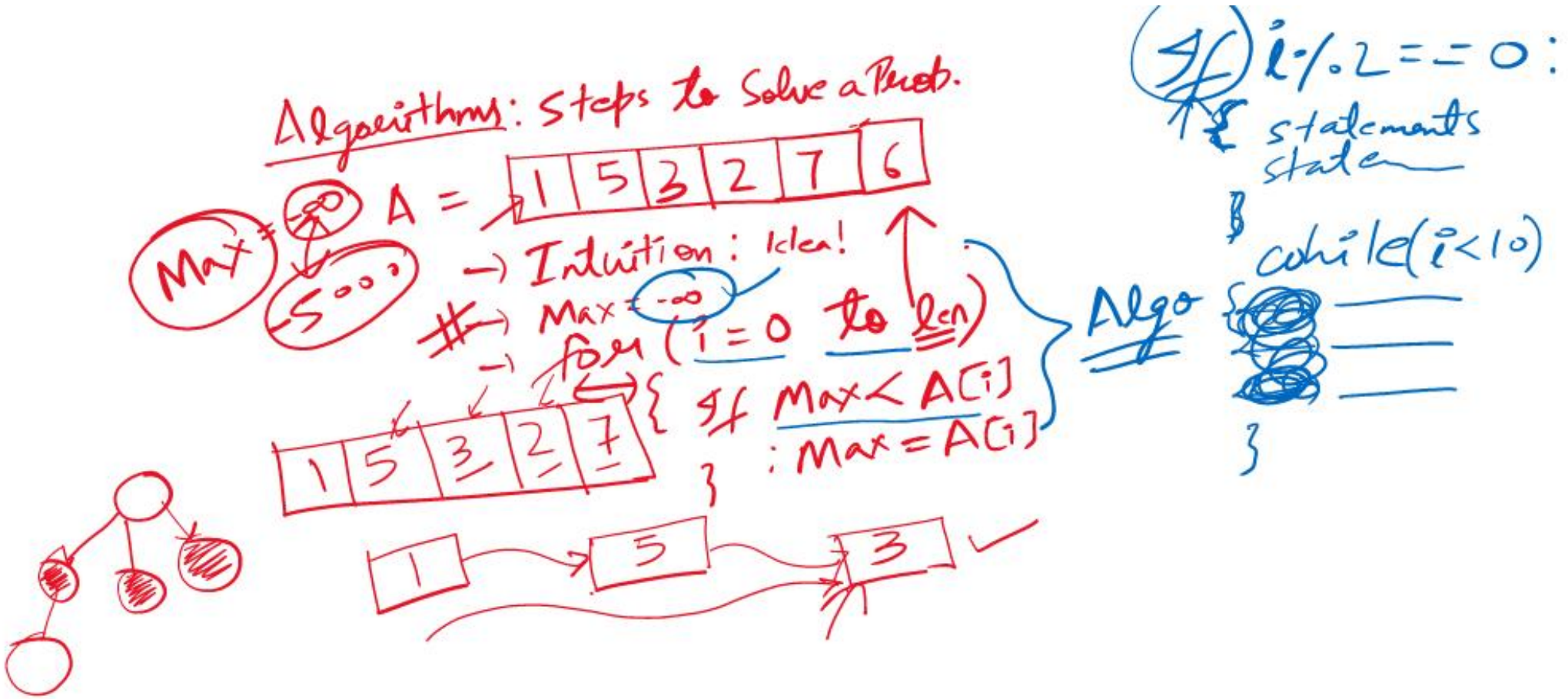
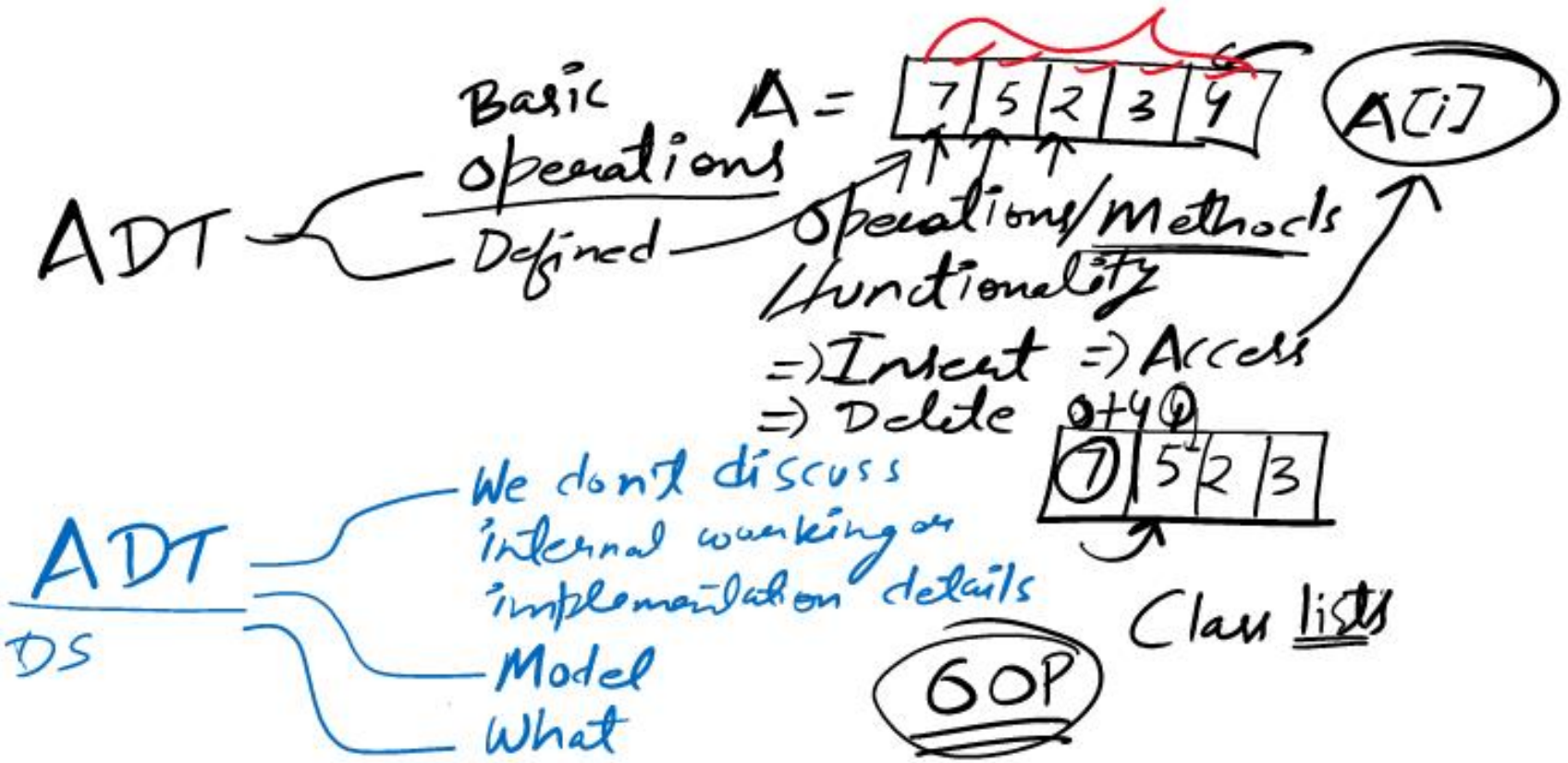
