**[프로세스란?]**

**3. 동기화와 데드락**

1. **동기와 비동기(Sync & Async)**

: 메소드를 실행시킴과 동시에 반환 값이 기대되는 경우를 ‘동기’라고 표현하고 그렇지 않은 경우를 ‘비동기’라고 표현한다. ‘동시에’라는 말은 실행되었을 때 값이 반환되기 전까지는 Blocking 되어 있다는 것을 의미한다. 비동기의 경우, Blocking 되지 않고 이벤트 큐에 넣거나 백그라운드 스레드에게 해당 Task를 위임하고 바로 다음 코드를 실행하기 때문에 기대되는 값이 바로 반환되지 않는다.

1. **프로세스 동기화**

**2-1) 임계영역(Critical Section)**

: 멀티 스레딩의 문제점에서 나오듯 동일한 자원을 동시에 접근하는 작업(공유 변수 사용, 동일 파일 사용 등)을 실행하는 코드 영역을 칭한다.

**2-2) 임계영역 문제(Critical Section Problem)**

: 프로세스들이 임계영역을 함께 사용할 수 있는 프로토콜을 설계하는 것이다.

1. **해결을 위한 기본 조건(Requirements)**
2. **상호 배제(Mutual Exclusion)**

: 프로세스 P1이 임계영역에서 실행 중이라면, 다른 프로세스들은 그들이 가진 임계영역에서 실행될 수 없다.

1. **진행(Progress)**

: 임계영역에서 실행 중인 프로세스가 없고, 별도의 동작이 없는 프로세스들만 임계영역 진입 후보로서 참여될 수 있다.

1. **한정된 대기(Bounded Waiting)**

: 프로세스 P1가 임계영역에 진입 신청 후부터 받아들여질 때까지, 다른 프로세스들이 임계영역에 진입하는 횟수는 제한이 있어야 한다.

1. **해결책**
2. **Lock**

: 하드웨어 기반 해결책으로써, 동시에 공유 자원에 접근하는 것을 막기 위해 임계영역에 진입하는 프로세스는 Lock을 획득하고, 임계영역을 빠져나올 때, L:ock을 방출하면서 동시에 접근이 되지 않도록 한다.

**(한계)** 다중처리기 환경에서는 시간적인 효율성 측면에서 적용할 수 없다.

1. **Semaphores(세마포)**

: 소프트웨어 기반 해결책으로써, 임계영역 문제를 해결하기 위한 동기화 도구

**(종류)** OS는 Counting / Binary 세마포를 구분한다.

**(단점)** Busy Waiting(바쁜 대기), Deadlock(교착상태)

1. **모니터**

: 고급 언어의 설계 구조물로서, 개발자의 코드를 상호배제 하게끔 만든 추상화된 데이터 형태이다.

: 공유 자원에 접근하기 위한 Key 획득과 자원 사용 후 해제를 모두 처리한다. (세마포는 직접 Key 해제와 공유 자원 접근 처리가 필요하다.)