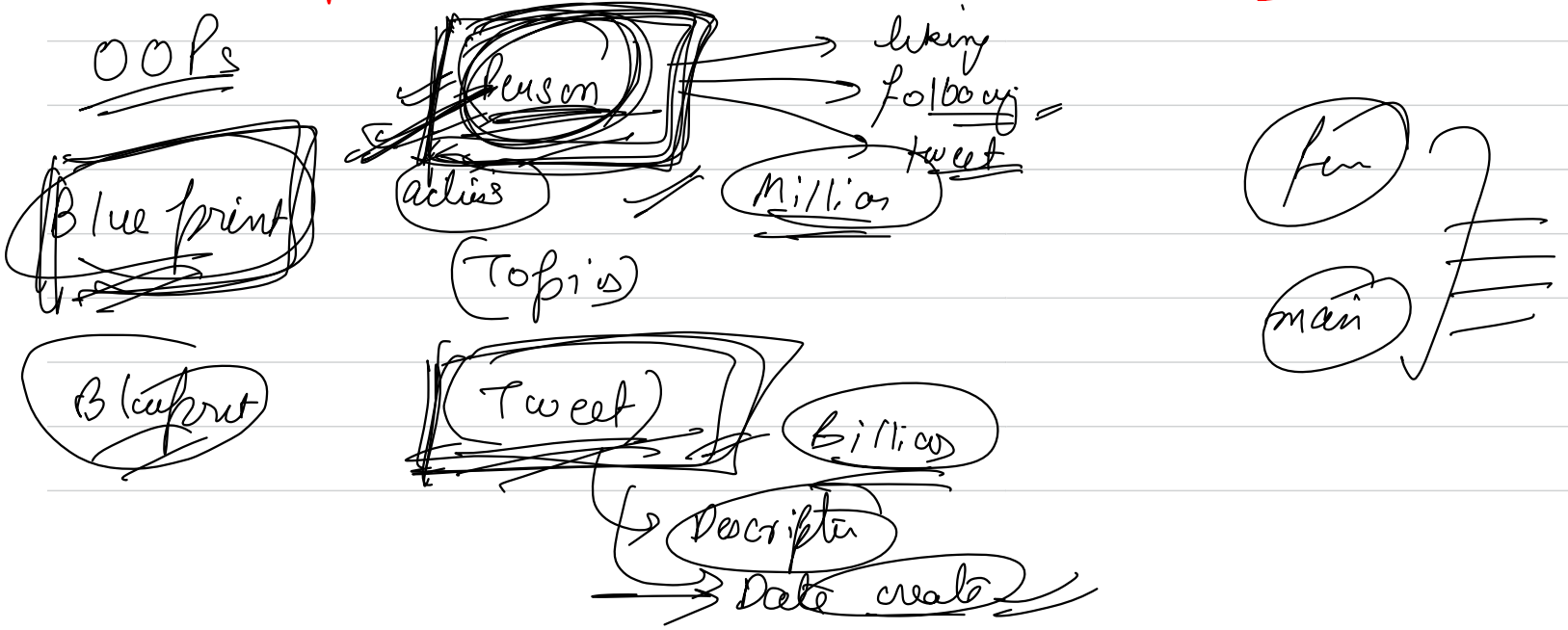
