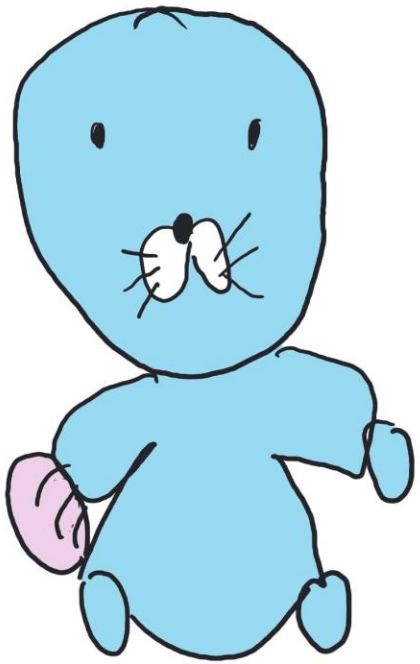
운영체제에서 가장 중요한 파트!!!!!!



가상 메모리

장동호

목차

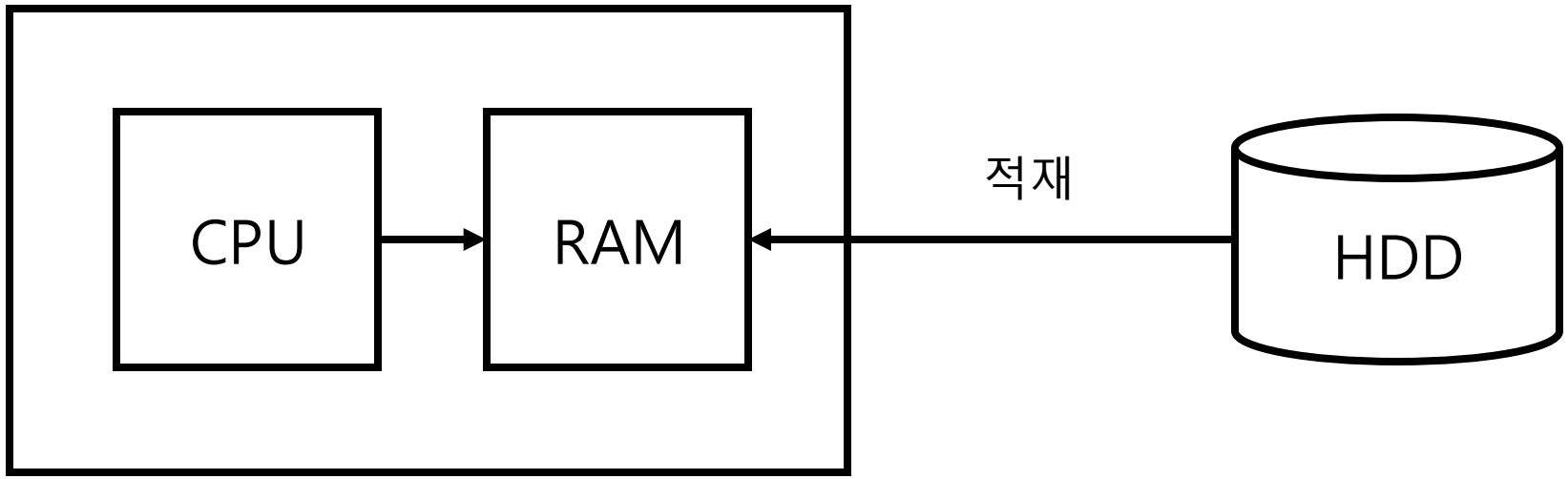
- 배경 지식
- 가상 메모리
- 페이징
- 요구 페이징

질문!!

- 4GB보다 큰 크기의 프로그램을 4GB인 물리 메모리로 실행할 수 있을까요?