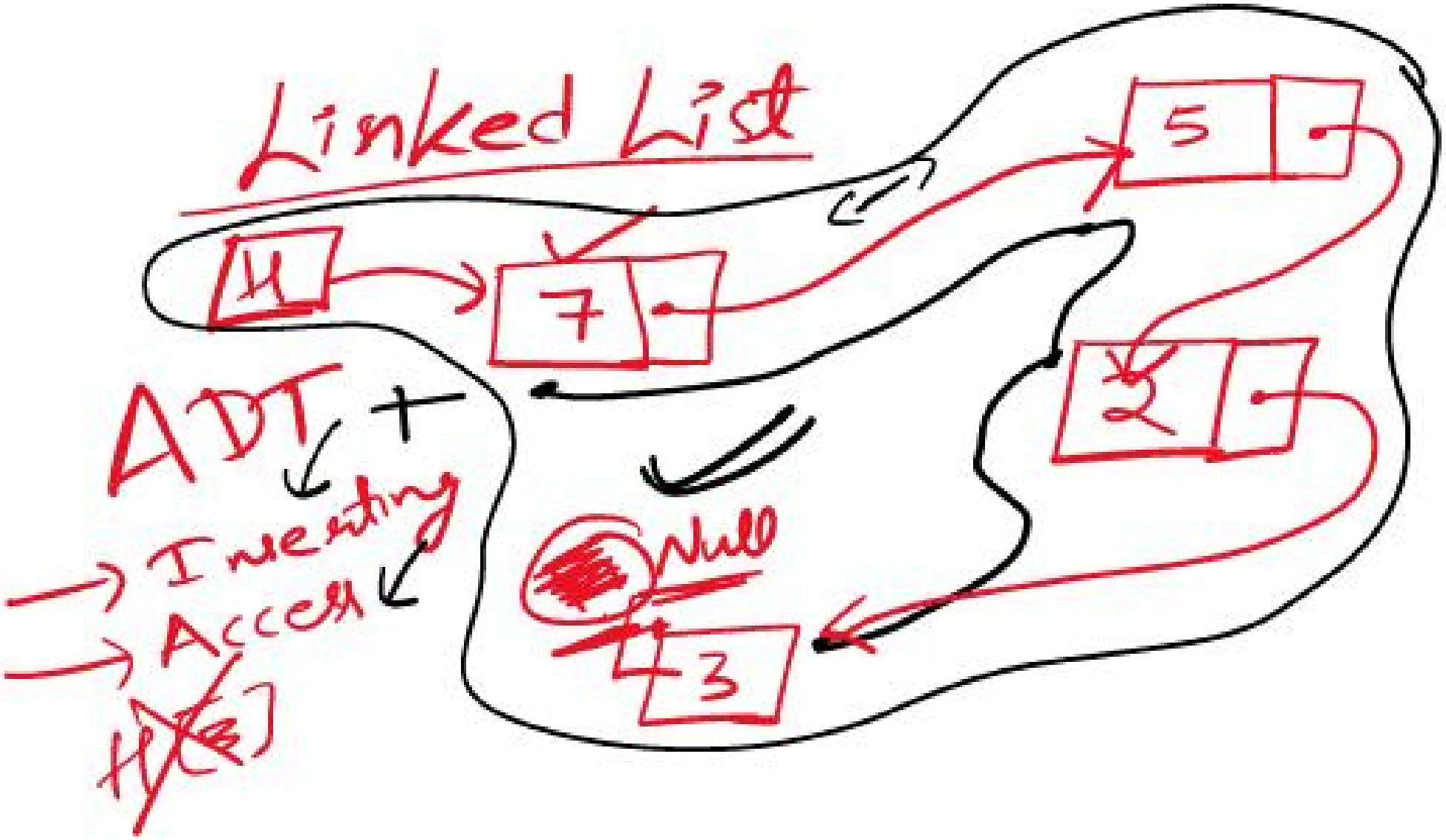

