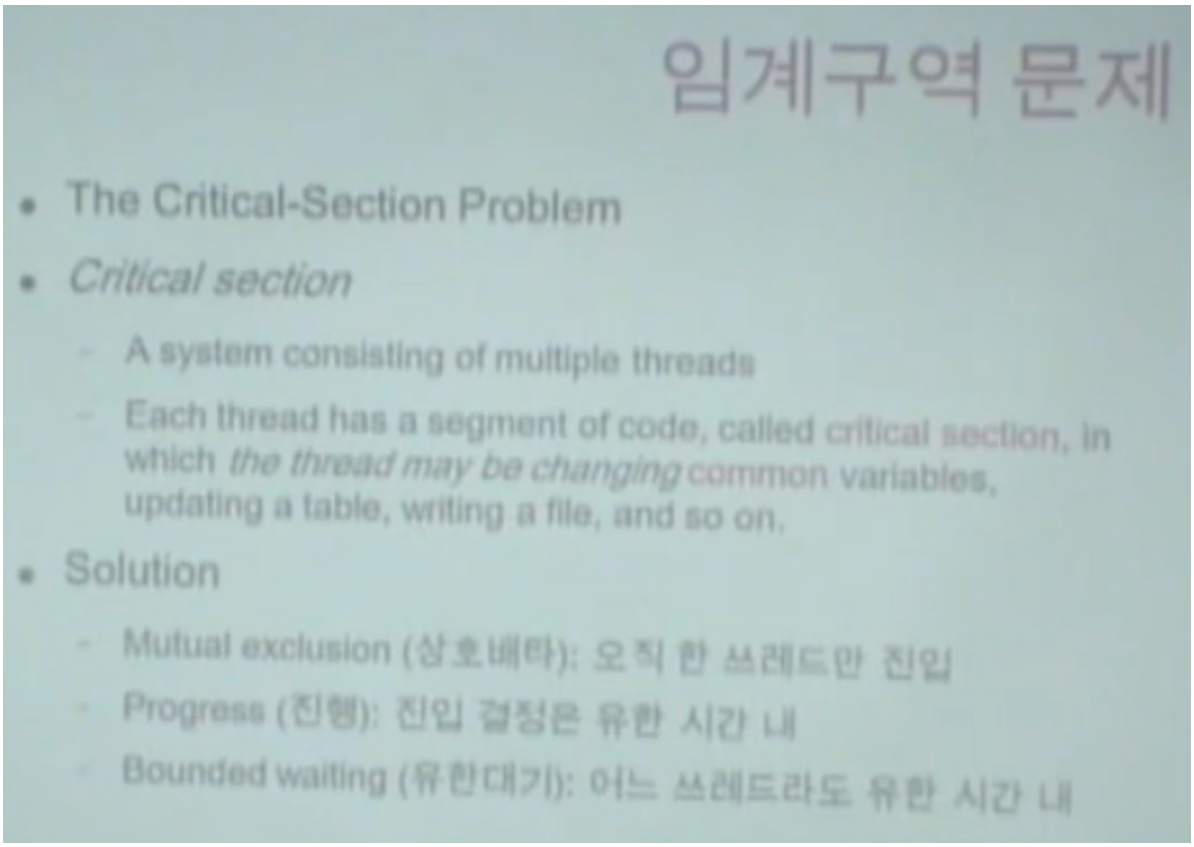
운영체제

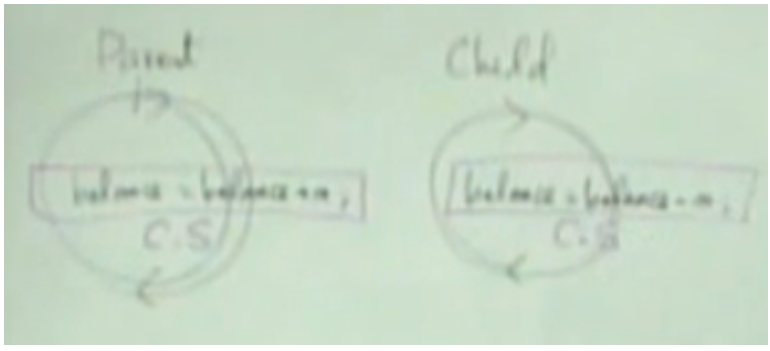
05 임계구역 문제

---

1. 임계구역 문제



1 - 1) 예시



부모: 무한루프를 들며 balance + n

자식: 무한루프를 돌며 balance - n

두개의 threads

common variable을 업데이트 하는 상태

동일한 값에 대해서 동시에 업데이트를 하려고 하는 문제가 발생

1 - 2) 해결법

1. 상호배타(mutual exclusion): 오직 한 스레드만 진입
2. 진행(progress): 진입 결정은 유한 시간 내 일어나야 함, 누가 먼저 들어갈까
3. 유한대기(bounded waiting): 어느 스레드라도 유한 시간 내 들어갈 수 있어야 함