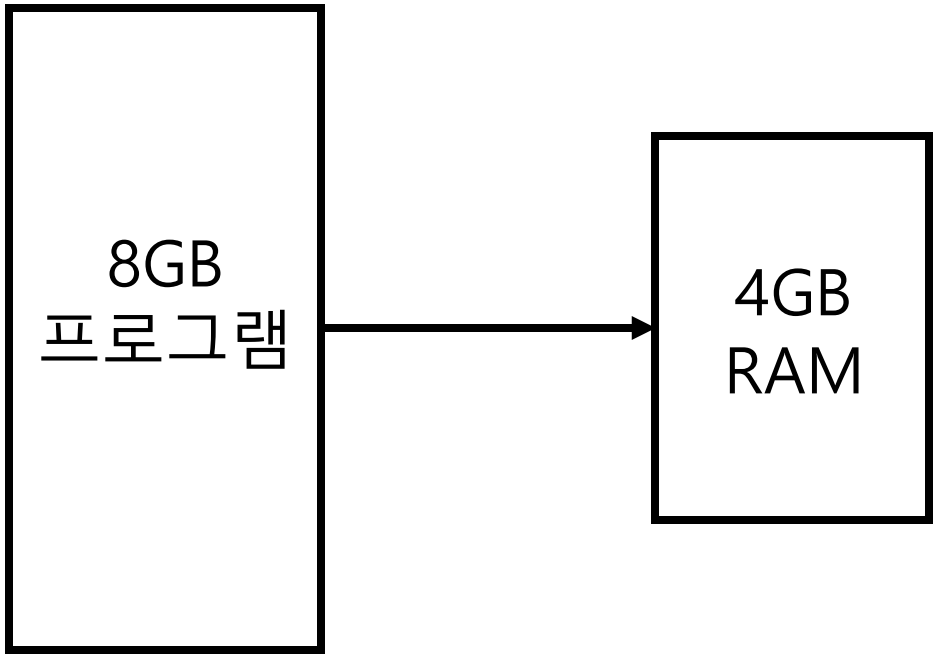
배경 지식

- 실행파일 형태로 존재하는 program이 memory에 적재되어 CPU에 의해 실행되는 것을 프로세스라고 한다.



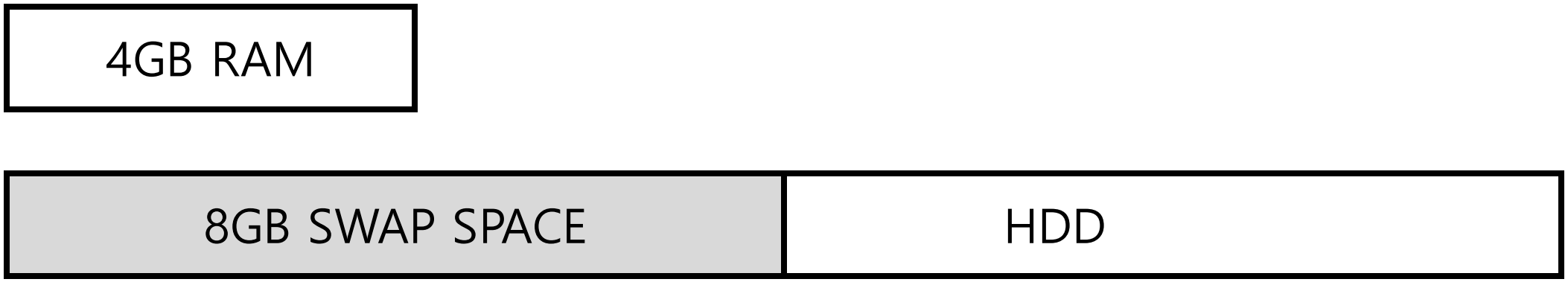
가상 메모리

- 프로그램을 실행시키기 위해서는 memory에 올려야하는데 자리가 없다!! 공간을 마련해야 하는데 어떻게 하지?



가상 메모리

- 실행하고자 하는 프로그램의 일부만 적재해, 실제 메모리보다 더 큰 프로세스를 실행할 수 있도록 만드는 메모리 관리 기법



가상 메모리

- 커넥트 지누 서버가 돌아가는 환경은 메모리가 1GB 밖에 안됨.
- 그래서 HDD의 일부를 스왑 메모리(2GB)로 지정하여 사용중!