Algorithms: Steps to Solve a Prob.
Max
-∞
5
A = 1 5 3 2 7 6
→ Intuition : Idea!
Max = -∞
for (i = 0 to len)
{ if Max < A[i]
: Max = A[i]
}
1 5 3 2 7
1
5
3
Algo
if i%2 == 0 :
{ statements
}
while(i<10)
{
}


ADT
Basic operations Defined
A =
7 5 2 3 9
A[i]
Operations/Methods
functionality
=) Insert
=) Delete
=) Access
0 1 2 3
7 5 2 3
ADT
DS
We don't discuss internal working or implementation details
Model
What
OOP
Class lists


Linked List
4
7
5
2
3
Null
ADT
Inserting
Access

Pseudocode?

False Code

Syntax ✗

→ Algo can be written in:

- Plain English
- Pseudocode

for(- - -)

while

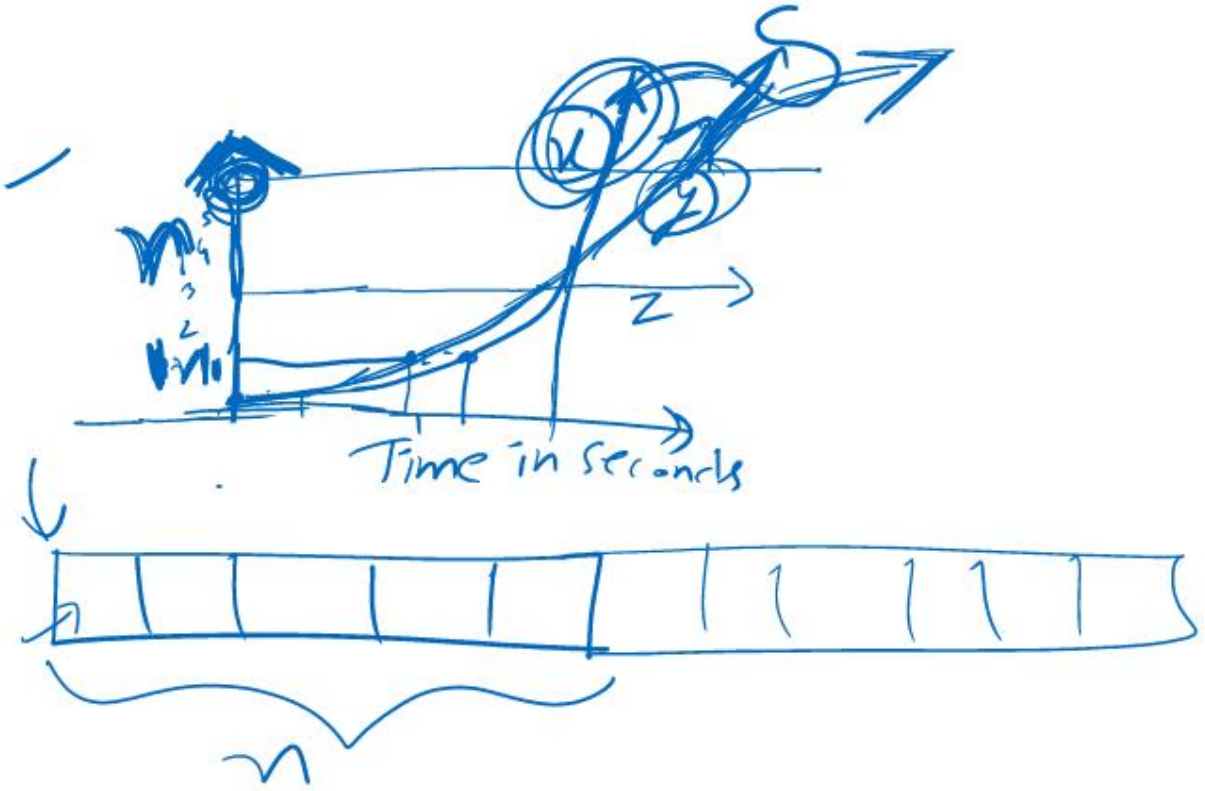
Amoeba

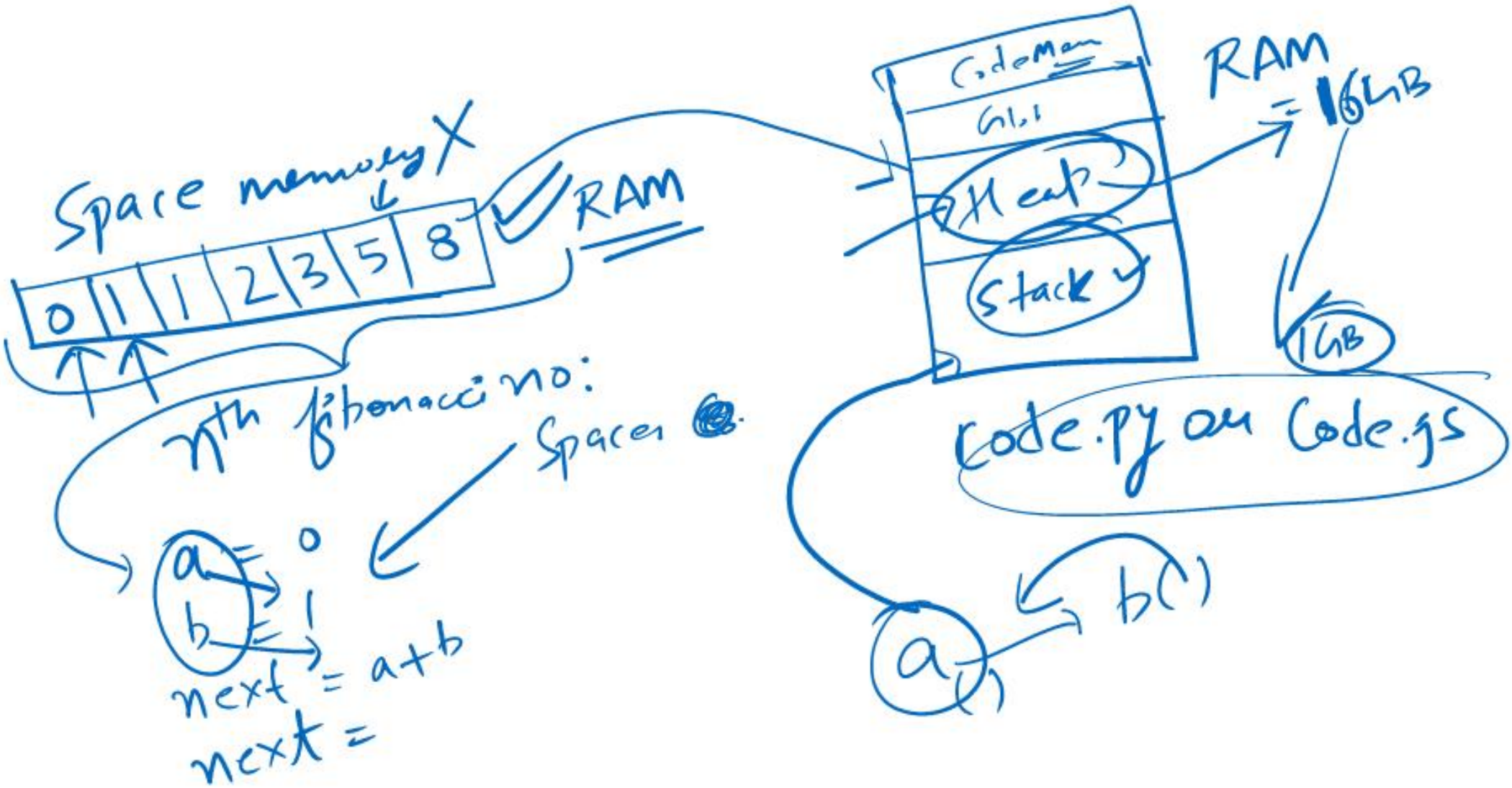
Pseudofeets?