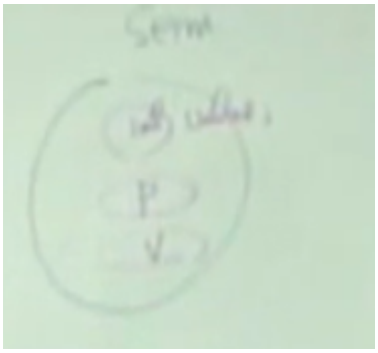
세 가지 조건을 만족해야 임계구역 문제를 해결 할 수 있다.

2. Semaphores(세마포)

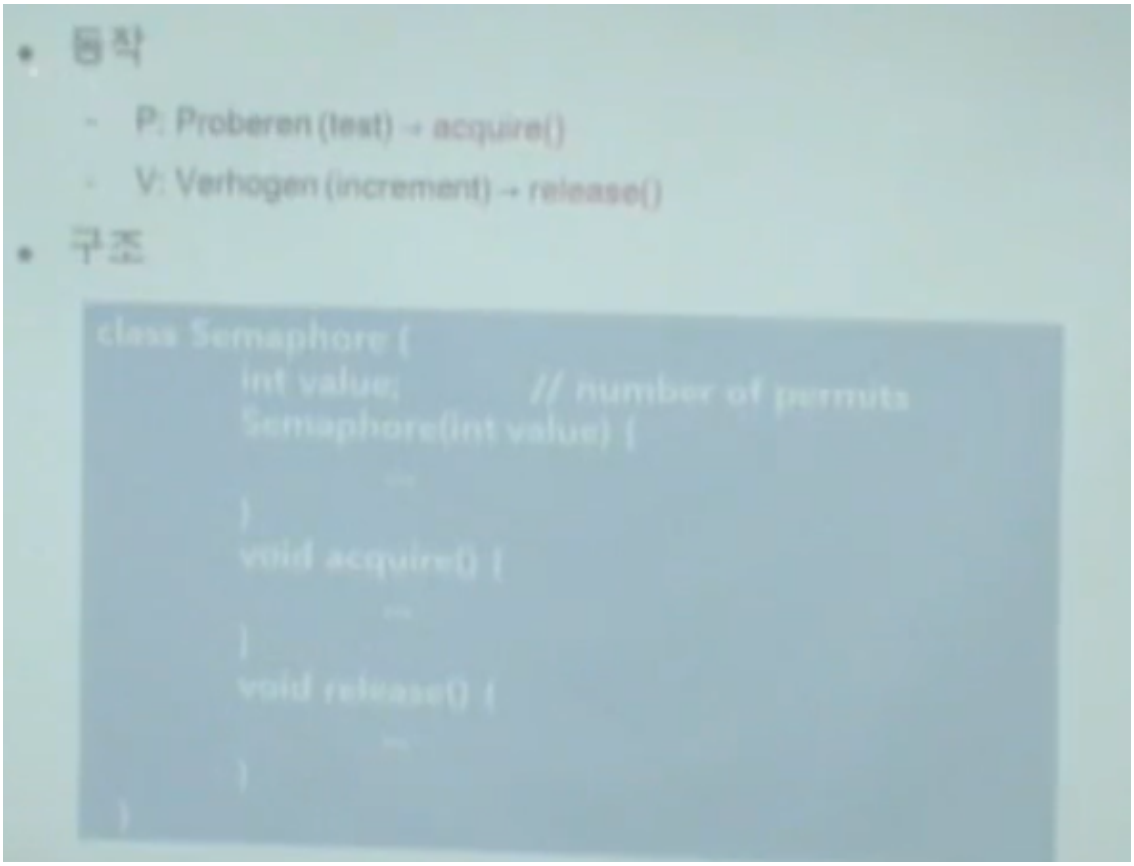
2 - 1) 동기화 도구

- semaphores
- monitor
- amc

2 - 2) semaphores란



- 동기화 문제 해결을 위한 소프트웨어도구
- 구조: 정수형변수 + 두개의 동작
-



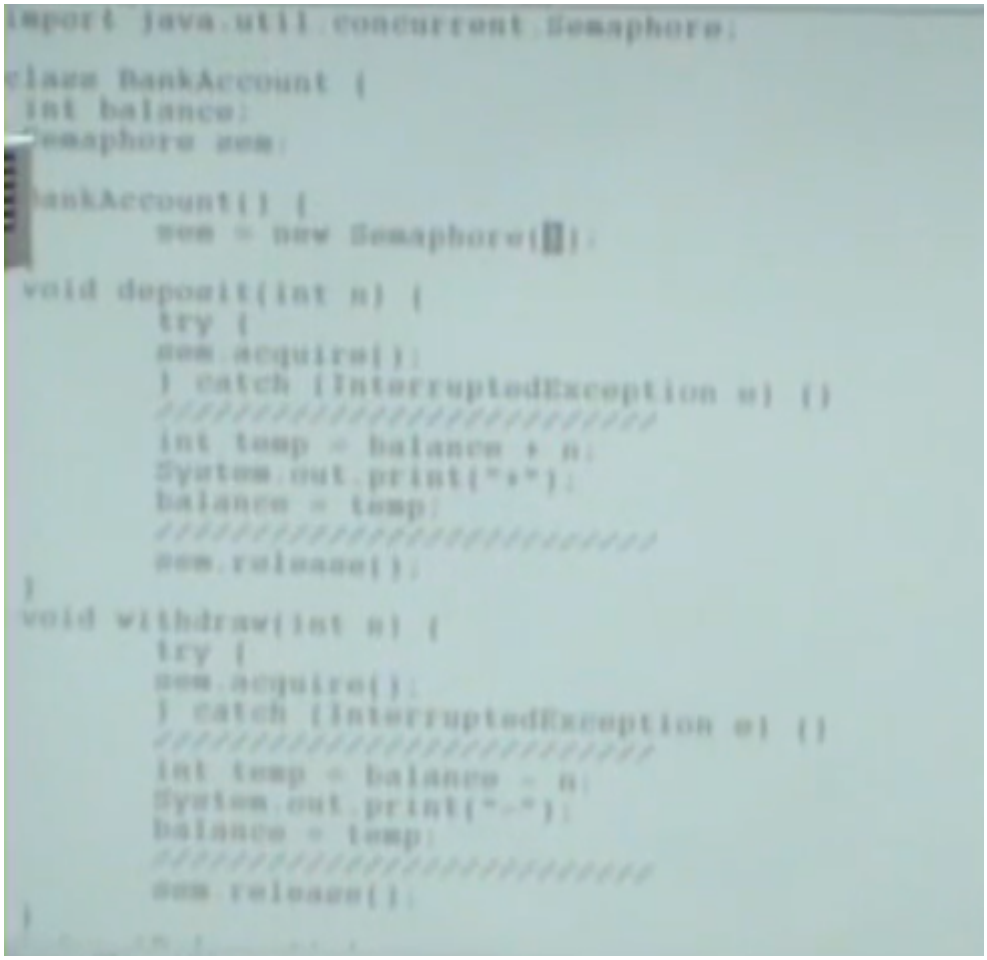
- acquire

```
<acquire>

정수값 1만큼 감소
0보다 작으면 queue안에 집어넣고 block
```

- release

```
<release>

정수값을 1만큼 증가
0보다 작거나 같으면 queue에서 꺼내서 해방
```

2 - 3) semaphores 예시



```
import java.util.concurrent.Semaphore;

class BankAccount {
 int balance;
 Semaphore sem;

 BankAccount() {
        sem = new Semaphore(1);

void deposit(int n) {
        try {
        sem.acquire();
        } catch (InterruptedException e) {}
        ////////////////////////////
        int temp = balance + n;
        System.out.print("+");
        balance = temp;
        ////////////////////////////
        sem.release();
}
void withdraw(int n) {
        try {
        sem.acquire();
        } catch (InterruptedException e) {}
        ////////////////////////////
        int temp = balance - n;
        System.out.print("-");
        balance = temp;
        ////////////////////////////
        sem.release();
}
```

시간 지연은 있을지라도 엉뚱한 결과가 나오는 일은 없음

결과값: balance는 0