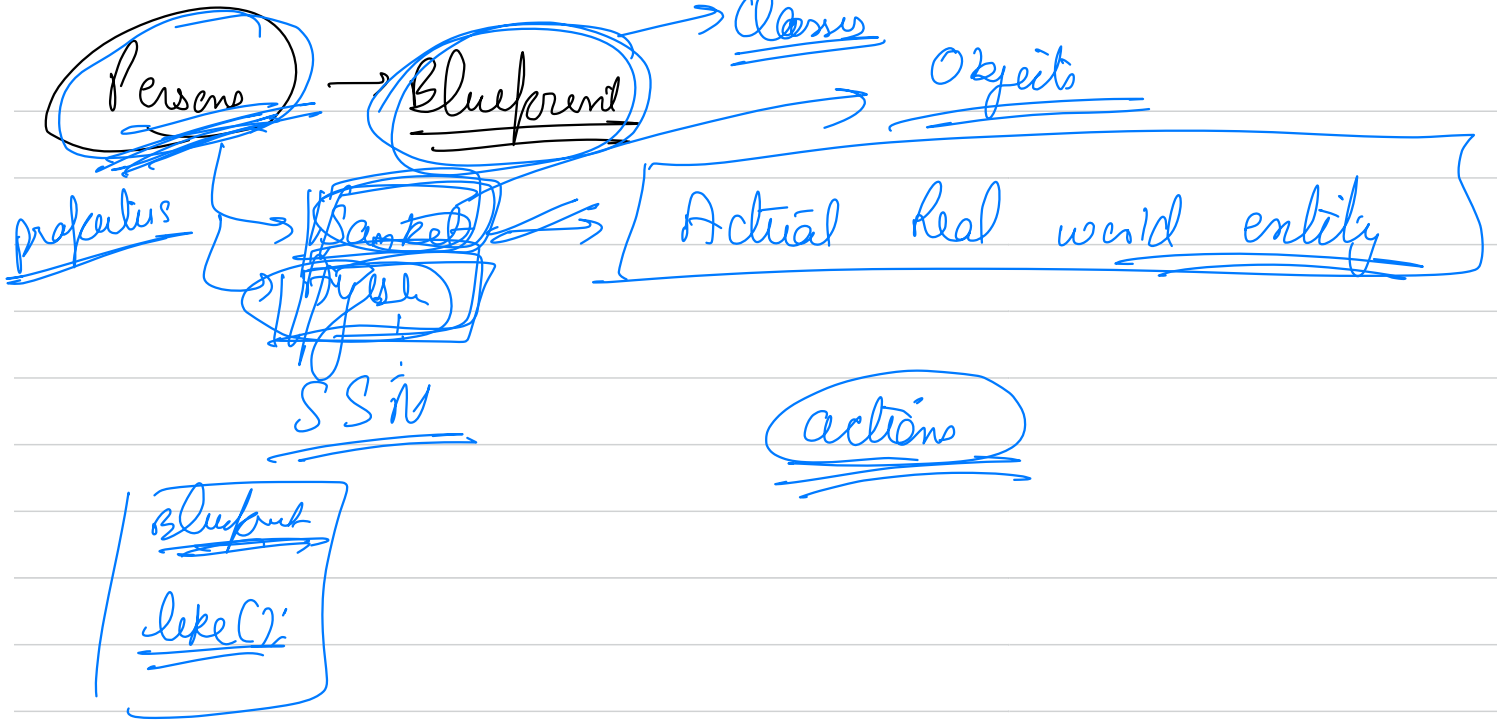
Good → Small talk

