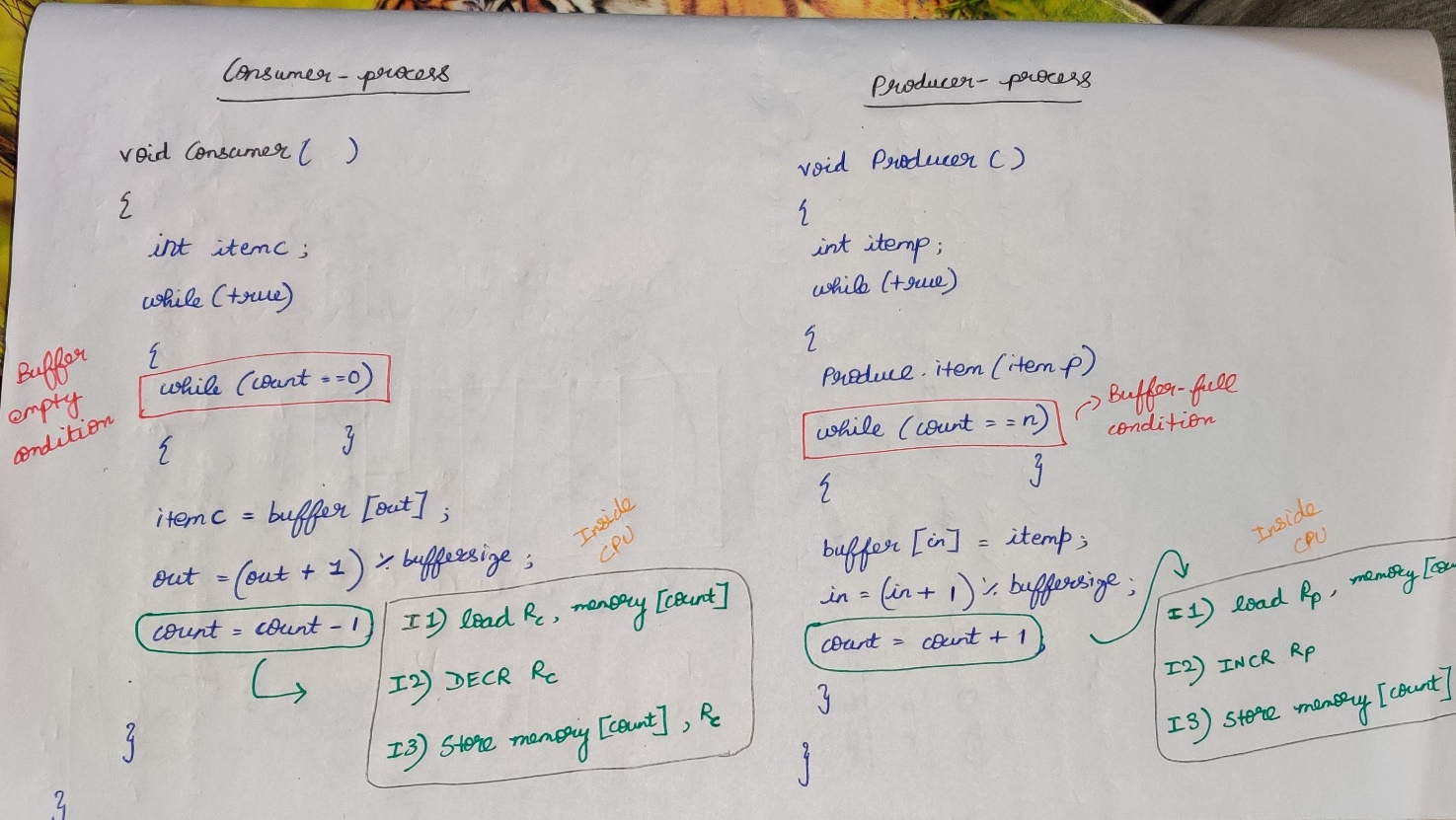
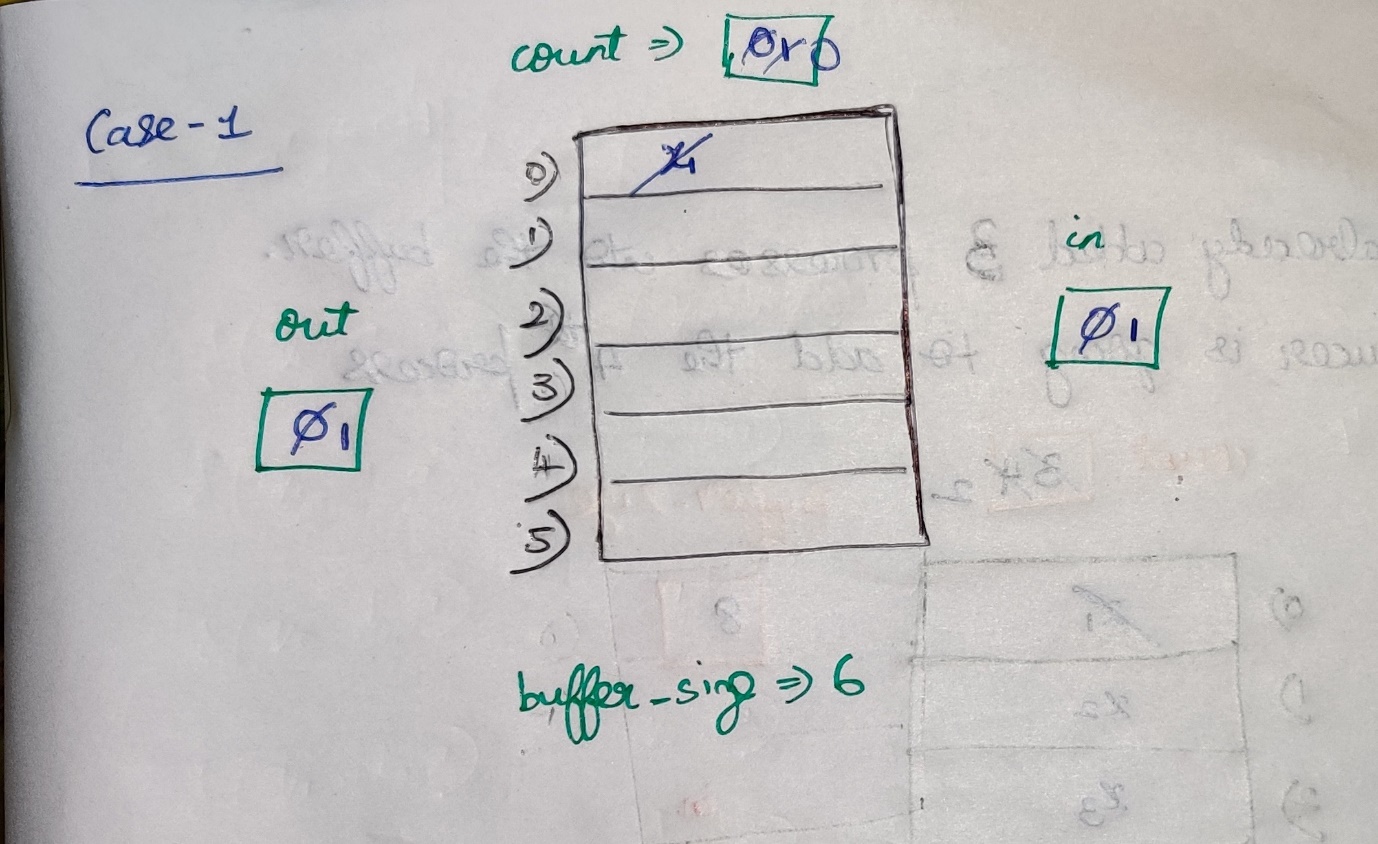


