CHAPTER # 07

"BOUNDED BUFFER PROBLEM"

→ Producer :
  ↳ Generates items that go into a buffer
  ↳ Maximum buffer capacity = N
  ↳ If the producer fills the buffer, it must wait

→ Consumer :
  ↳ Consumes things from the buffer
  ↳ If there's nothing in the buffer, it must wait

→ Agr multiple threads ye perform krengi to issue ayega.
→ Race condition k ilawa jaha 2nd problem ye hai k agr ek producer data produce nhi krrha tek consumer baar baar empty buffer mein aakr check krrha hoi k data aya k nhi. To is waja se bohut se CPU cycles waste hoshe hoge consumer thread ki waja se sirf ye check krne k liye k data aya ya nhi. This is called polling.
→ To resolve this we use Semaphore.

Producer :	Consumer :
while (true) {	while (true) {
// produce an item	
wait (empty) → agr empty jaga nhi hai to wait kro means when empty=0 then wait	wait (full) → wait kro agr jaga full nhi hai
wait (mutex)	wait (mutex)
// CS	// CS
signal (mutex)	
signal (full) → full ko signal dedo k ek jaga bhar chuki hai	signal (mutex);
	signal (empty); → empty ko signal do k ek jaga khaali hogai hai.
}	}

" READER - WRITER PROBLEM "

→ Multiple Readers data saath read kr sakte hai.

→ Lekn multiple writers, ya writer or readers saath me execute nhi kr sakte.

→ ~~Three~~ Semaphores are used

↳ mutex ⇒ To maintain synchronization b/w diff readers [to protect read count]

↳ rw_mutex ⇒ " " " " reader & writer.

writer:

```
while (true) {
    wait (rw_mutex) {
        // writing performed (CS)
    signal (rw_mutex );

}
```

jab writer CS mein hoga to or koi order nhi jaasakta jab tak writer baahir na nikal jaye.

wrt_mutex = 1
mutex = 1
ctr = 1

Reader:

```
while (true) {

    wait (mutex) ;
    read_count++;
    if (read_count == 1)
        wait(rw_mutex);
    signal(mutex);
    // perform Reading - (CS)
    wait (mutex);
    reader_count--;
    if(count == 0)
        signal(rw_mutex);
    signal(mutex);
}
```

ye bs isliye kiya hai taake ek waqt mein ek reader hi read_count ko access kr sake.

ye bs check krega 1st reader ko. 1st reader jo hai mutex ko lock krdega taake koi writer order na bajaake. Baaqi jo readers honge wo aasakte hai.

ye last reader ko check krega. Last reader kizimedari hai k wo writer ko signal de k wo CS access kr sakta hai

→ "First readers-writer" Problem: where a writer process ~~them~~ never writes. It requires that no reader be kept waiting unless a writer has already obtained permission to use shared object. Here writers may starve.

→ "Second-readers-write" Problem: Once a writer is ready, that writer perform its write as soon as possible. Here readers may starve.

"DINING PHILOSOPHER PROBLEM"