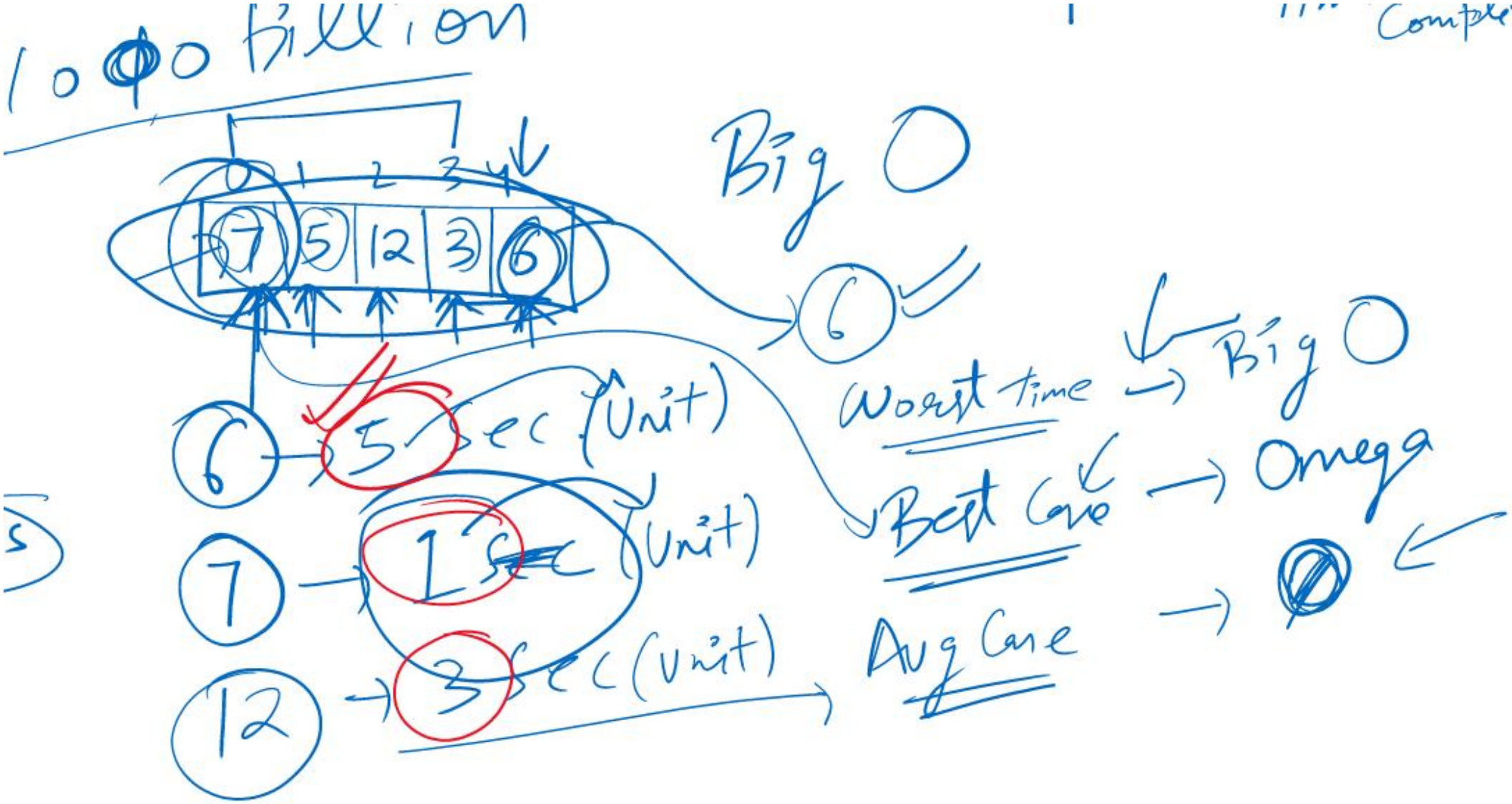
Algo => Pseudocode (Partial / Eng+

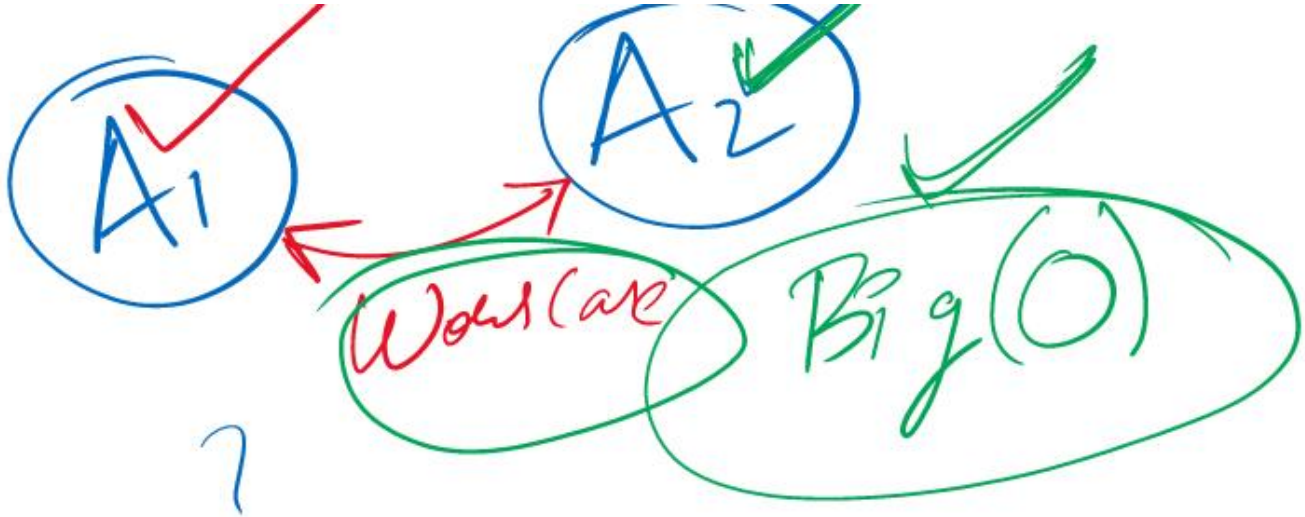
Syntax X

Partial code)



