12장 프로세스 동기화

프로세스 동기화란?

- 프로세스들 사이의 수행 시기를 맞추는 것을 의미
  - 실행 순서 제어 : 프로세스를 올바른 순서대로 실행하기
    - 특정 조건이 만족되어야만 실행할 수 있는 상황에서 올바른 순서대로 실행하게 하는 것
  - 상호 배제 : 동시에 접근해서는 안 되는 자원에 하나의 프로세스만 접근하게 하기
- 프로세스뿐만 아니라 스레드도 동기화 대상 → 실행의 흐름을 갖는 모든 것은 동기화의 대상임

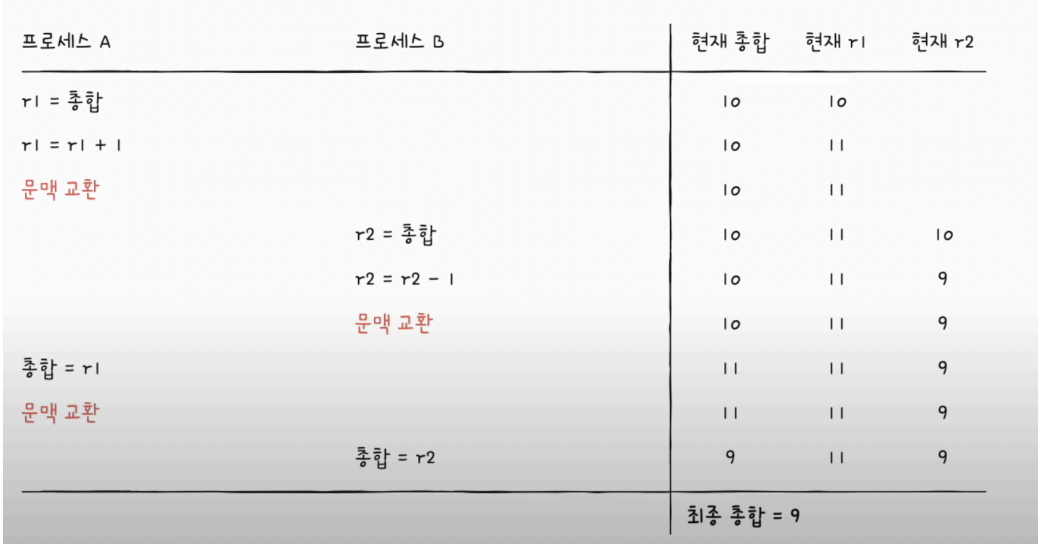
생산자와 소비자 문제

> 총합이 10일 떄 생산자는 총합에 1을 증가시키고 소비자는 총합에 1을 감소 시킬 때 생산자와 소비자를 동시에 실행시키면?

- 계속 10에 머물러 있지 않고 이후 합계를 예상할 수 없음
- 이는 제대로 동기화되지 않았기 때문에 발생한 문제인데, **동시에 접근해서는 안 되는 자원에 동시에 접근했기에 발생한 문제**