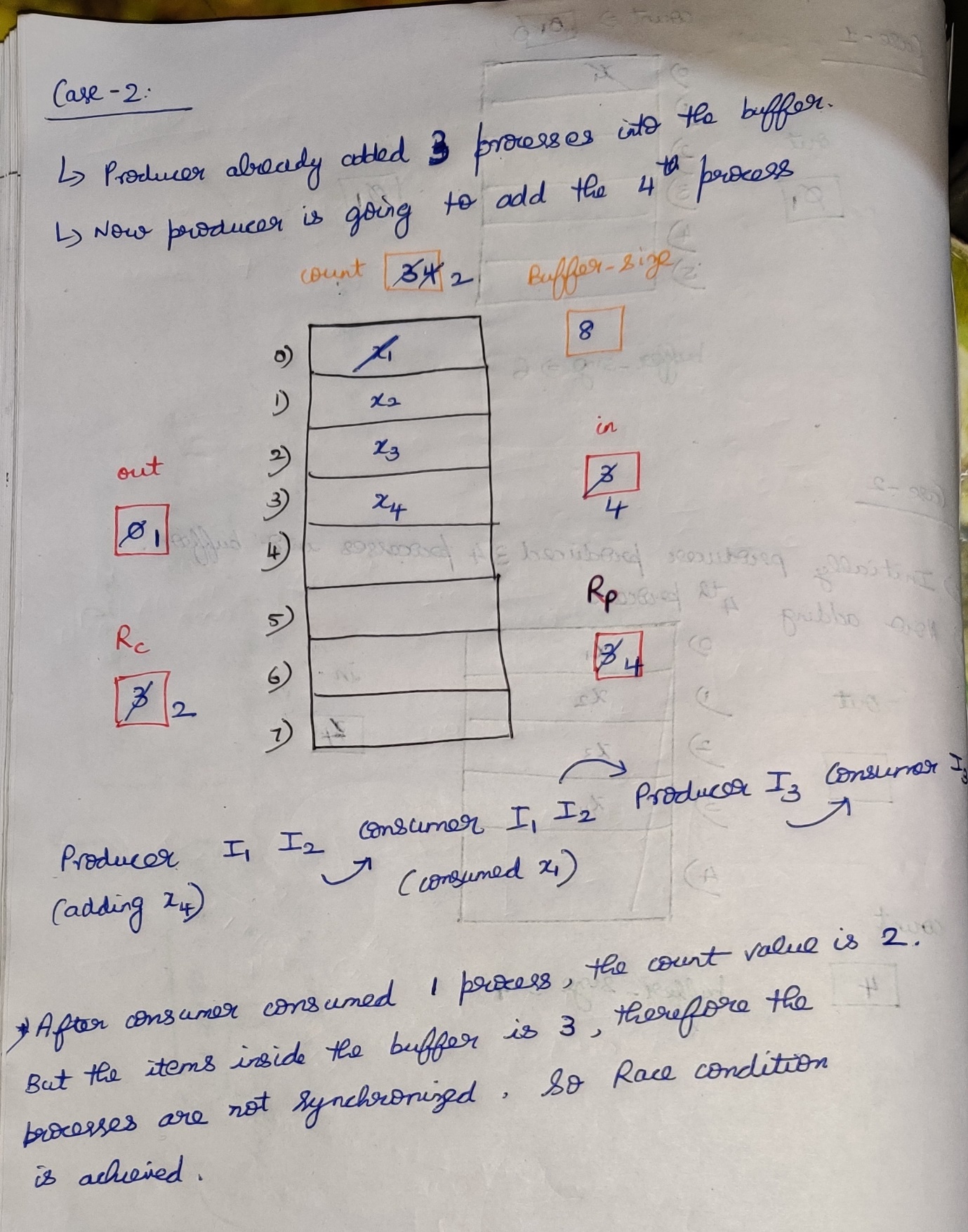
**Producer-Consumer Problem**

count 🡪 how many items are there in buffer  
in 🡪 will point out the next empty slot(producer)

