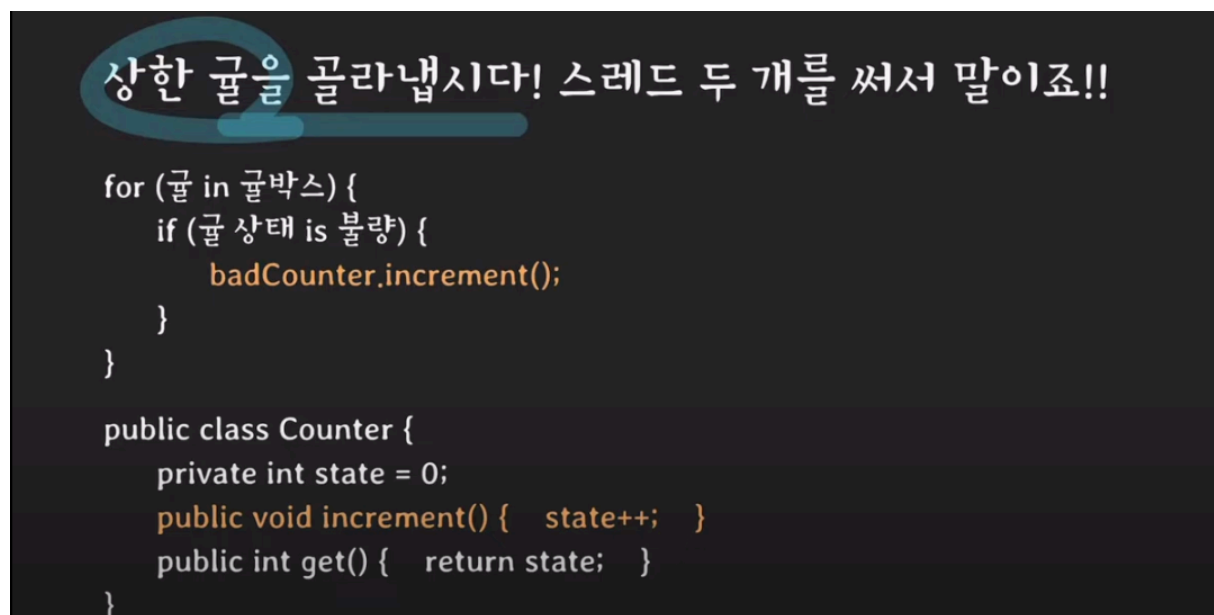


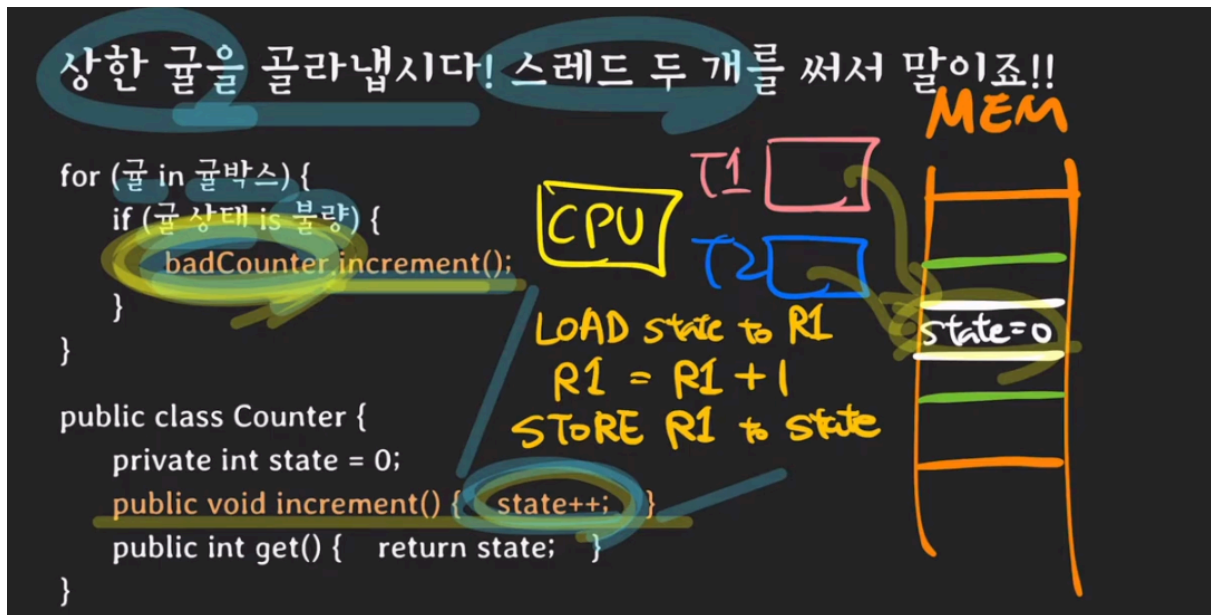
OS 4강

해당 내용은 유튜브 쉬운코드 님의 강의를 개인적인 공부 차원에서 정리하고자 작성한 자료입니다.

해당 회차는 프로세스 동기화가 발생할 수 있는 조건에 대해 말하고 있다



굴 갯수를 체크하는 메서드를 통해 동기화 문제를 지적하고 있다



increment 메서드 안의 state++ 는 프로그램 명령어지 실제로 CPU가 수행하는데는

Load state to R1

R1 = R1 + 1

Store R1 to state

세가지 작업으로 이뤄져있다.