공유 자원과 임계 구역

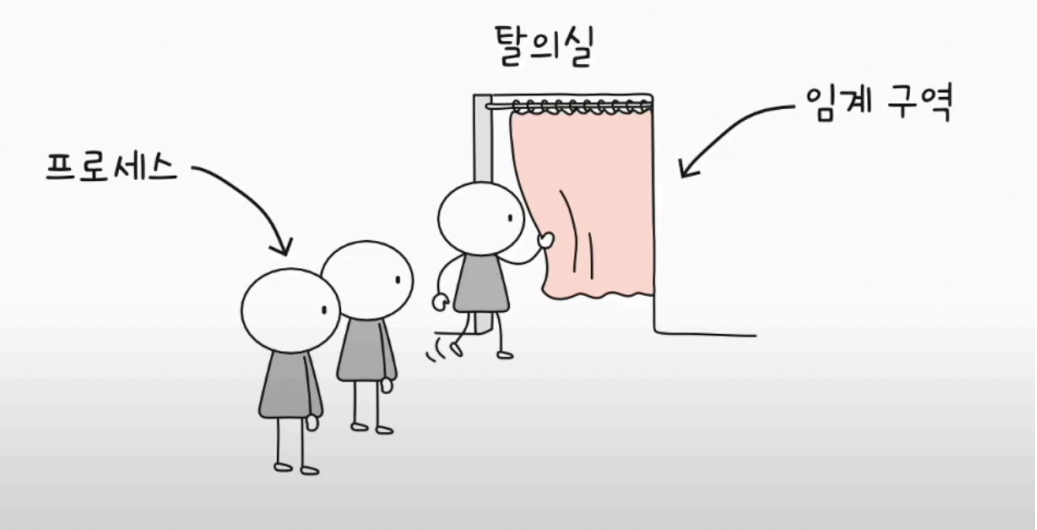
- **공유 자원** : 전역 변수, 파일, 입출력장치, 보조기억장치
- **임계 구역** : 동시에 실행하면 문제가 발생하는 자원에 접근하는 코드 영역
- **레이스 컨디션 :** 두 개 이상의 프로세스가 동시에 실행되면 안 되는 영역이지만, 잘못된 실행으로 인해 여러 프로세스가 동시 다발적으로 임계 구역의 코드를 실행하여 문제가 발생하는 경우 → 데이터의 일관성이 깨짐



프로세스 A	프로세스 B	현재 총합	현재 r1	현재 r2
r1 = 총합		10	10	
r1 = r1 + 1		10	11	
문맥 교환		10	11	
	r2 = 총합	10	11	10
	r2 = r2 - 1	10	11	9
	문맥 교환	10	11	9
총합 = r1		11	11	9
문맥 교환		11	11	9
	총합 = r2	9	11	9
		최종 총합 = 9		

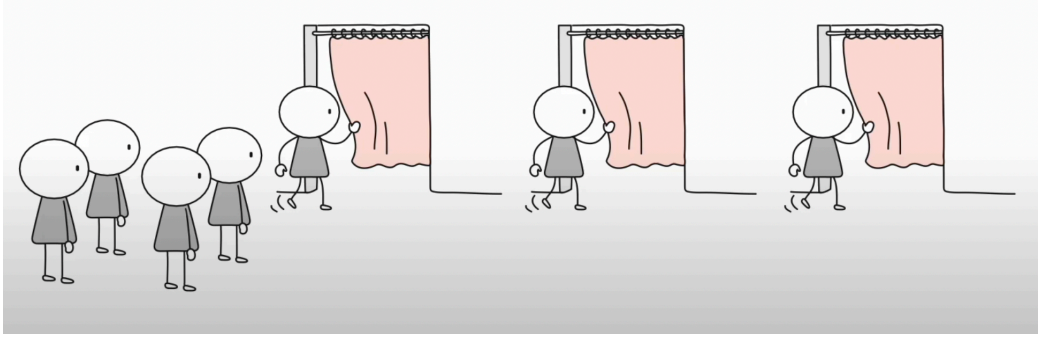
- 여러 줄의 저급 언어로 변환된 고급 언어 한 줄을 실행하는 과정에서 문맥 교환이 일어날 수 있음
- 따라서 `상호 배제를 위한 동기화`는 이와 같은 일이 발생하지 않도록 두 개 이상의 프로세스가 임계 구역에 접근하지 못하도록 관리하는 것을 의미
- **상호 배제를 위한 동기화 원칙**
  - **상호 배제** : 한 프로세스가 임계 구역에 진입했다면 다른 프로세스는 임계 구역에 들어올 수 없다.
  - **진행** : 임계 구역에 어떤 프로세스도 진입하지 않았다면 임계 구역에 진입하고자 하는 프로세스는 들어갈 수 있어야 한다.
  - **유한 대기 :** 한 프로세스가 임계 구역에 진입하고 싶다면 그 프로세스는 언젠가는 임계 궁역에 들어올 수 있어야한다.(즉, 임계 구역에 들어오기 위해 무한정 대기해서는 안 된다)