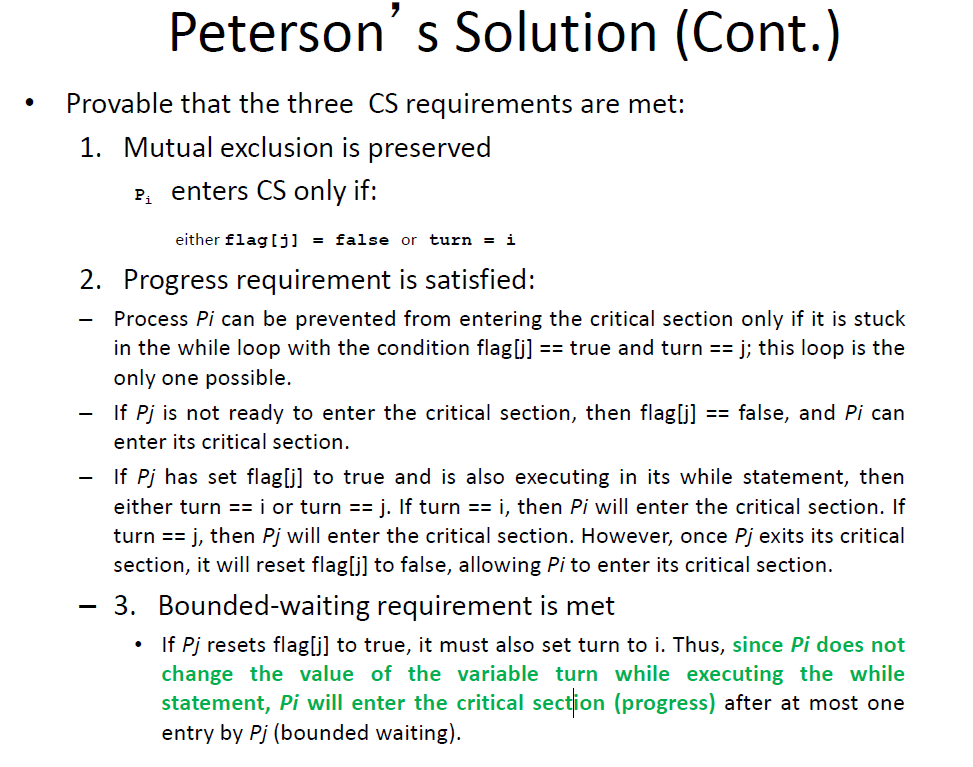
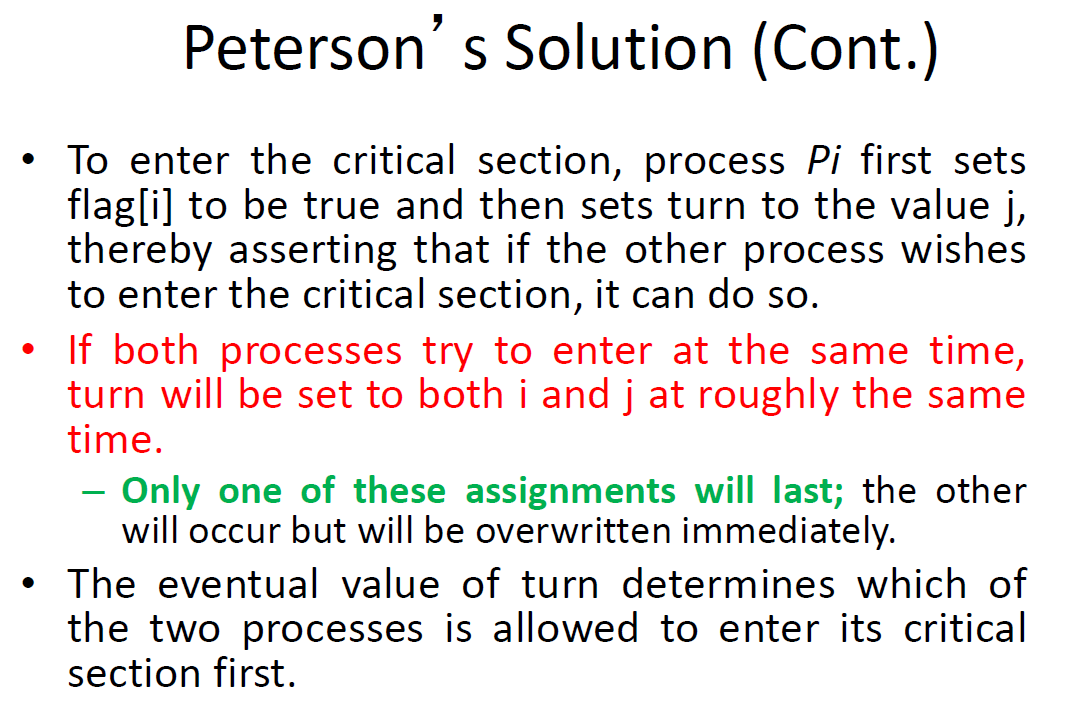


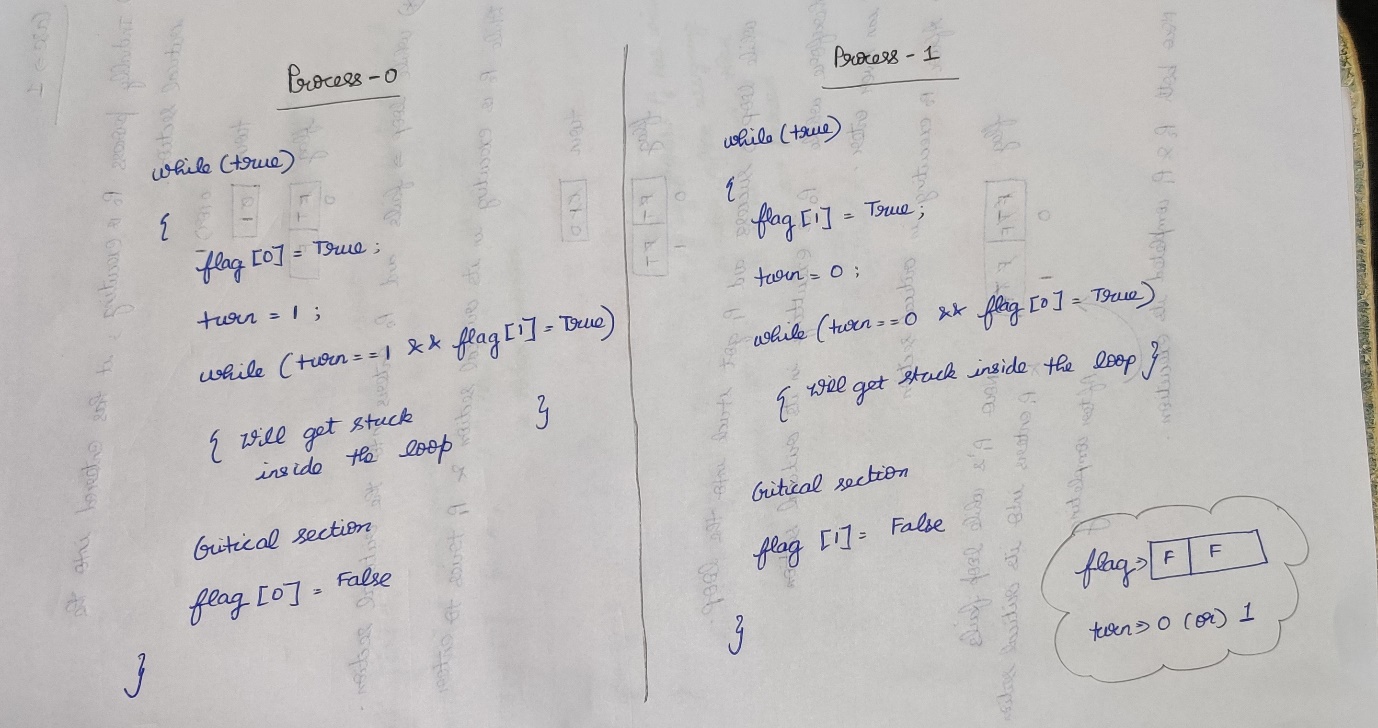
Case-1: Adding 1 item into the buffer and consuming it.