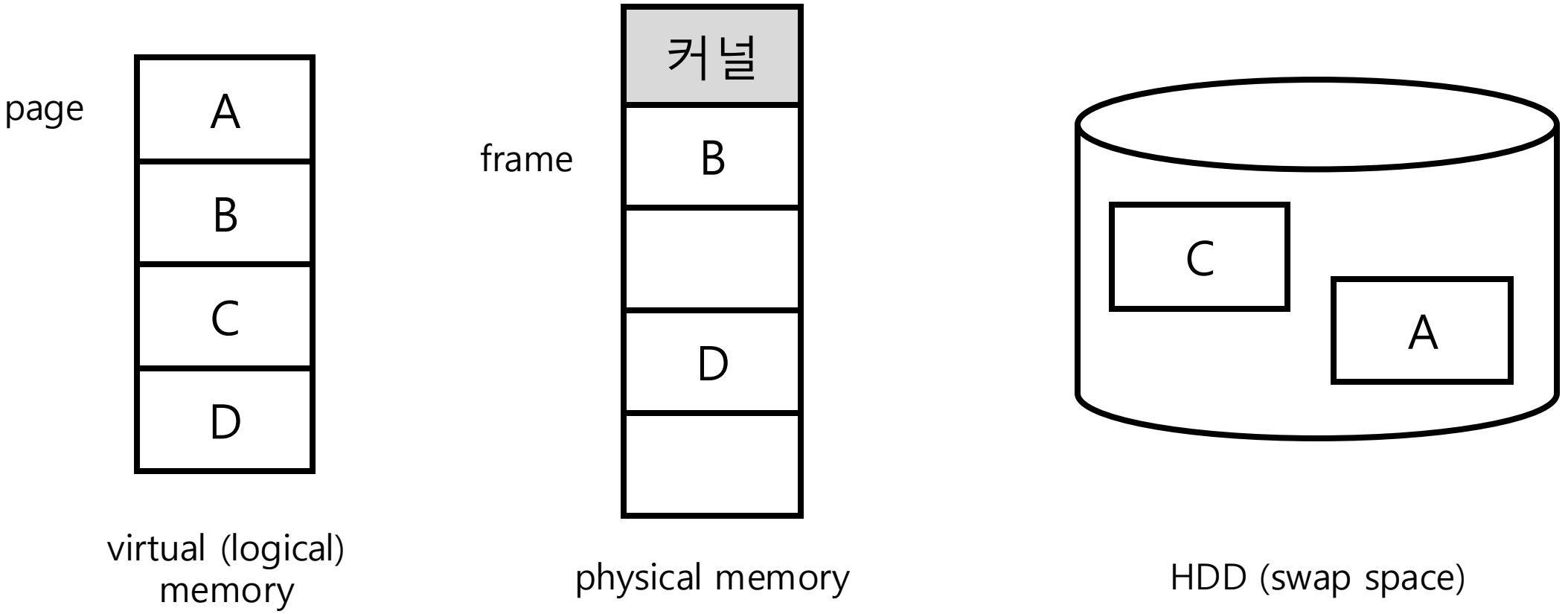
```
donghо@connectgnu-instance:~$ free -h
               total        used        free      shared  buff/cache   available
Mem:           954Mi       581Mi        64Mi       0.0Ki       308Mi       221Mi
Swap:          2.0Gi       374Mi       1.6Gi
```

꼬리 질문!!

- 무턱대고 그렇게 짤라서 넣으면 프로그램 실행에 이상이 생기지 않나요? 자르는 기준이 뭐죠?