OOPs in C++ / STL

#C++ OOPs (Object Oriented Programming)

↳ We map real world entities into our code.




Person
Blueprint
Classes
Objects
properties
Actual Real world entity
SSN
actions
Blueprint
like C++

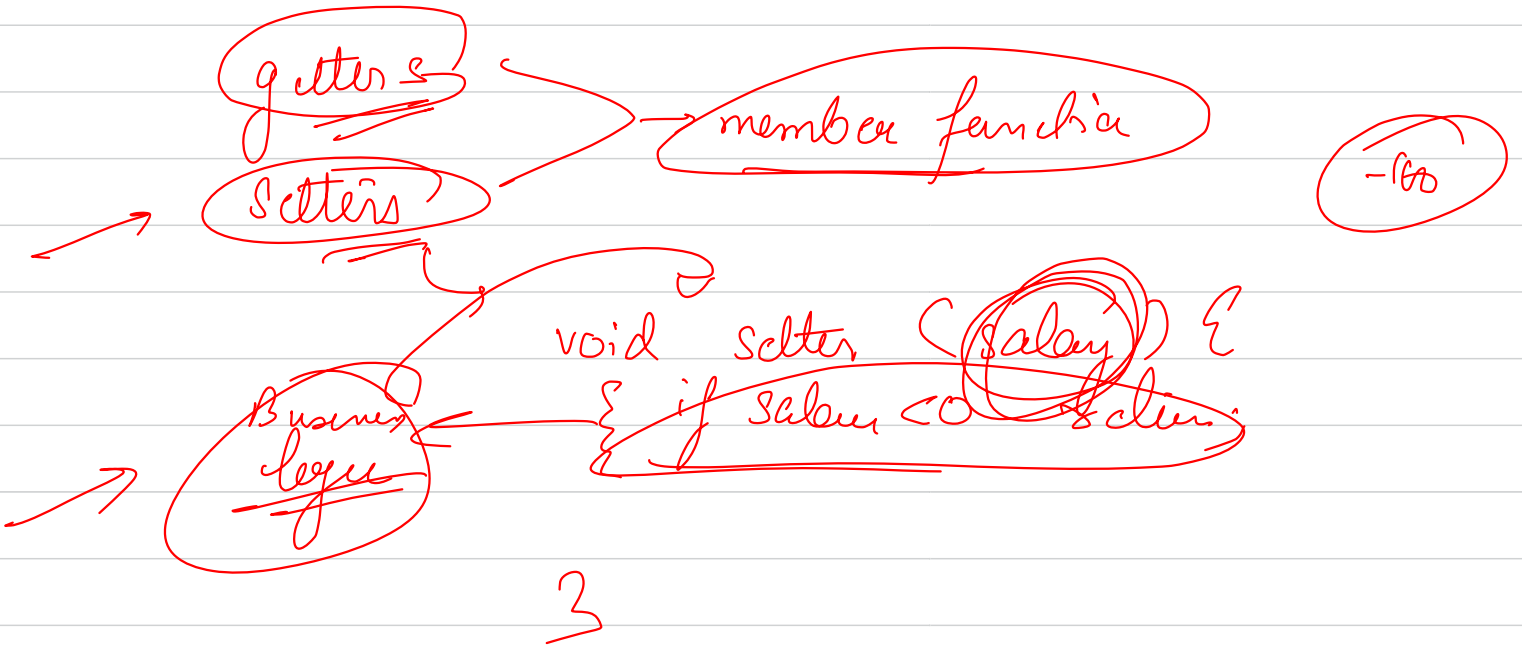
Object → Any real world entity that has a state & behaviour is called object.

Classes → Blueprint of collection of objects is called class.

Data hiding

We want to define salary at the time of creating object

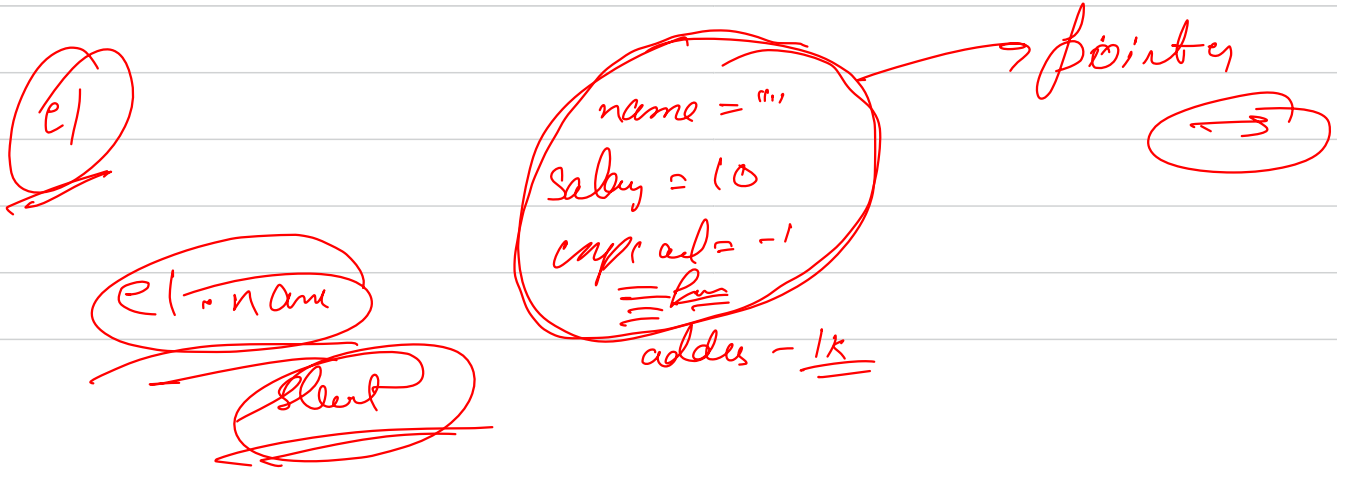
if you want to later update there are some constraints

