COMPUTER SCIENCE & IT

OPERATING SYSTEMS

Process Synchronization/ Coordination

Lecture No-02

Dr. KHALEEL KHAN

Topics of Discussion

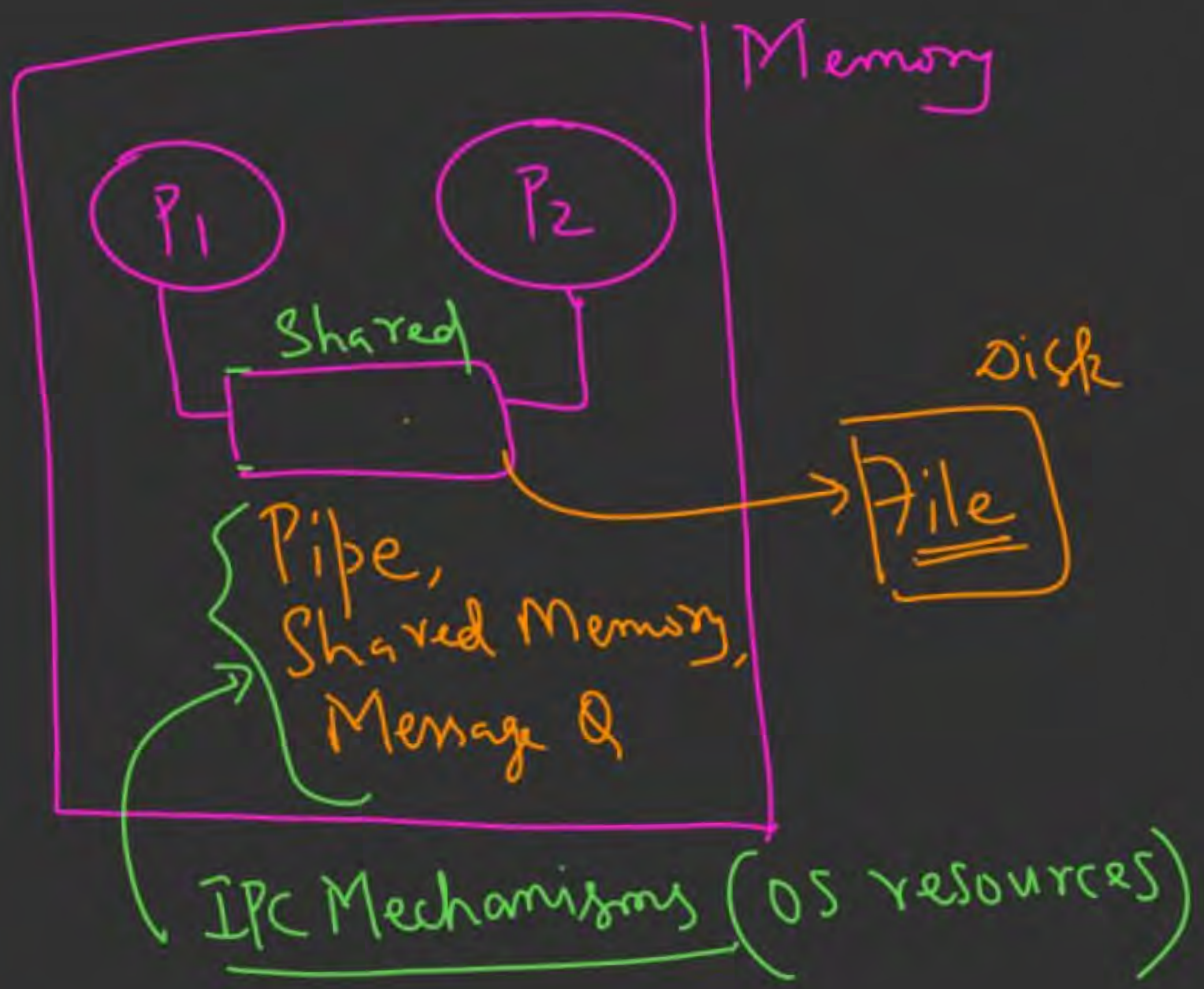
1. IPC ✓
2. Need for Synchronization ✓
3. Types of Synchronization ✓
4. Necessary Conditions for Synchronization Problems ✓
5. Critical Section Problem – Terminology ✓
6. Requirements of CS Problem ✓