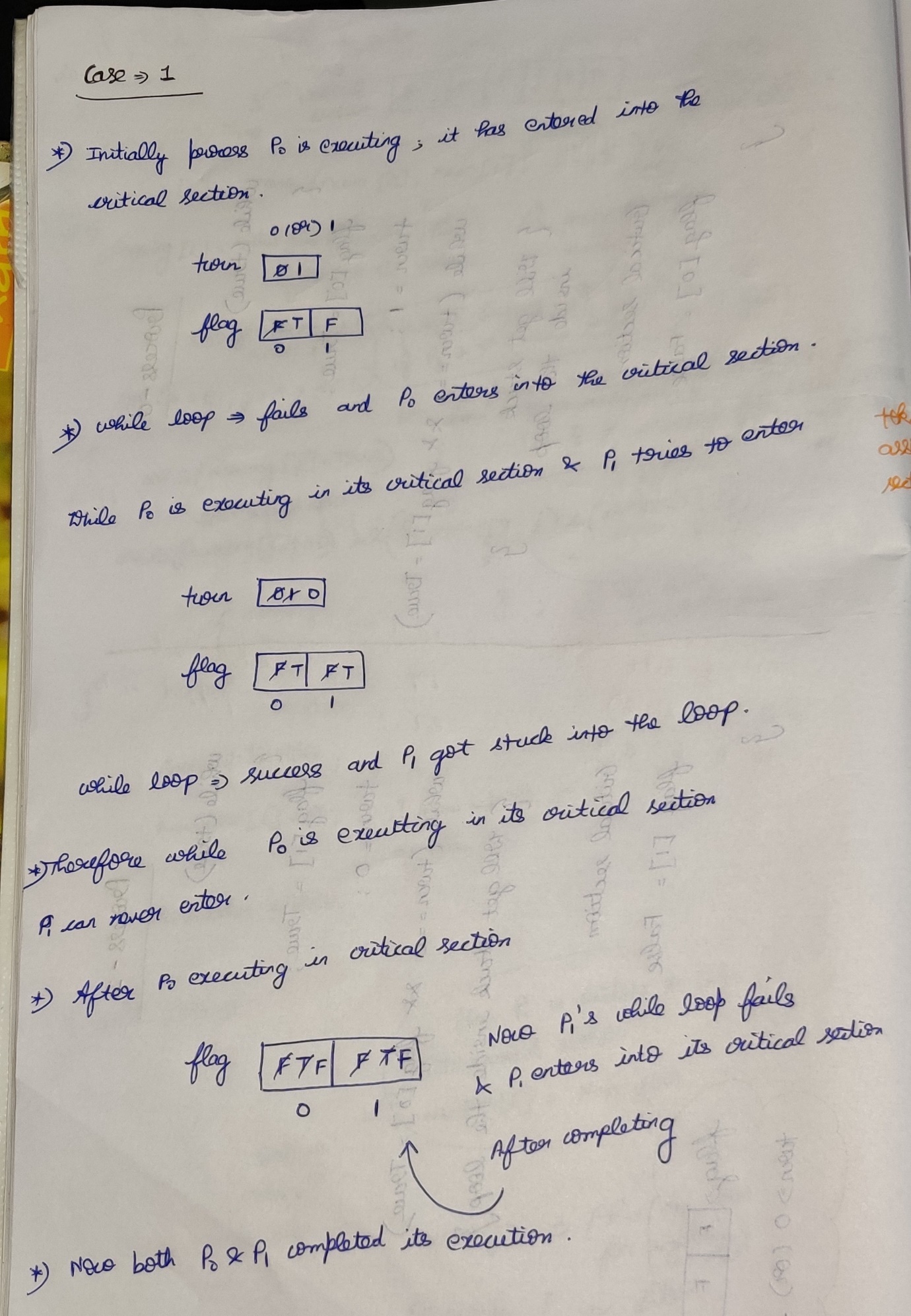


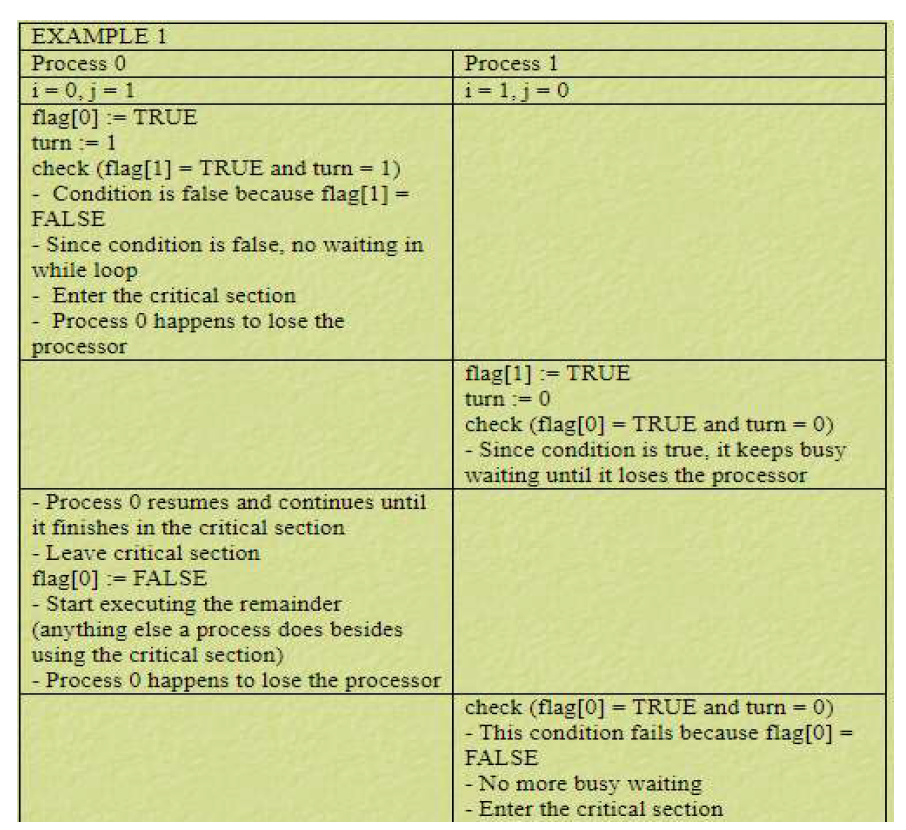


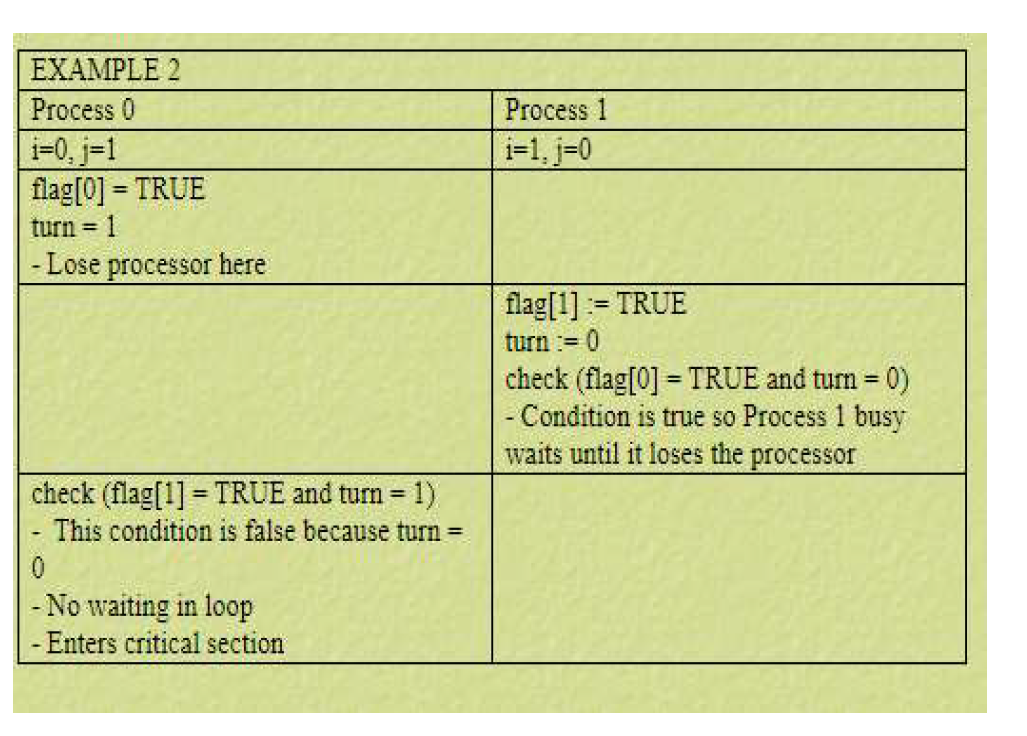
**Peterson’s solution to Producer-Consumer (two -process solution)**