뮤텍스 락



- 동시에 접근해서는 안 되는 자원에 동시에 접근하지 않도록 만드는 도구. 즉, 상호 배제를 위한 동기화 도구임
- 임계 구역에 진입하는 프로세스느 “내가 지금 임계 구역에 있음”을 알리기 위해 뮤텍스 락을 이용해 임계 구역에 자물쇠를 걸어둘 수 있고, 다른 프로세스는 임계 구역이 잠겨 있다면 기다리고, 잠겨있지 않다면 임계 구역에 진입할 수 있음
- 자물쇠 역할 : 프로세스들이 공유하는 전역 변수 Lock
- 임계 구역을 잠그는 역할 : acquire 함수
  - 프로세스가 임계 구역에 진입하기 전에 호출하는 함수
  - 만일 임계 구역이 잠겨 있다면 임계 구역이 열릴 때까지 임계 구역을 반복적으로 확인하고, 임계 구역이 열려 있다면 임계 구역을 잠그는 함수
  - 바쁜 대기 : 임계 구역이 잠겨 있을 경우, 프로세스는 반복적으로 lock을 확인하는 작업
- 임계 구역의 잠금을 해제하는 역할 : release 함수
  - 임계 구역에서의 작업이 끝나고 호출하는 함수

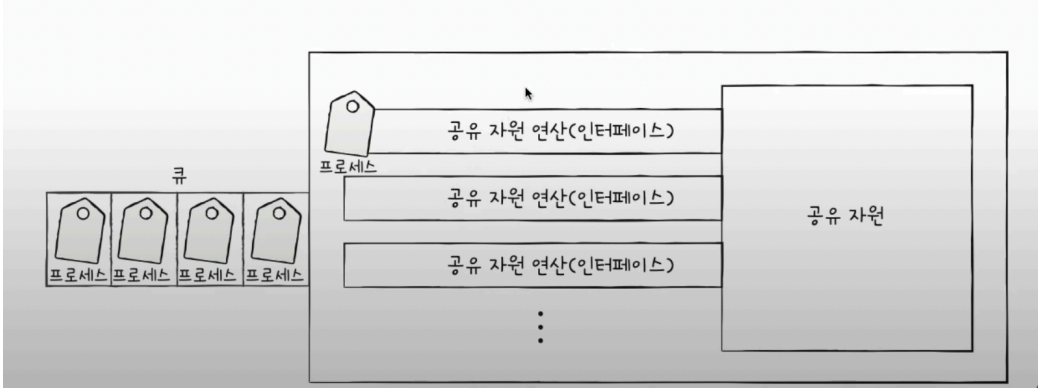
세마포



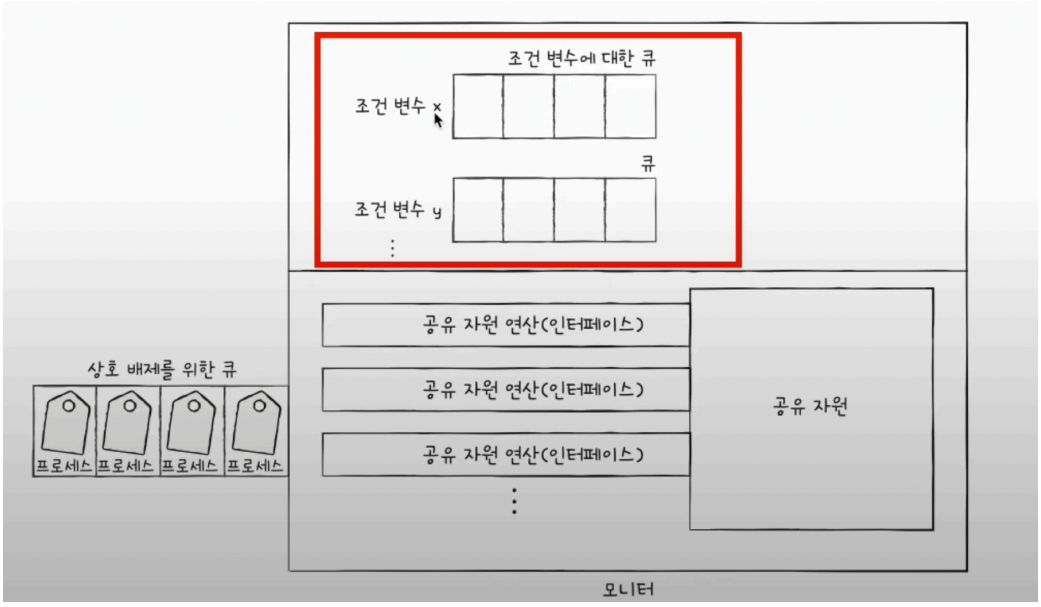
- 공유 자원이 여러 개 있는 상황에서도 적용이 가능한 동기화 도구
  - 이진 세마포(뮤텍스 락과 동일)와 카운팅 세마포가 있음
- 임계 구역에 진입할 수 있는 프로세스의 개수(사용 가능한 공유 자원의 개수)를 나타내는 전역 변수 S
- 임계 구역에 들어가도 좋은지, 기다려야 할지를 알려주는 wait 함수
- 임계 구역 앞에서 기다리는 프로세스에 '이제 가도 좋다'고 신호를 주는 signal 함수