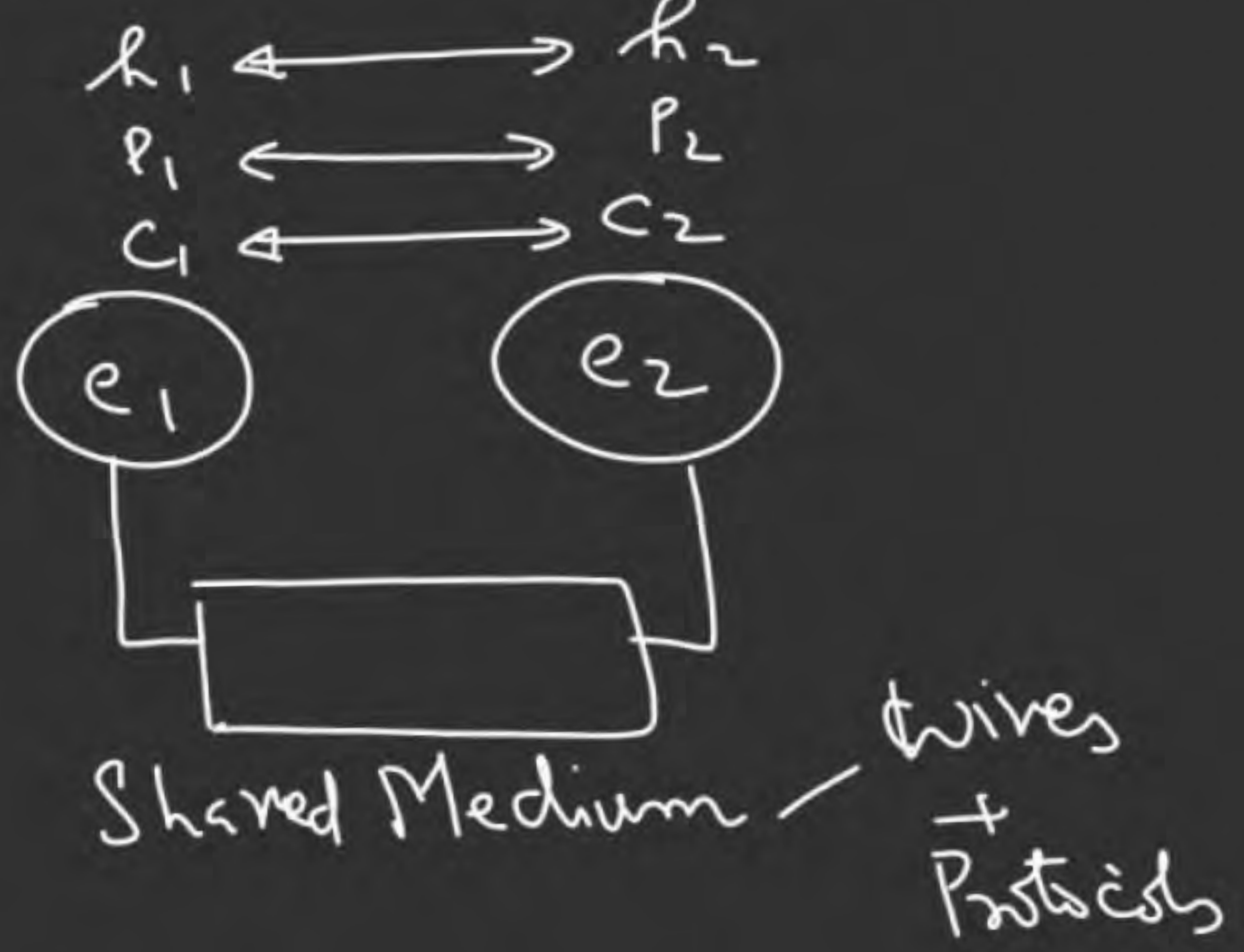
foundation

7. Synchronization Mechanisms
   - a) Lock Variable ✓
   - b) Strict Alternation ✓
   - c) Peterson Solution ✓
   - d) Hardware Synchronization : TSL & SWAP ✓
   - e) Blocking Mechanisms
     - I) Sleep () & Wakeup () ✓
     - II) Semaphores ✓
     - III) Monitors ✓
8. Classical IPC Problems

What is IPC:

↳ Inter Process Communication;





The communication must be Synchronized

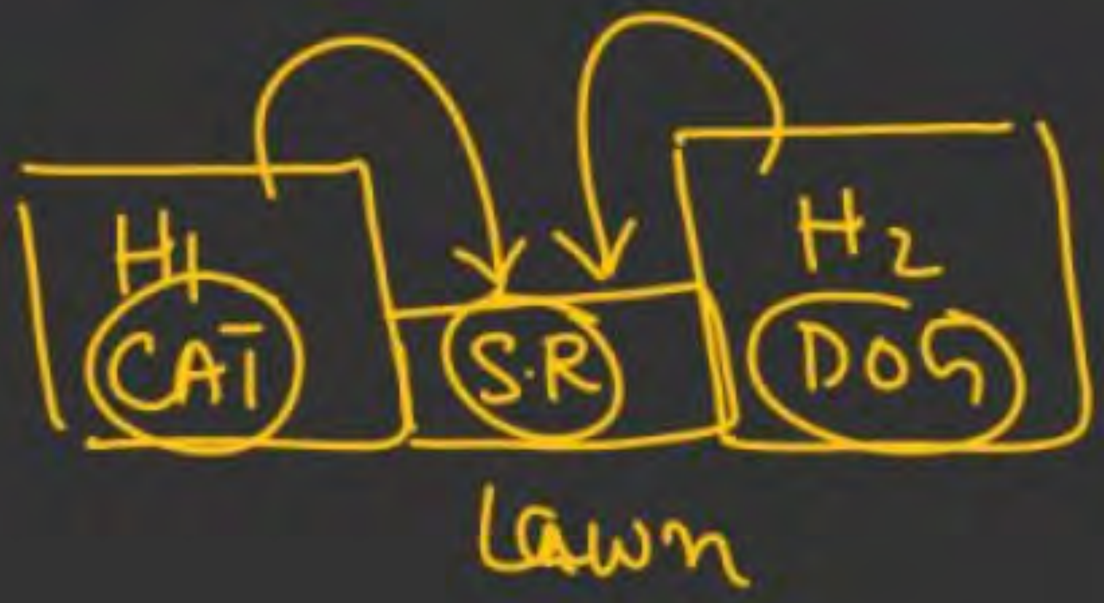
otherwise it will Lead to Problems;

{Need for Synchronization} { (i) Inconsistency (Incorrectness)

(ii) Data Loss

(iii) Deadlock (Processes gets Blocked for ever)

(i)