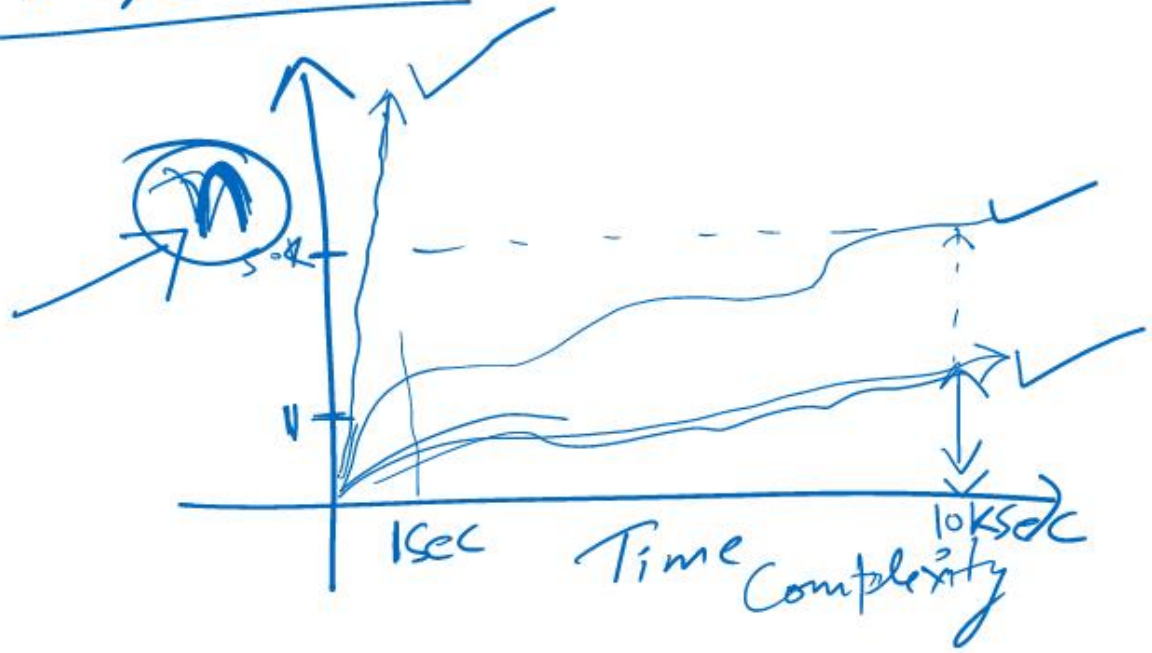

Space memory X
0 1 1 2 3 5 8
RAM
nth fibonacci no:
Space
a = 0
b = 1
next = a+b
next =
Code.py
Global
Heap
Stack
RAM = 16GB
1GB
code.py or Code.js
a()
b()