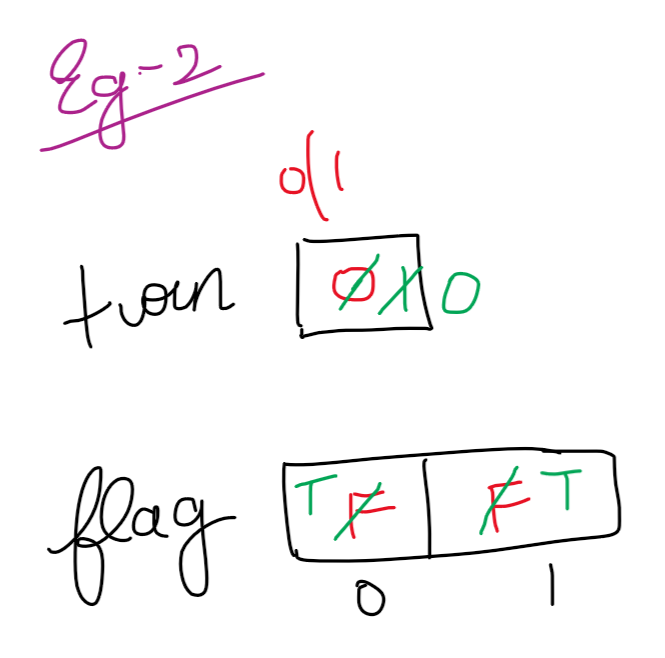


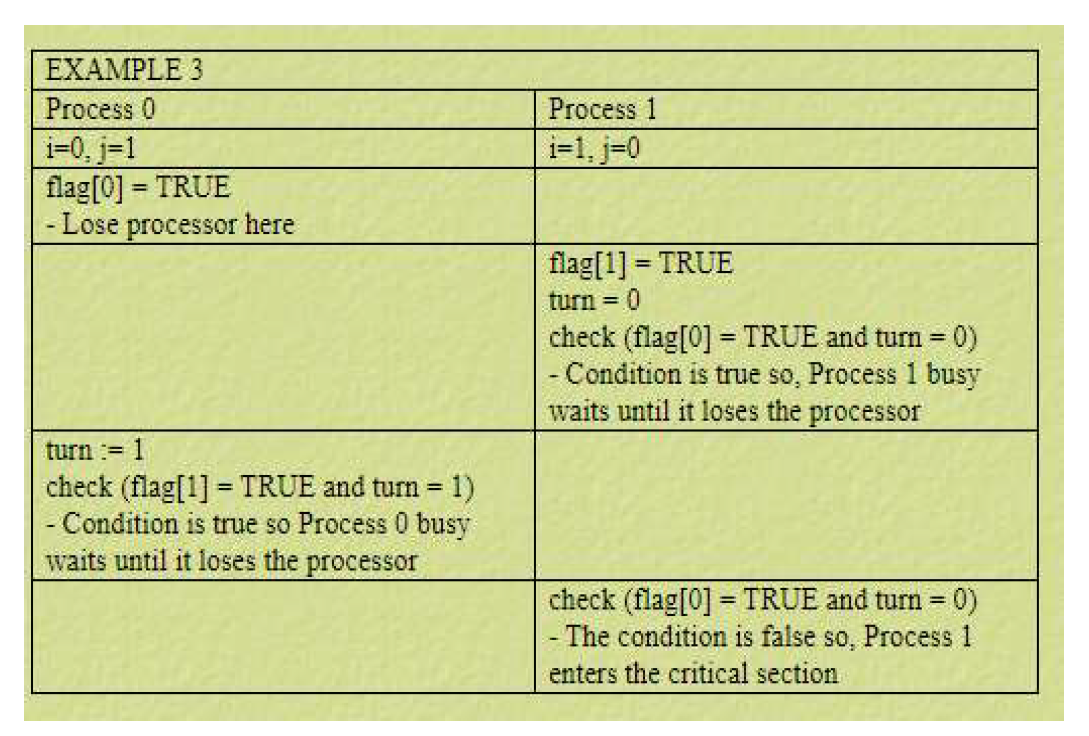


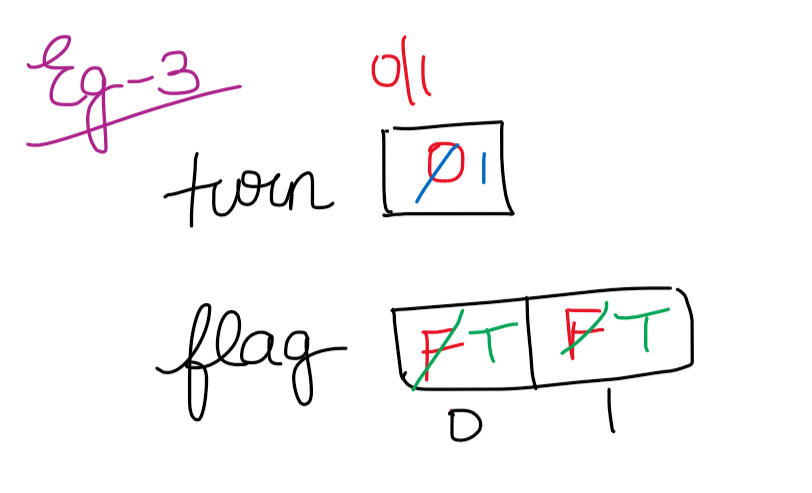




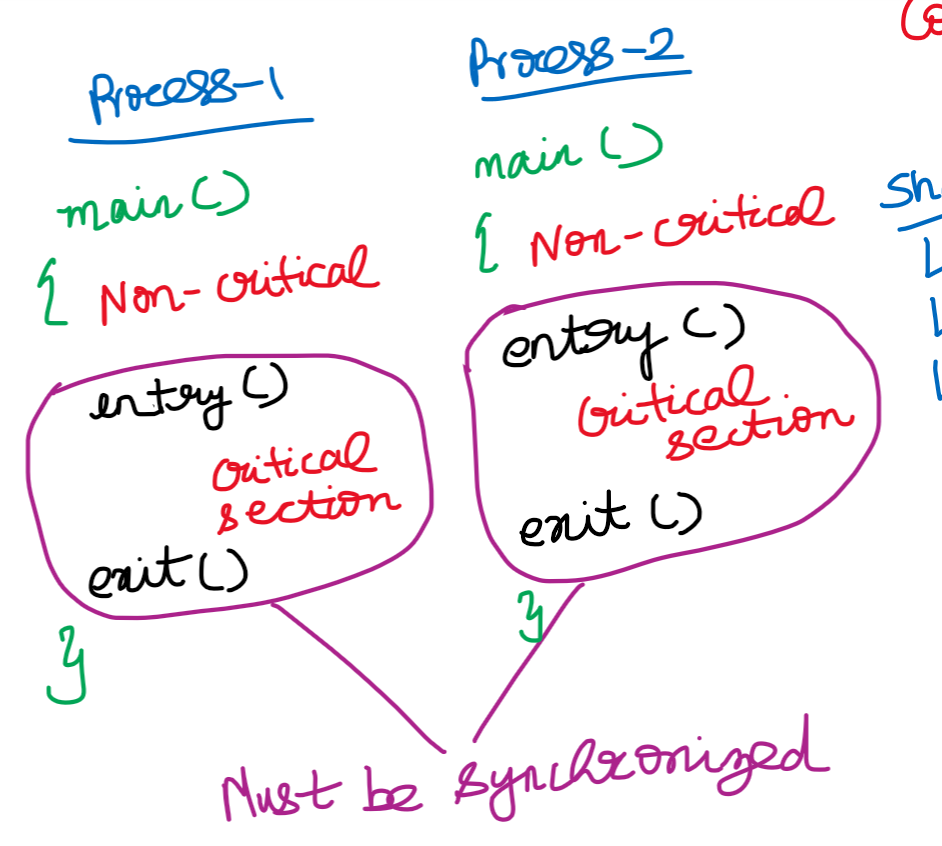