(ii) Paying Guest Milk

Note
Fridge

{3 Bachelors}

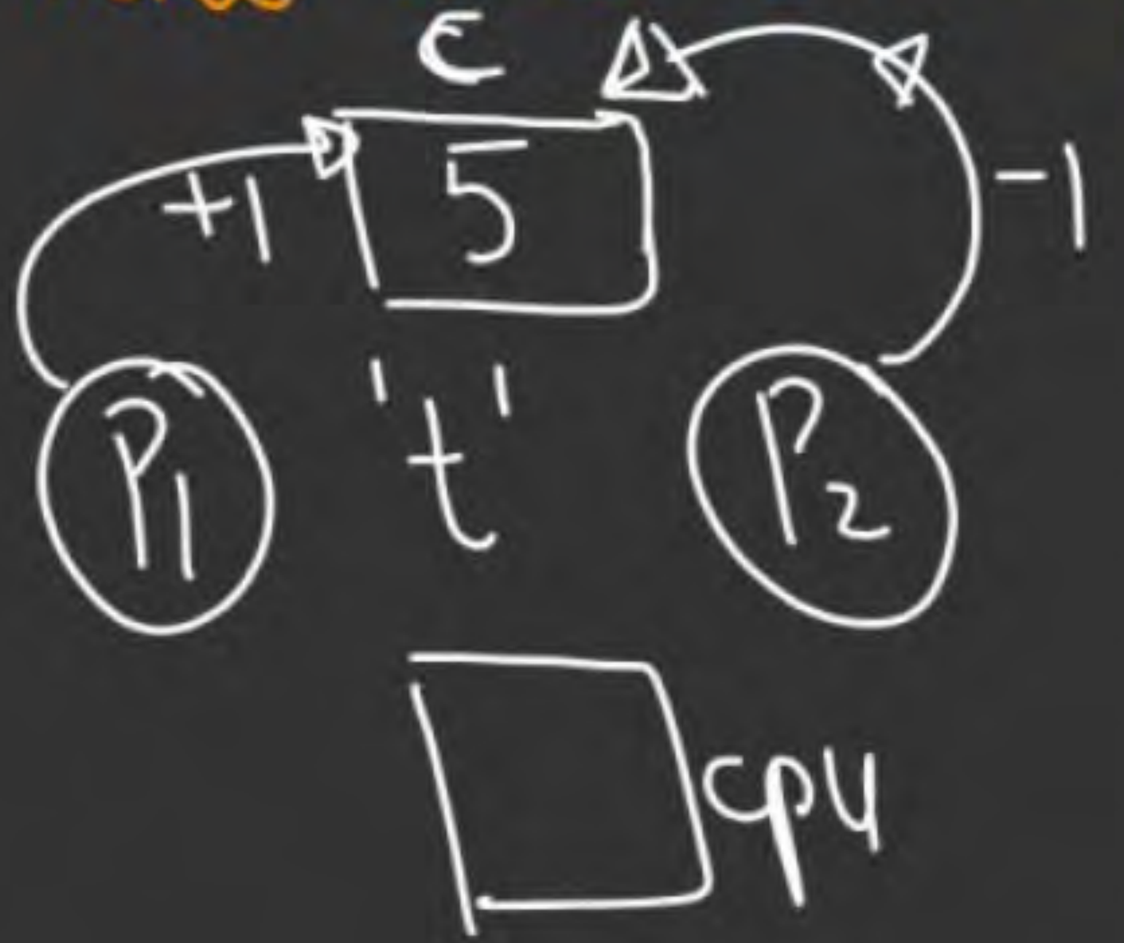
Move MILK

Types of Synchronization (in IPC Environment)

Competitive

Co operative

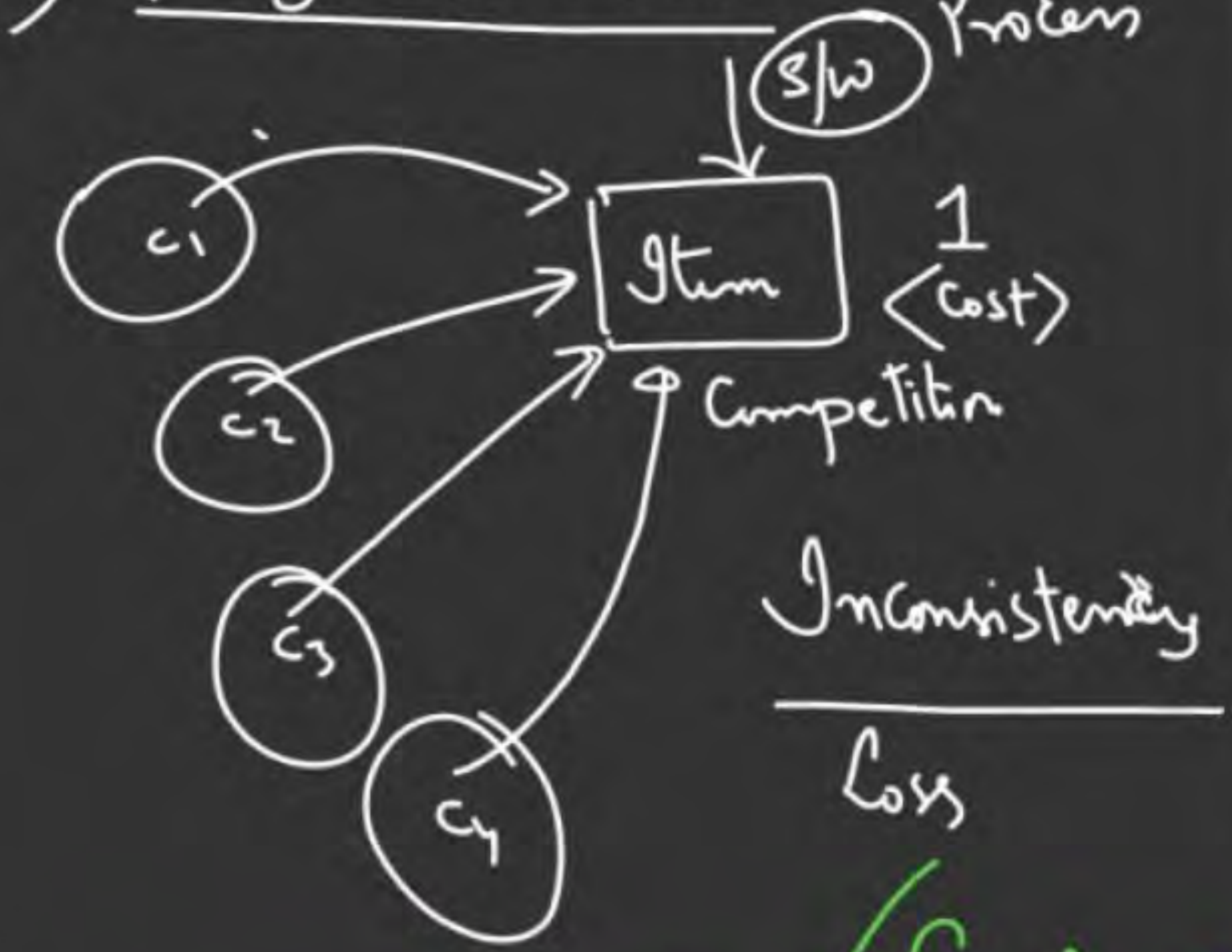
1) <u>Competition</u>: Two or more Processes are said to be in Competitive Synchronization, iff they Compete/Content for the accessability of a Shared resource;