여기서 카운팅에 문제가 발생할 수 있는게

동시에 스레드가 접근했을때

T1이 스레드가 LOAD까지 실행하고 +1를 했을때

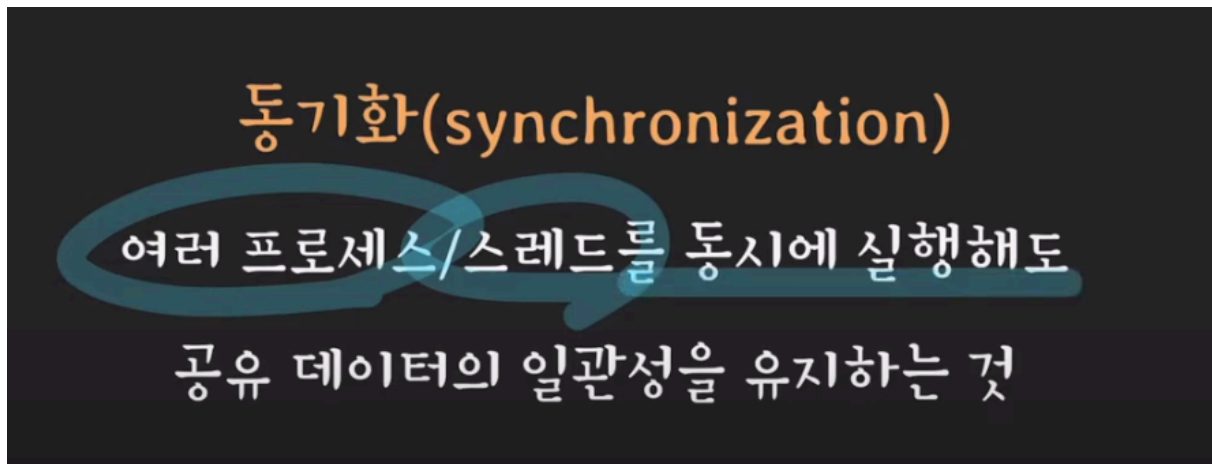
T2가 LOAD 작업에 들어간다면 T1의 연산 작업을 덮어씌울 수 있는것이다

이런 상황을 경쟁조건이라고 한다.

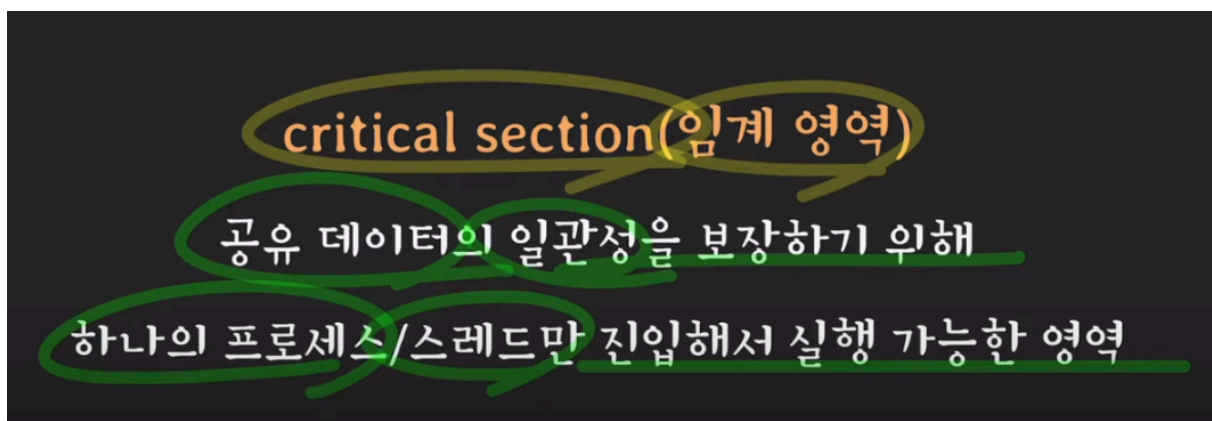
race condition(경쟁 조건)

여러 프로세스/스레드가 동시에 같은 데이터를 조작할 때
타이밍이나 접근 순서에 따라 결과가 달라질 수 있는 상황

스레드나 프로세스를 동시에 실행해도 경쟁조건을 제거하고 일관성을 유지하는 것이 동기화다.



해당 문제를 해결하기 위해 일반적으로 하나의 스레드 / 프로세스만 작업할 수 있게 만들어주는 방법이 있는데, 이것을 임계 영역이라고 한다



critical section problem의 해결책이 되기 위한 조건

1. mutual exclusion (상호 배제)
2. progress (진행)
3. bounded waiting (한정된 대기)

임계 영역 문제의 솔루션이 되기 위해서는 3가지 조건이 필요하다.

1. 상호 배제 (한번의 하나의 프로세스 / 스레드만 진입할 수 있게)
2. 진행 (만약에 임계영역이 비어있고, 스레드가 들어가길 원한다면 그중에 하나는 임계 영역 안에서 실행될 수 있게 해야한다)
3. 한정된 대기 (스레드나 프로세스가 들어가기 위해서 무제한으로 대기해서는 안된다)

++

Thread-unsafe를 조심하세요

SimpleDateFormat

Date formats are not synchronized.

It is recommended to create separate format instances for each thread. If multiple threads access a format concurrently, it must be synchronized externally.

우리가 자주 사용하는 메서드들중에서도 thread-safety 하지 않은 메서드들이 있으니 주의해서 사용할 것