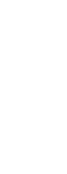


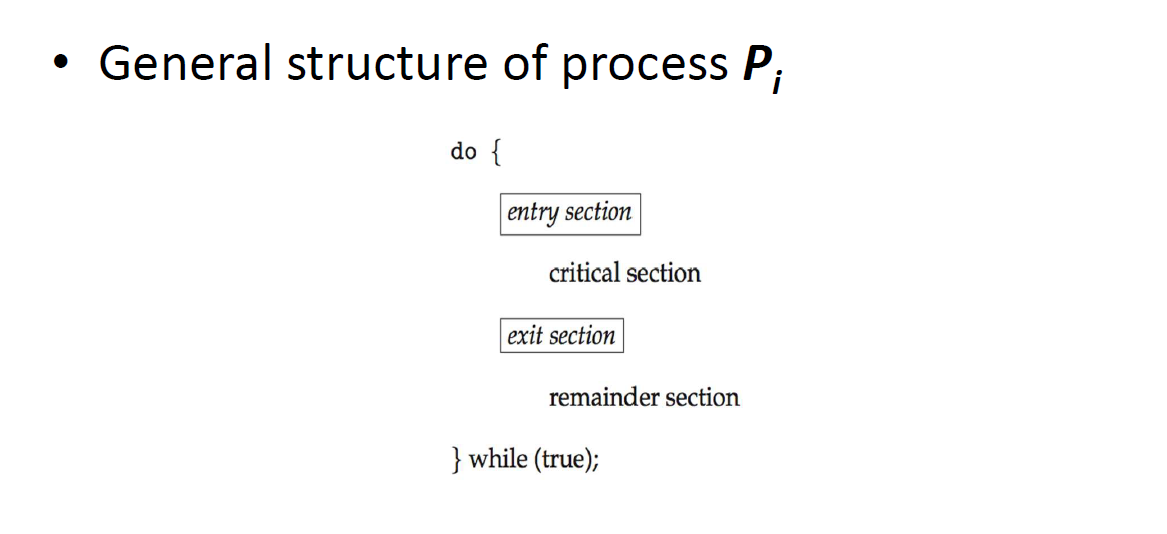


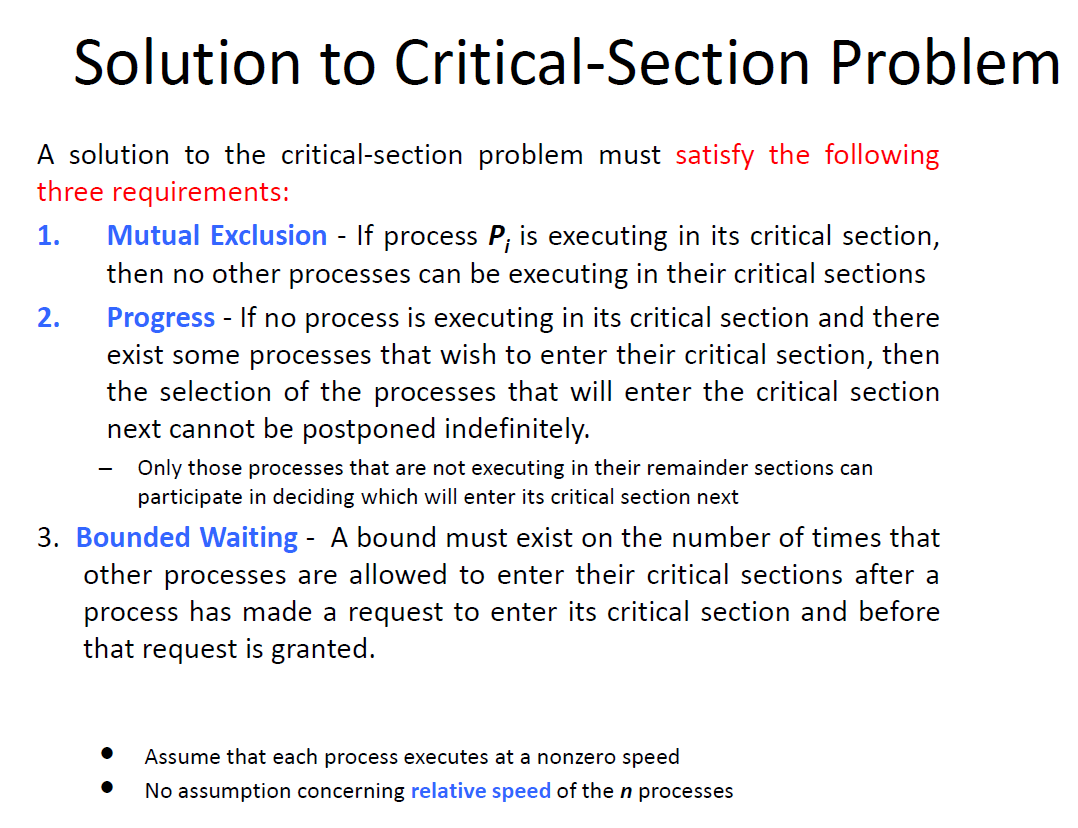
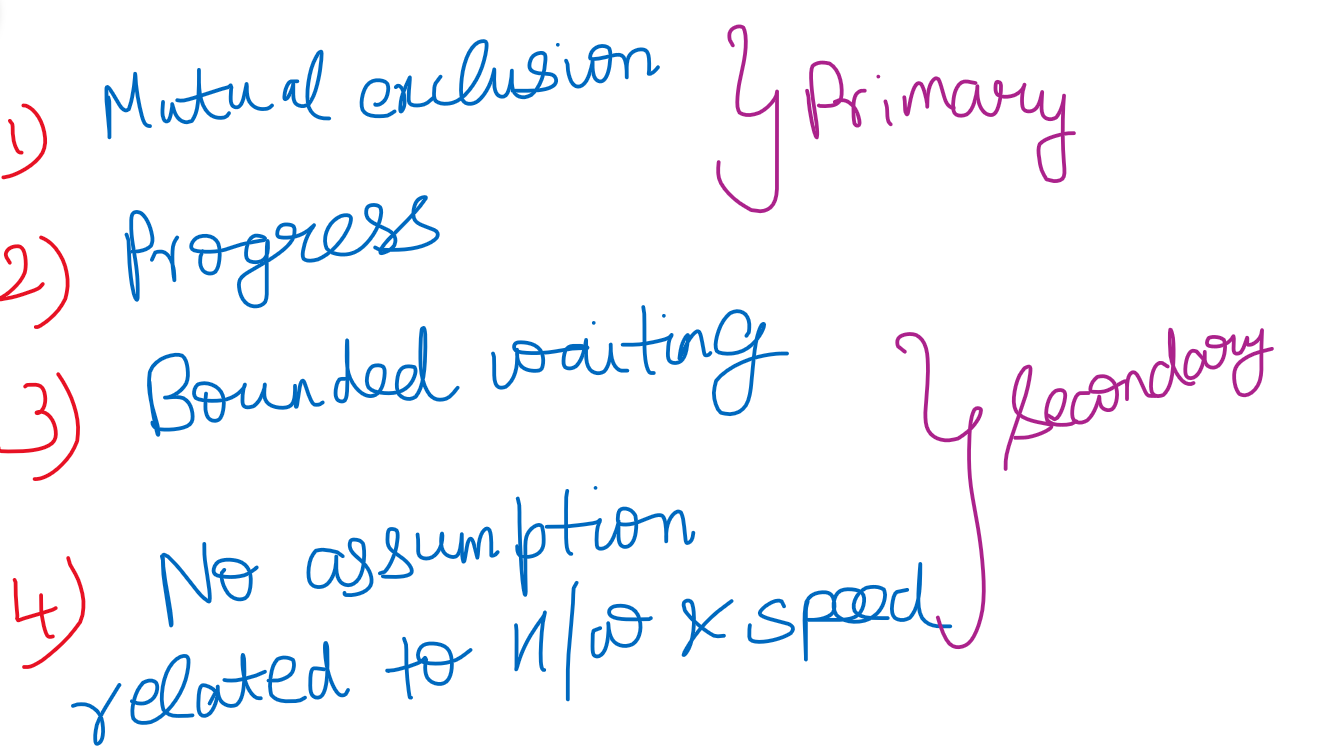


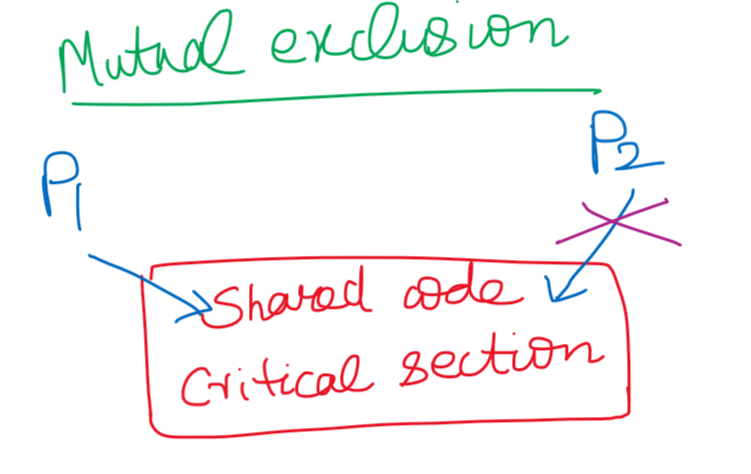
Critical Section 🡪 It is the part of the program where the shared resources are accessed by various process(i.e co-operative process)  