Final/Correct-value-of $c = 5$

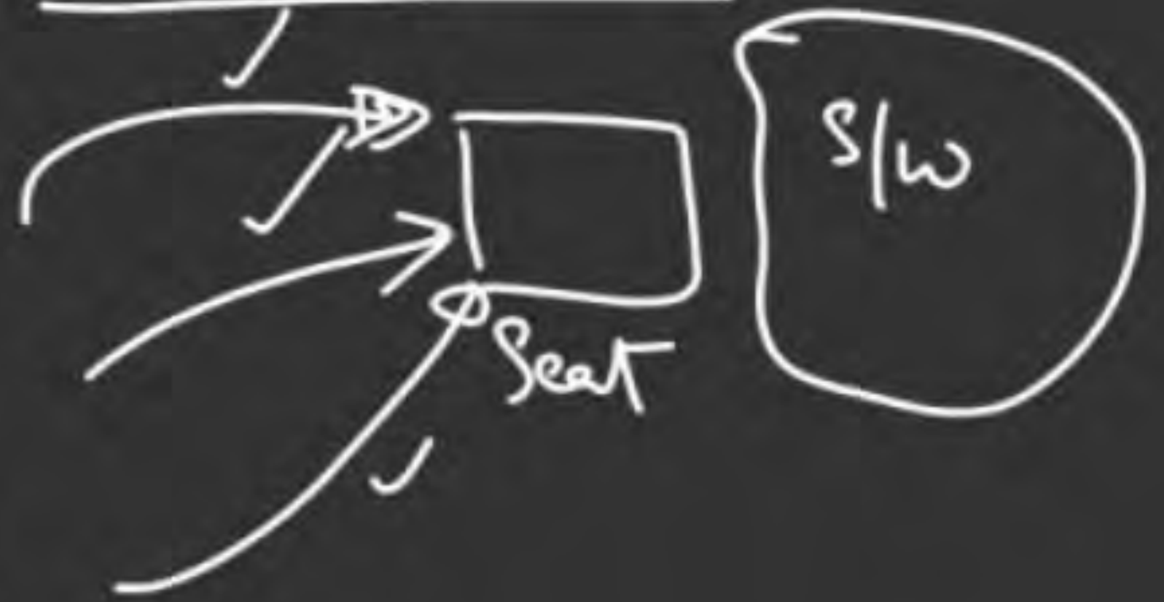
4
6
Inconsistency

1) Amazon Purchase



2) Reservation System

Airline | Train



→ Note: (Lack of Synchronization among competing (Competition) Processes may lead to the problem of Inconsistency (or) Data loss;)

