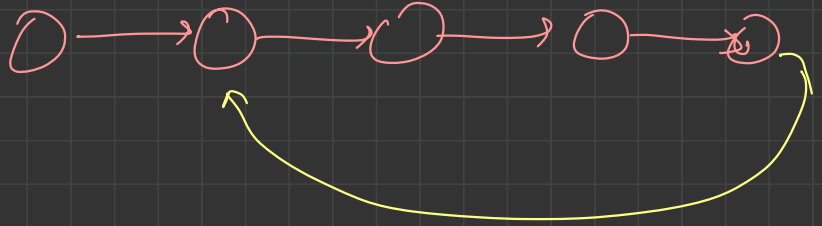
Deep copy ≠

void operator= ( ) {

ptr = new char[n]

}

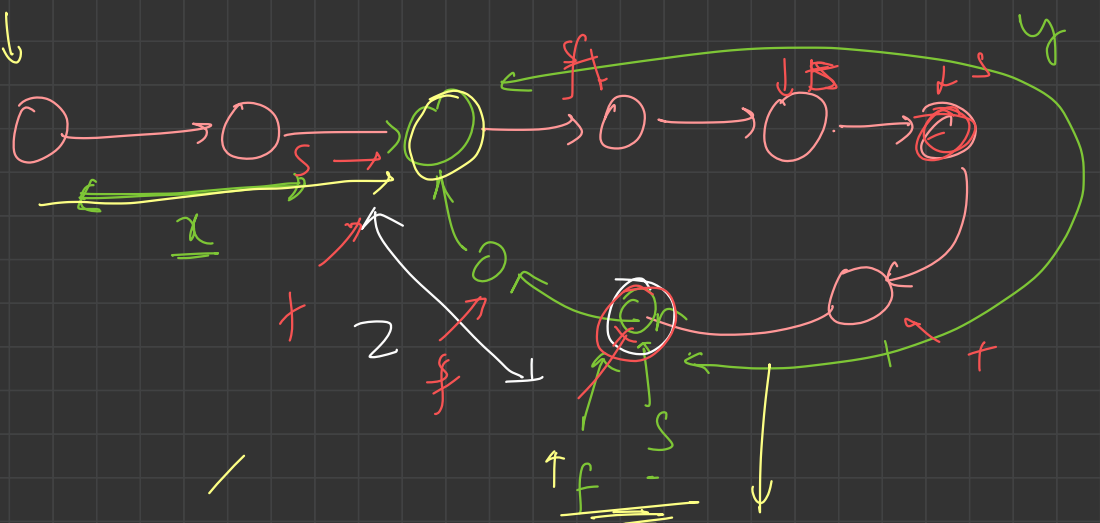
a = b



Cycle Detection

① Mark the nodes and if found a node that is already in the map/set ⇒ cycle,
else (if null node) ⇒ no cycle

② Floyd - cycle detection algorithm.
↳ slow & fast pointer



at meet point dist covered

slow

$x+y$

+ cycle length

fast

$+y+z)+y$

→ should $2(x+y)$

+ 2 × cycle length

$x+y+z+y = 2(x+y)$

$x+y+z = 2x+2y$

$x = z$

at meetup slow has gone through cycle ⇒ p-times
fast ⇒ q-times

slow: $x + (y+z) \times p + y$

fast: $x + (y+z) \times q + y = 2(x + (y+z)p + y)$

$$2(x + (y+z)p + y) = x + (y+z)q + y$$

$$2x + (y+z)(2p) + 2y = x + (y+z)q + y$$

$$x = (y+z)q - (y+z)(2p) - y$$

$$x = (y+z)(q-2p) - y$$

$$= (y+z)(q-2p) - y - z + z$$

$$= (y+z)(q-2p) - (y+z) + z$$

$$x = (y+z)(q-2p-1) + z$$

cycle len

$$x = m \cdot (y+z) + z$$

no effect