- `세마포`도 `바쁜 대기` 가 발생함 → CPU 주기를 낭비한다는 점에서 손해임
  - 그러나 뮤텍스보다 좀 더 좋은 방법을 사용함
  - wait 함수는 만일 사용할 수 있는 자원이 없을 경우 해당 프로세스 상태를 대기 상태로 만들고, 그 프로세스의 PCB를 세마포를 위한 대기 큐에 집어넣음
  - 다른 프로세스가 임계 구역에서의 작업이 끝나고 signal 함수를 호출하면 signal 함수는 대기 중인 프로세스를 대기 큐에서 제거하고, 프로세스 상태를 준비 상태로 변경한 뒤 준비 큐로 옮겨줌

모니터

- 세마포의 경우 매번 임계 구역에 앞뒤로 일일일 wait와 signal 함수를 명시하는 것은 번거로운 일
  - 세마포를 누락시키거나, wait와 signal 순서를 헷갈린 경우, wait와 signal을 중복해서 사용한 경우



- 모니터는 공유 자원을 다루는 인터페이스에 접근하기 위한 큐(모니터에 진입하기 위한 큐)를 만들고, 모니터 안에 항상 하나의 프로세스만 들어오도록하여 상호 배제를 위한 동기화를 제공
- 특정 조건을 바탕으로 프로세스를 실행하고 일시 중단하기 위해 `조건 변수` 를 사용하는데, 조건 변수는 프로세스나 스레드의 실행 순서를 제어하기 위해 사용하는 특별한 변수
  - 특정 프로세스가 아직 실행될 조건이 되지 않았을 때는 wait를 통해 실행을 중단한다.
  - 특정 프로세스가 실행될 조건이 충족되었을 때에는 signal을 통해 실행을 재개한다.



- 상호 배제를 위한 큐 : 모니터에 한 번에 하나의 프로세스만 진입하도록 하기 위해 만들어진 큐
- 조건 변수에 대한 큐 : 모니터링에 이미 진입한 프로세스의 실행 조건이 만족할 때까지 잠시 실행이 중단되어 기다리기 위해 만들어진 큐