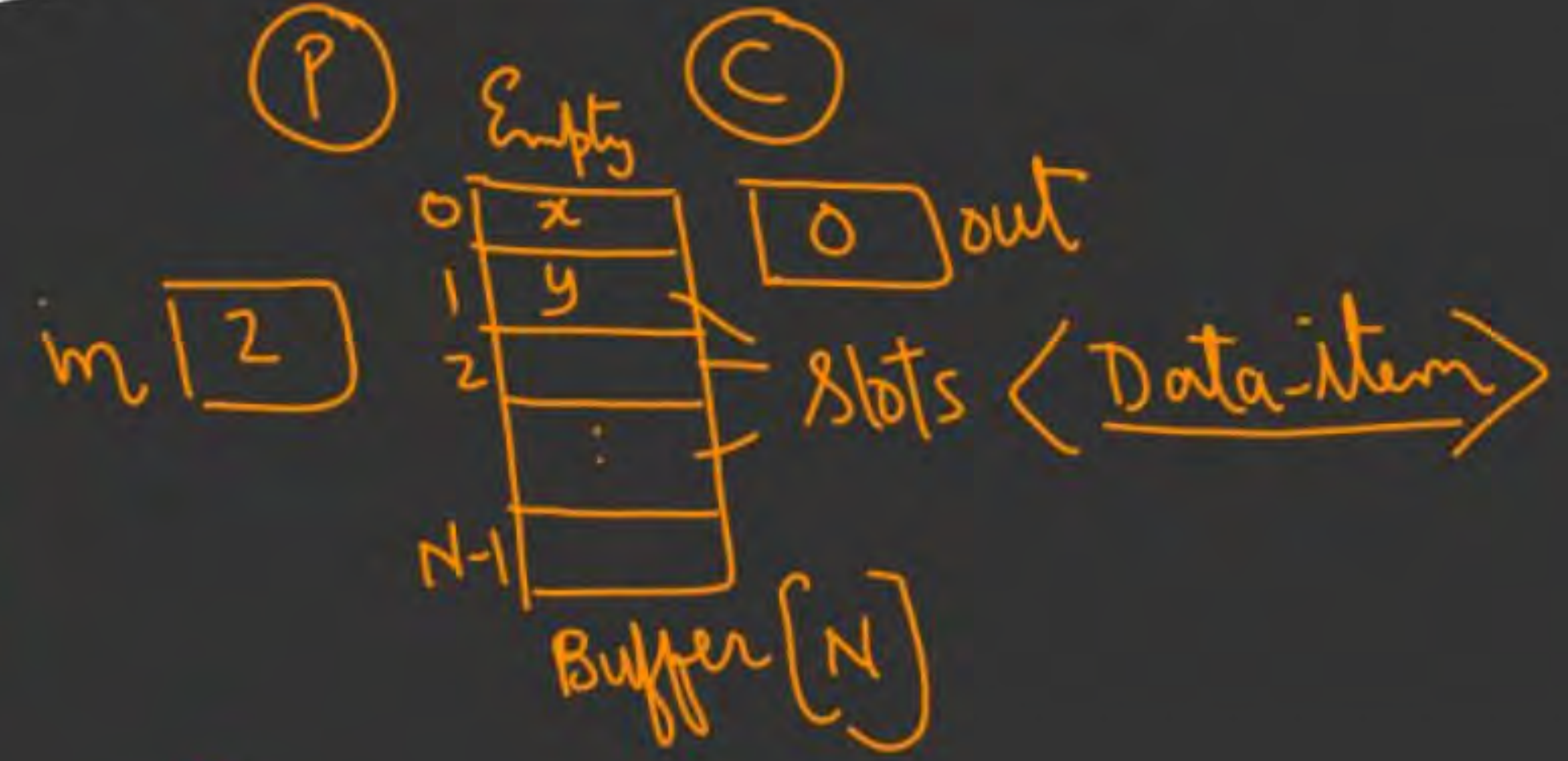
2) Cooperative Synchronization:

Two/more processes are said to be in Cooperative Synchronization, iff they get affected by each other;

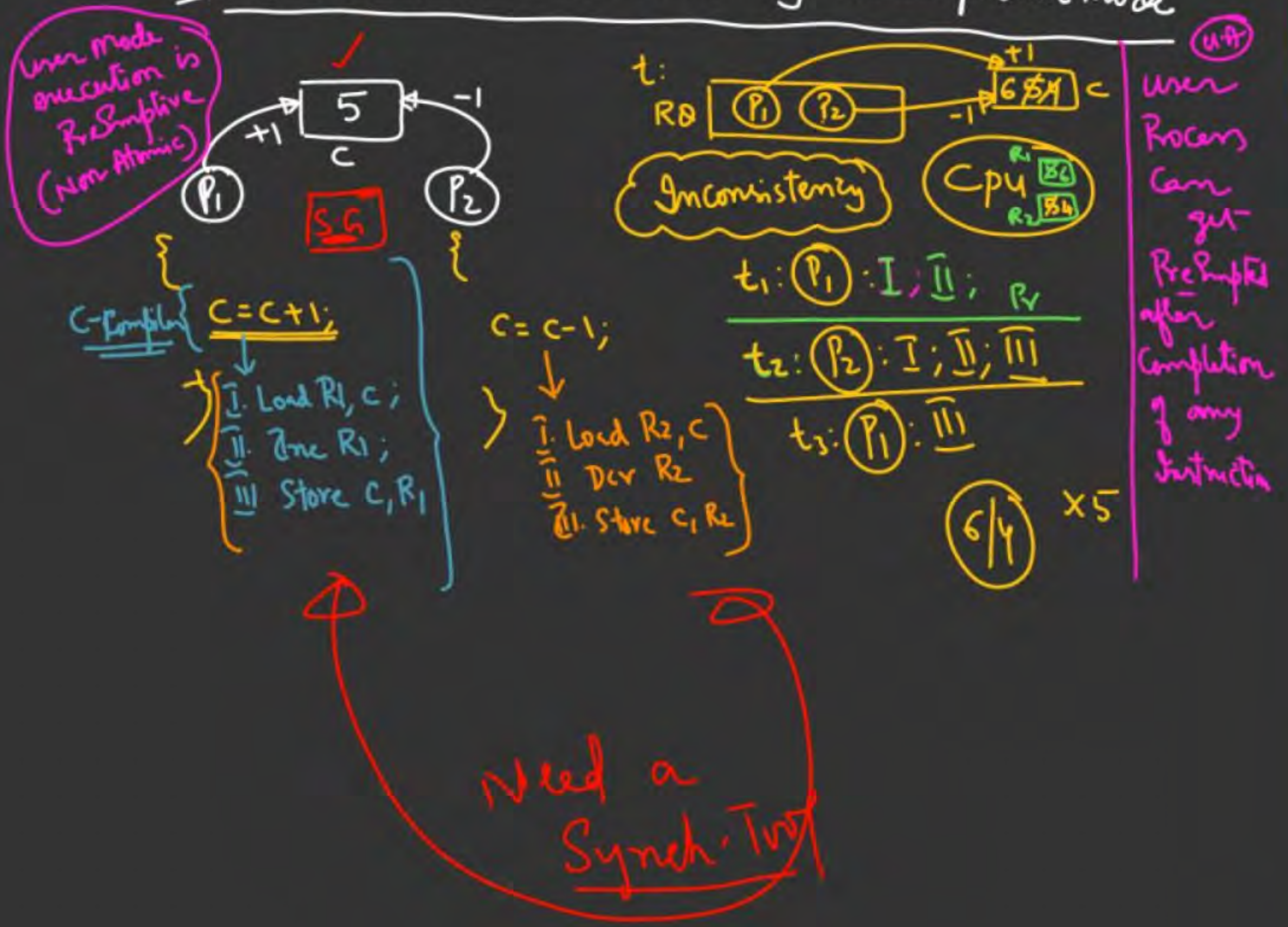
"Execution of one Process affects the Other"