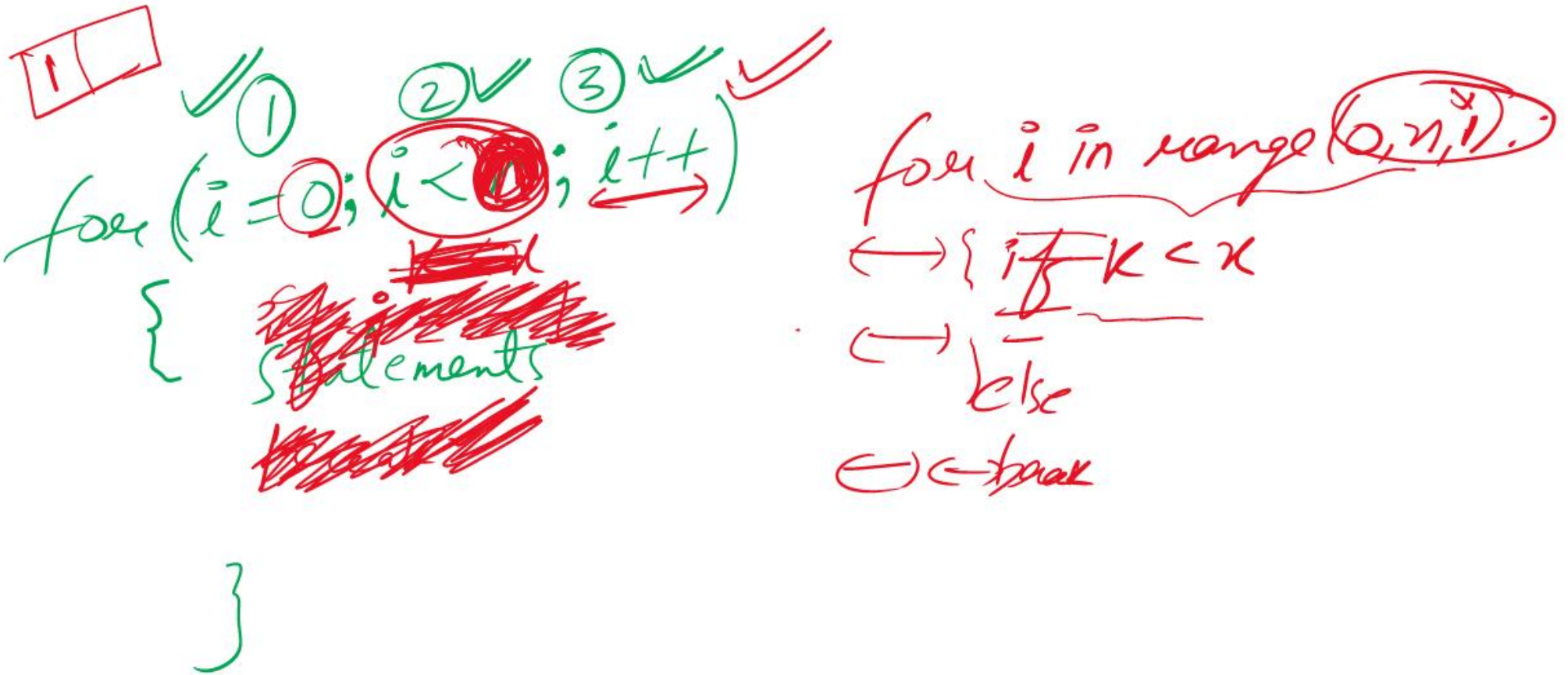

1000 billion
Complexity
Big O
1 2 3 4
7 5 12 3 6
6
6
5 sec (Unit)
7
1 sec (Unit)
12
3 sec (Unit)
Worst time → Big O
Best Case → Omega
Avg Case →