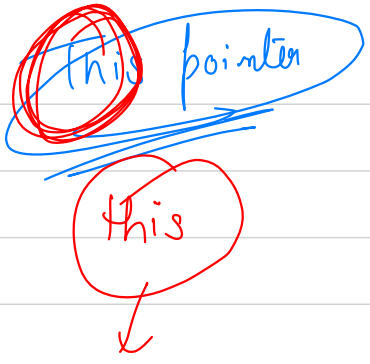

getters
setters
member functions
-100
void setter (salary) {
if salary < 0 setter
Business logic

3

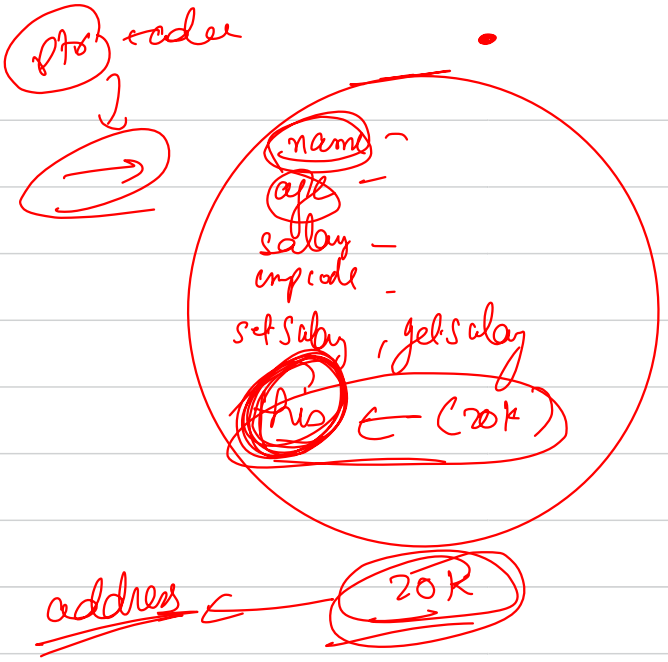
Constructors → new object memory

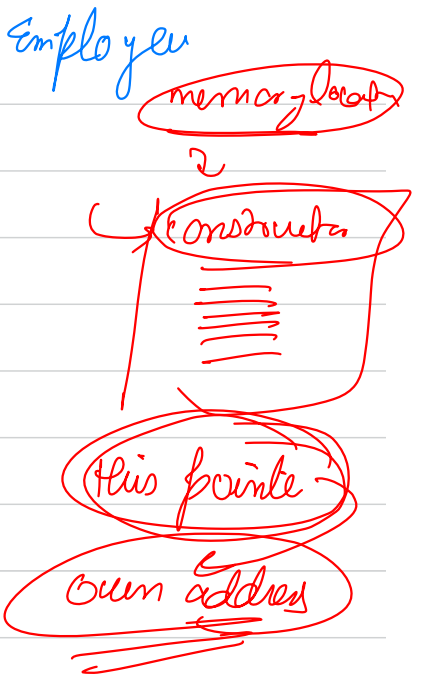
when u initialize an object, Constructor is the first member func" to be executed