{"Lack of Synch. b/w Cooperating Processes May Lead to Deadlock"}

Ex: Producer-Consumer:



Ex: to demonstrate Inconsistency in Competition Mode



user mode execution is PreEmptive (Non Atomic)

```
P1 {
C=C+1;
  I. Load R1, c;
  II. Inc R1;
  III. Store C, R1
}

P2 {
C=C-1;
  I. Load R2, C
  II Dcr R2
  III. Store C, R2
}
```

t_1: P1 : I; II;

t_2: P2 : I; II; III

t_3: P1 : III

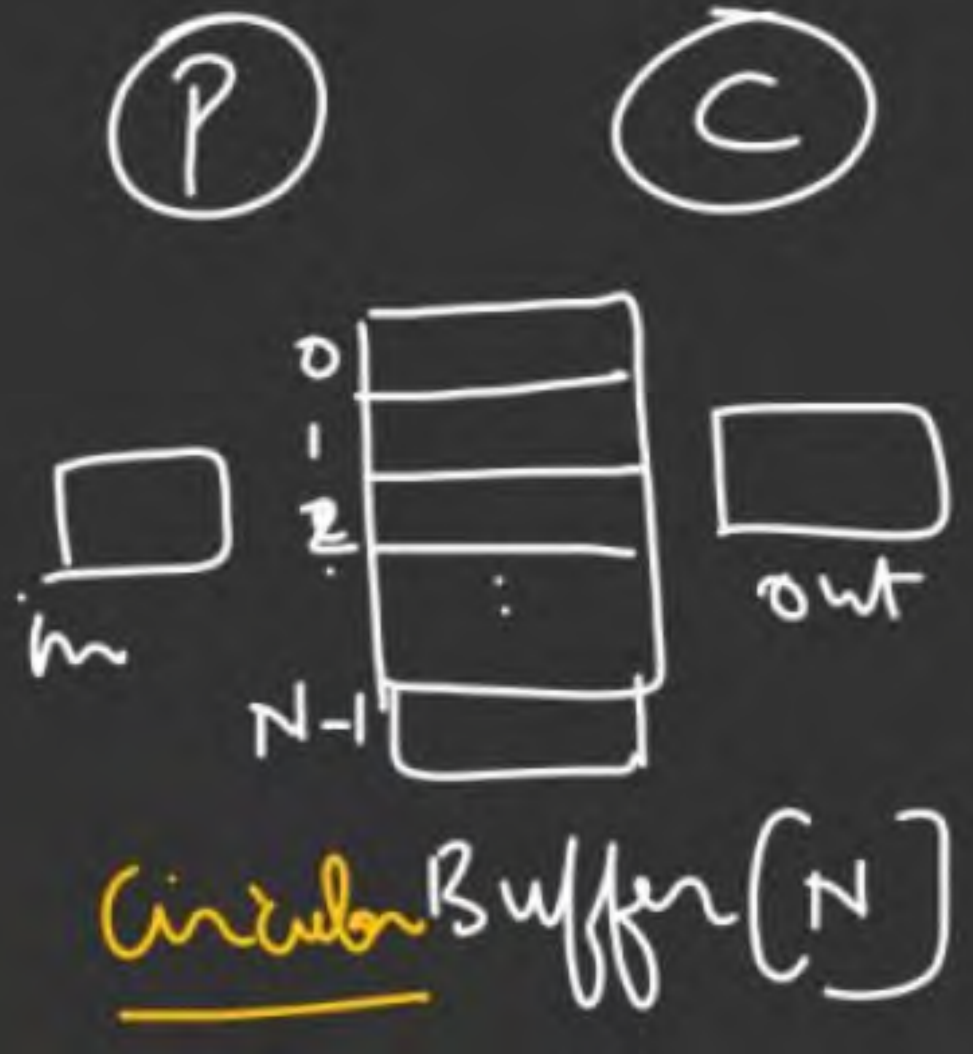
user Process can get PreEmpted after Completion of any Instruction