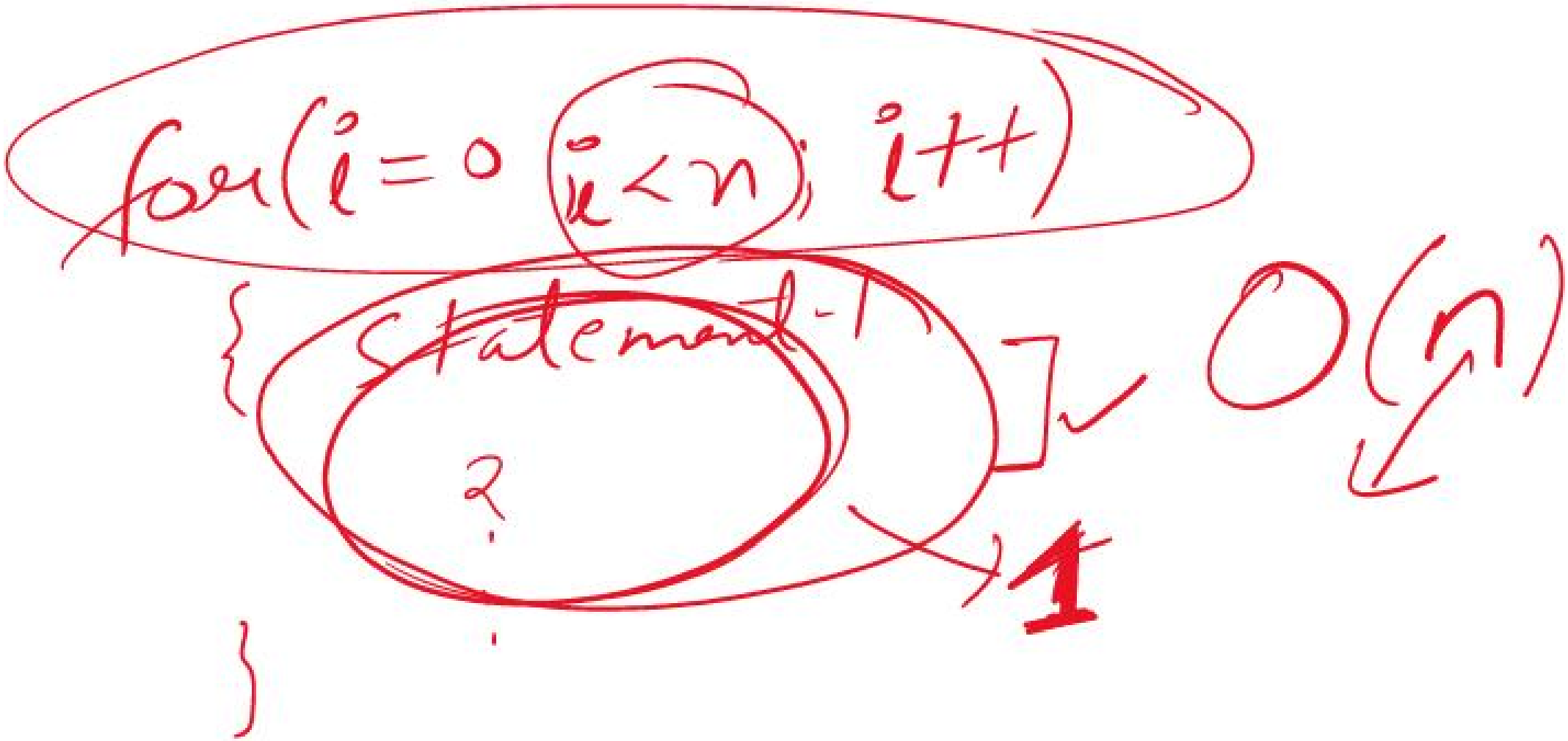


?

Worst Case Scenarios Imp?



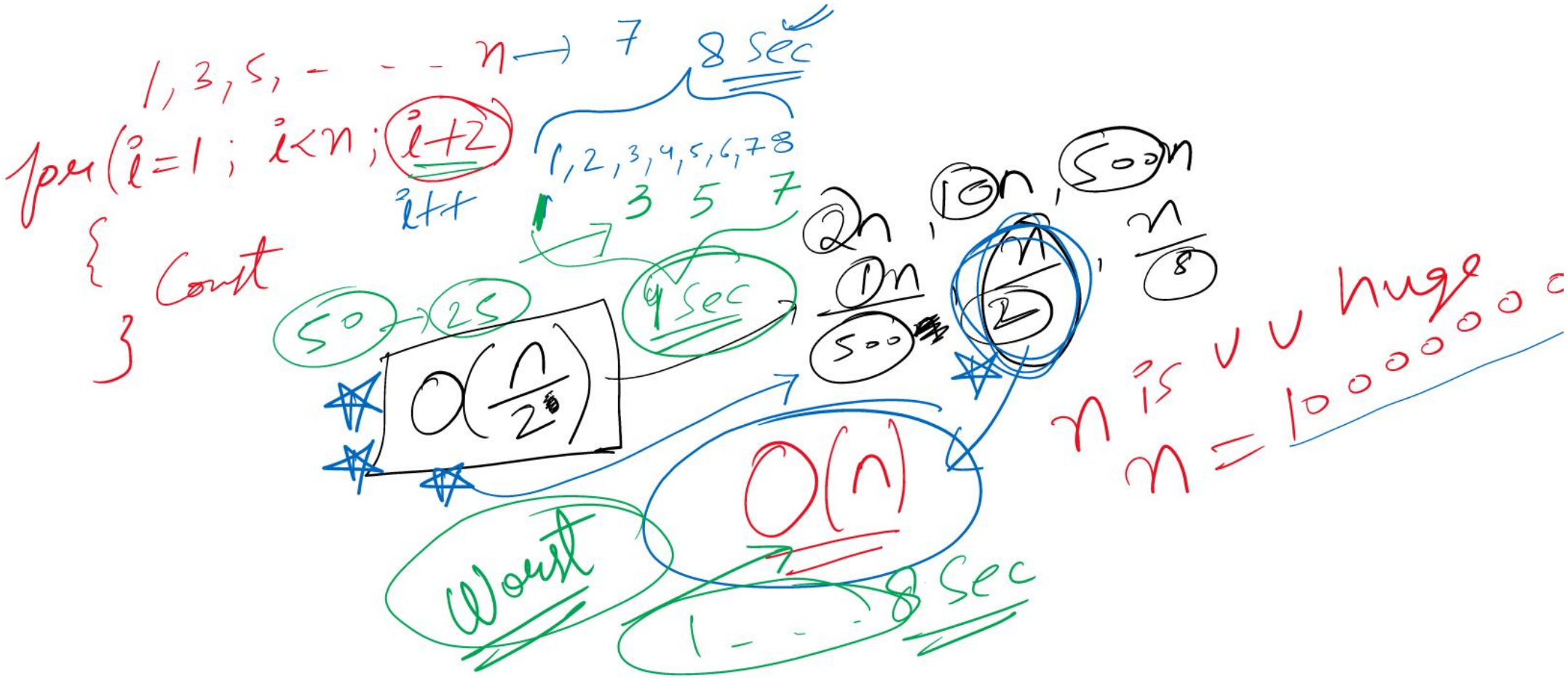

1
for (i = 0; i < n; i++)
{
Statements
}
for i in range (0, n, 1):
if k < x
else
break


for(i=0 i<n; i++)
{
statement-1
?
}
1
O(n)

```
for (i=n; i>0; i--)
{
    cout
}
```

$O(n)$

$n \times 1 = n$ sec


1, 3, 5, - - - n → 7
8 sec
for(i=1; i<n; i+2)
{ Count
}
i++
1, 2, 3, 4, 5, 6, 7, 8
1 3 5 7
50 → 25
4 sec
O(n/2)
2n, 10n, 500n
n/8
n is v v huge
n = 10000000
Worst
O(n)
1 - - - 8 sec