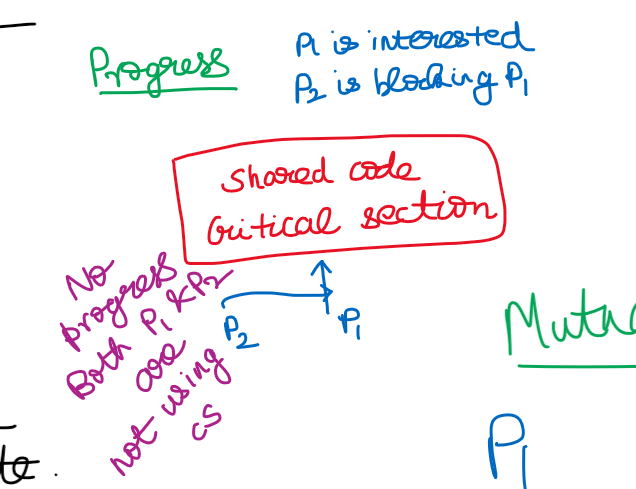



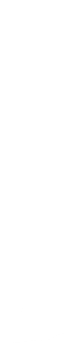


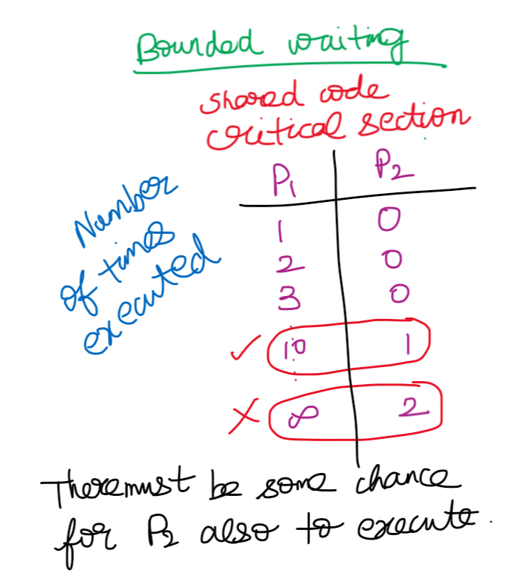




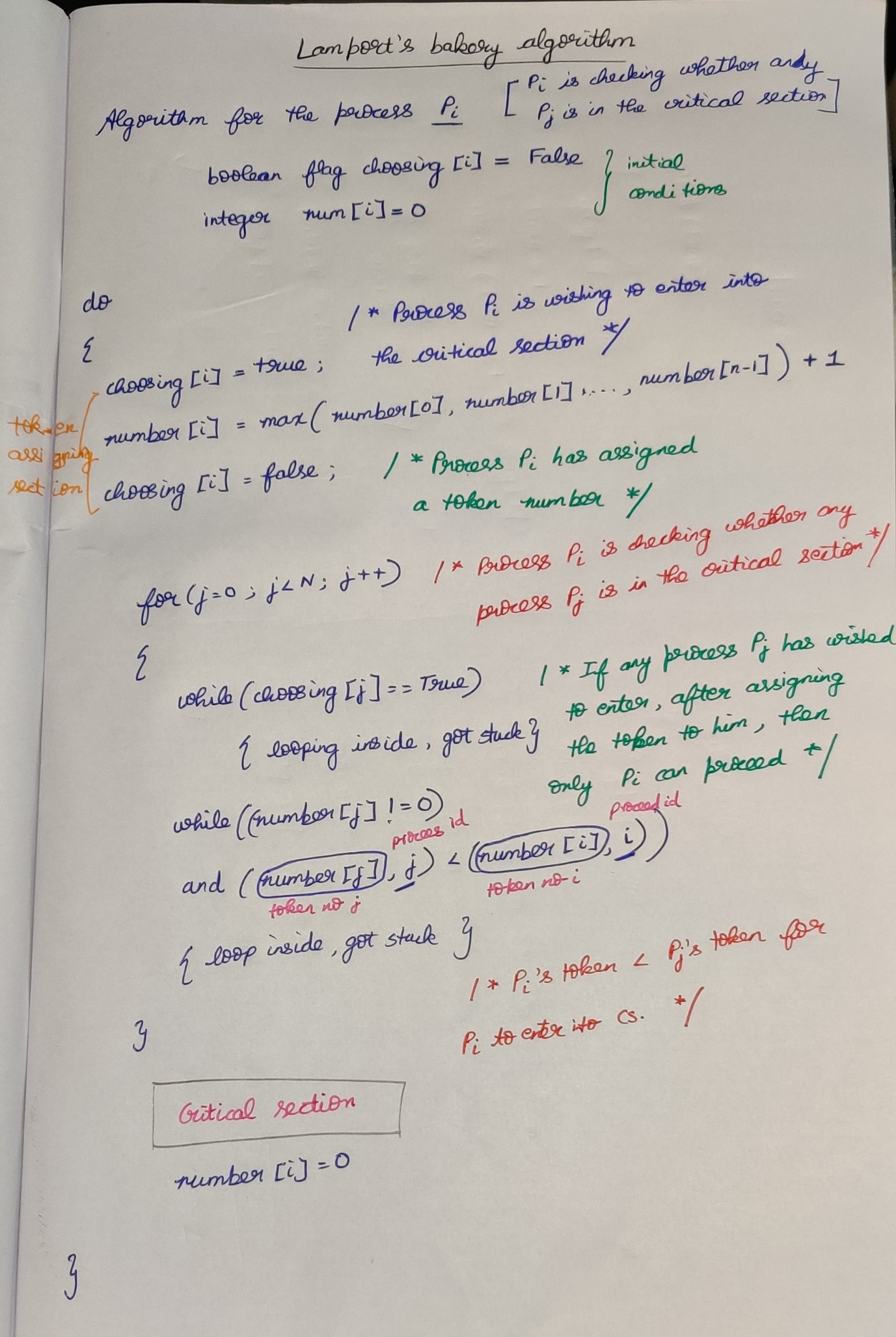






**Lamport’s Bakery Algorithm(N-process solution)**

* Works for the synchronization of an arbitrary set of N co-operative process.
* This algorithm is used when n number of processes wish to enter in their critical section at the same time.
* Before entering into its critical section, that particular process receives a token number. Process having the smallest token number will enter into its critical section.
* If the process Pi and Pj receives the same token number and i<j, then the process Pi will be served first and then process Pj.
* The token numbers are always on the increasing order of enumeration.   
  (i.e 1,2,2,3,4,5,5,5,6,…)
* When a process finishes its execution, its token number is set to 0.



**Semaphores to be continued**