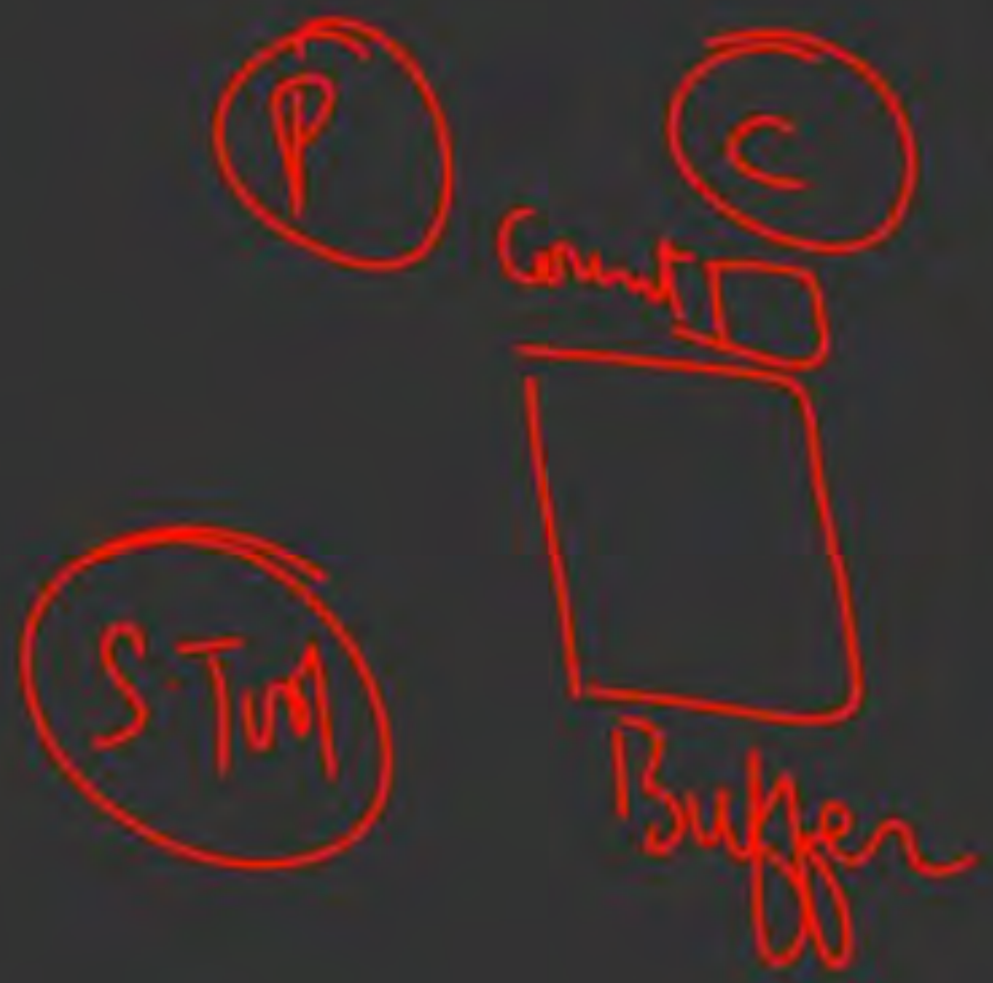
→ Inconsistency is Possible with PreEmption during the execution of Process;

→ In case of no PreEmption during the execution, we always get correct result

Implementation of Producer-Consumer

Q: Is this Impl of P & C Correct/Incorrect ?





```
#define N 100
int Buffer[N];
int Count = 0;

void Producer(void)
{
    int itemp, in=0;
    while(1)
    {NCS
        1. itemp = Produce_item();
    CS  2. while(Count==N);
    CS  3. Buffer[in] = itemp;
    NCS 4. in = (in+1)%N;
        5. Count = Count+1;
    }CS
}
```

Inconsistency

Comp

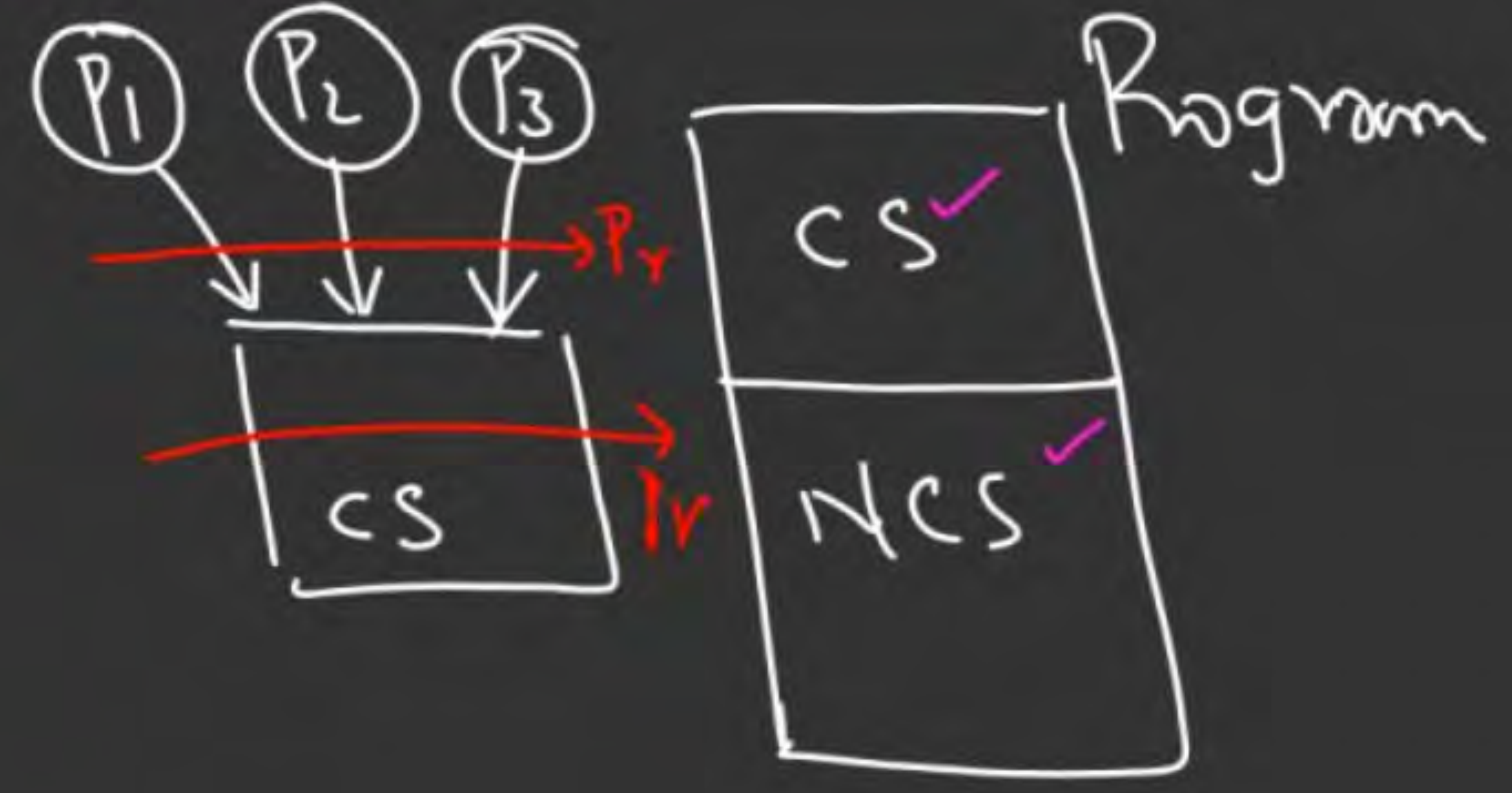
```
void Consumer(void)
{
    int itemc, out=0;
    while(1)
    {
        1. while(Count==0);
        2. itemc = Buffer[out];
        3. out = (out+1)%N;
        4. Count = Count-1;
        5. Process_item(itemc);
    }
}
```