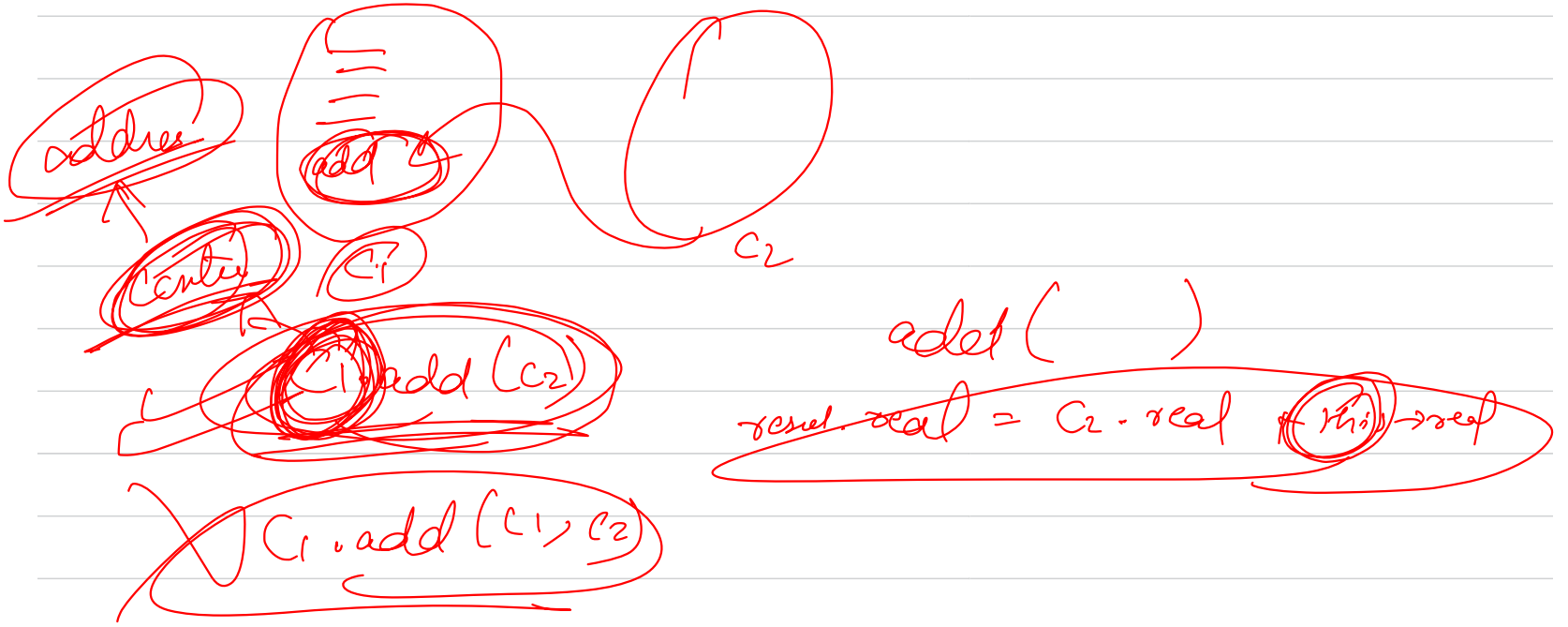




⇒ Stores the address of the object as a property in the object






add
c1
c2
c1.add(c2)
c1.add(c1, c2)
add( )
result.real = c2.real + this->real

Standard Template Library

DataStructs & Algorithms

- vectors
- maps
- pq
- list
- stack
- queue
- unordered_map
- ⋮

- lower_bound, upper_bound → BS
- sort
- next_permutation
- gcd

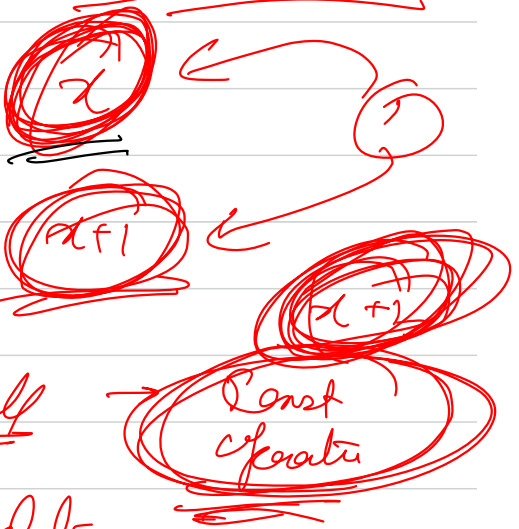
vectors

arrays