Necessary Conditions for Synch. Problems to occur in IPC Environment

1. ✓ **Critical Section (CS):** is that part of the Program where Shared resources are accessed ⟨Terminology⟩

   → Non-CS / other Section : is that part of the program which does not access Shared resource;

2. **Race Condition:** Situation wherein multiple Processes tries to access Shared resource (CS) & the final outcome depends on the order in which they finish;

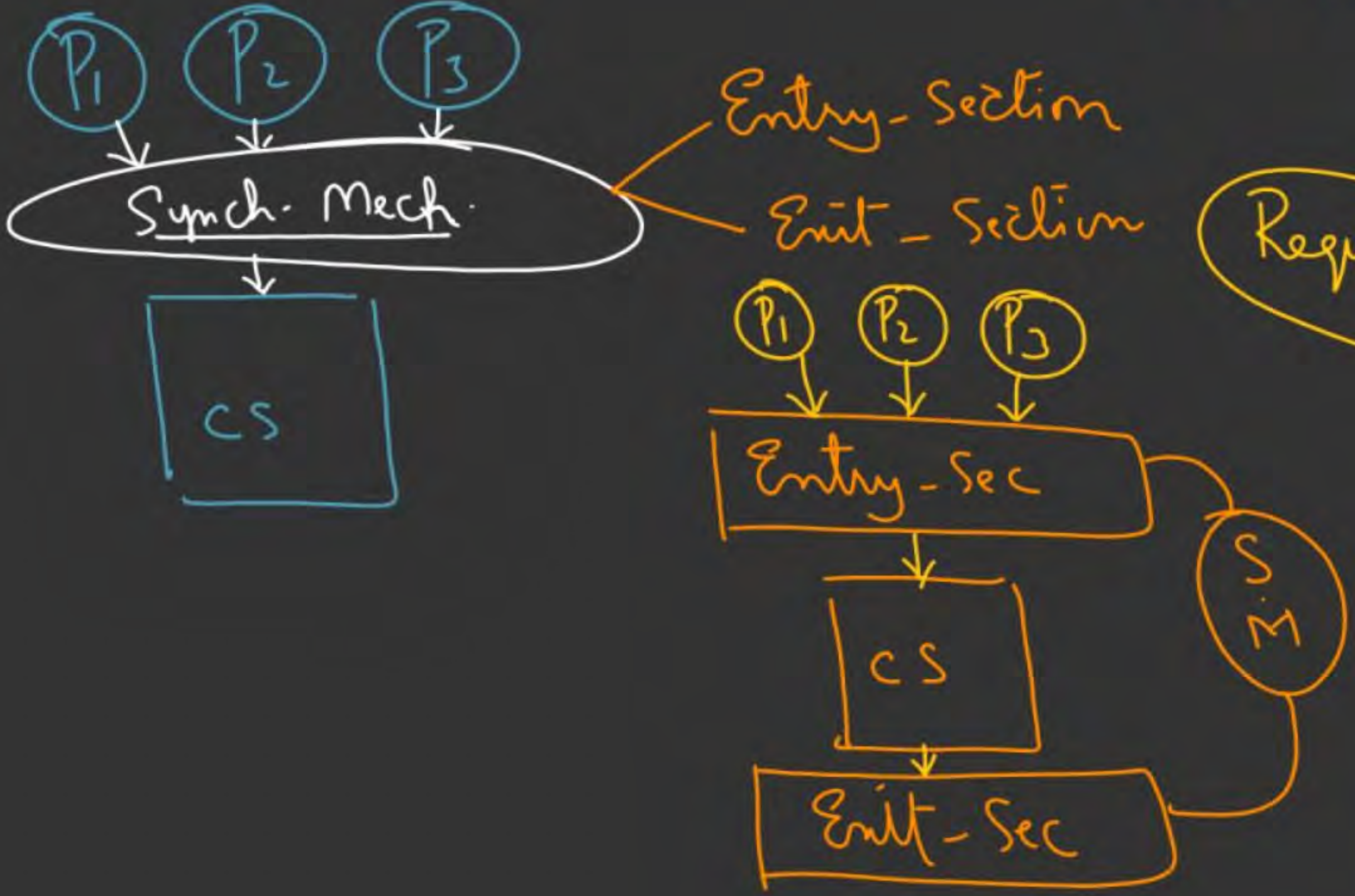


3. Pre Emption

Name of this problem : Critical Section Problem

CS – Problem

11:45 am



Requirements

Soln : Synch. Mech.

Algorithm Tool

Architecture

Model
of
Synch Tool | CS Problem

THANK YOU GW SOLDIERS !