n

runtime → we can't grow or shrink the array

new array

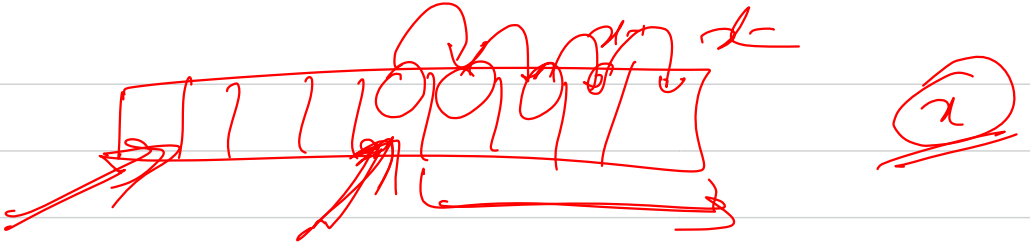


they grow or shrink automatically

runtime

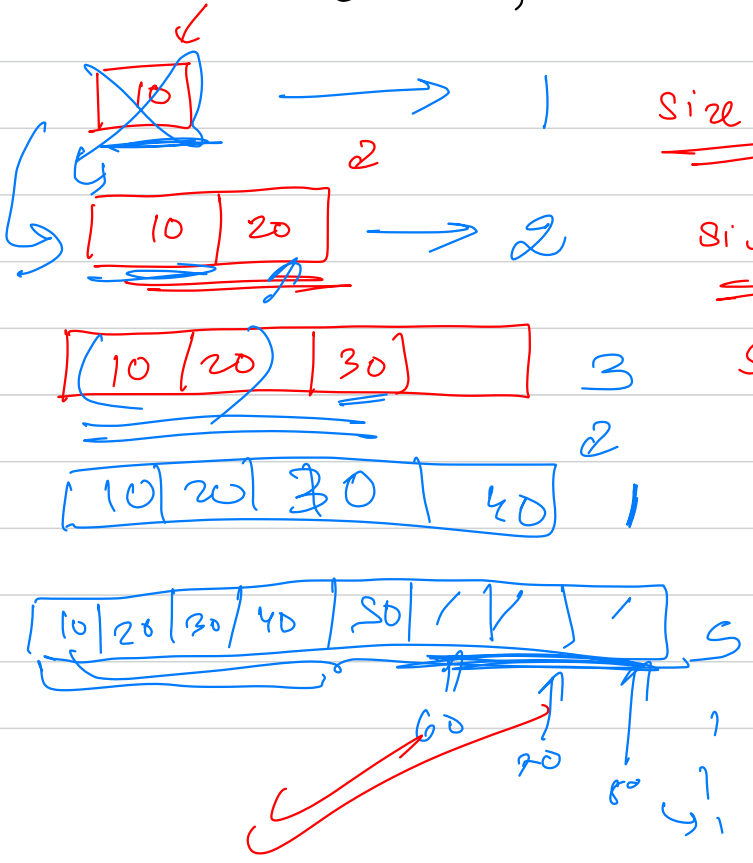
efficiently → easily

work efficiently if we add/delete from last

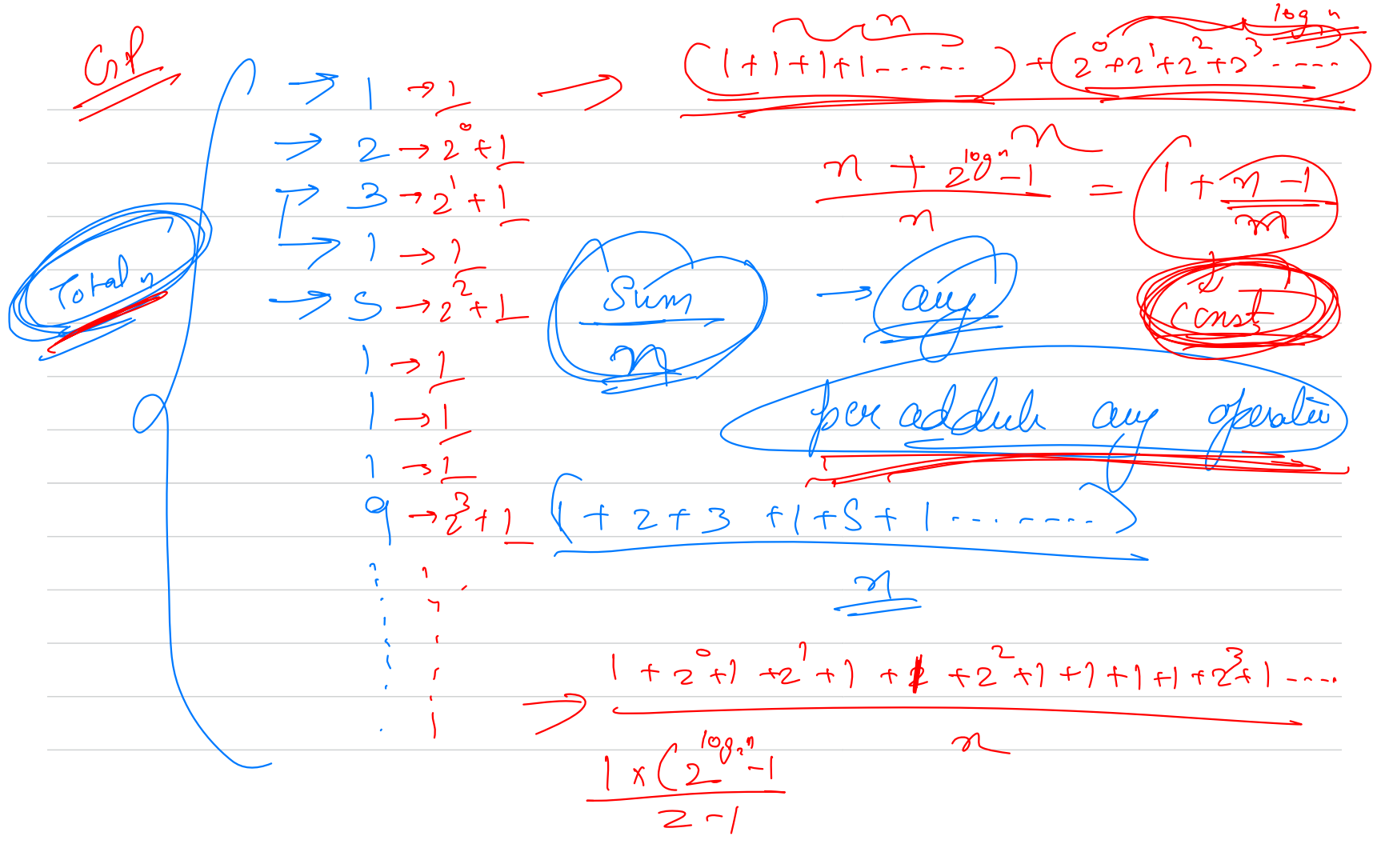


10, 20, 30, 40, 50, 60, 70, 80, 90, 70

Vectors → if at any point of time vector can't add new element, it will double its size



Space-time tradeoff


Ip
Total n
1 → 1
2 → 2^0 + 1
3 → 2^1 + 1
1 → 1
5 → 2^2 + 1
1 → 1
1 → 1
1 → 1
9 → 2^3 + 1
Sum
n
any
per addition any operation
1 + 2 + 3 + 1 + 5 + 1 ....
n
1 + 2^0 + 1 + 2^1 + 1 + 1 + 2^2 + 1 + 1 + 1 + 1 + 2^3 + 1 ....
n
1 x (2^(log2 n) - 1) / 2 - 1
(1 + 1 + 1 + 1 ....) + (2^0 + 2^1 + 2^2 + 2^3 ....)
n
log n
n + 2^(log n) - 1 / n = 1 + n - 1 / n
1
const

Total
time

$2^k + 1 = n$

$2^k \approx n$

$k = \log_2 n$