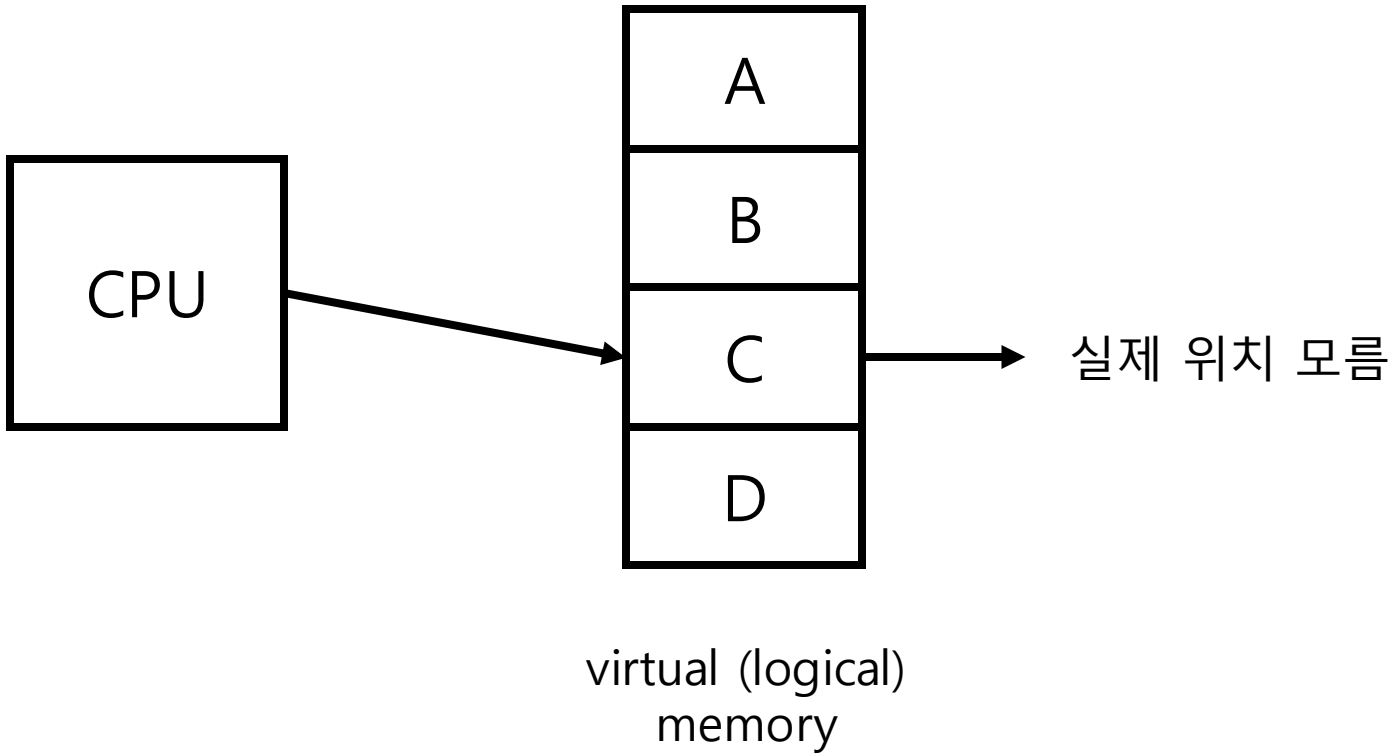
페이징

- 프로세스의 논리 주소 공간을 **페이지**로 나누고, 동일 크기의 **프레임** 단위로 물리 메모리에 할당하는 가상 메모리 관리 기법



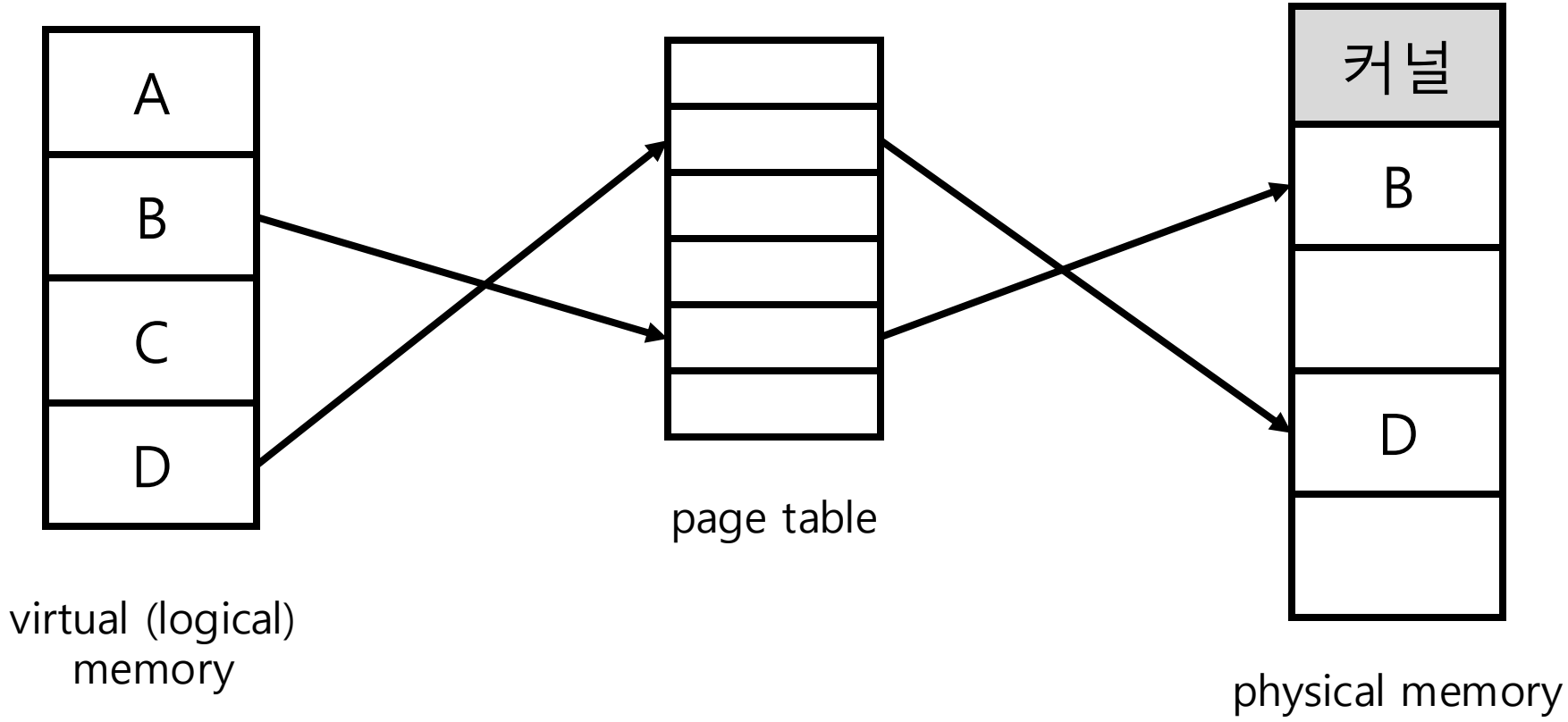
페이지 테이블

- 당장 사용할 공간을 page 단위로 메모리에 적재하는 방법



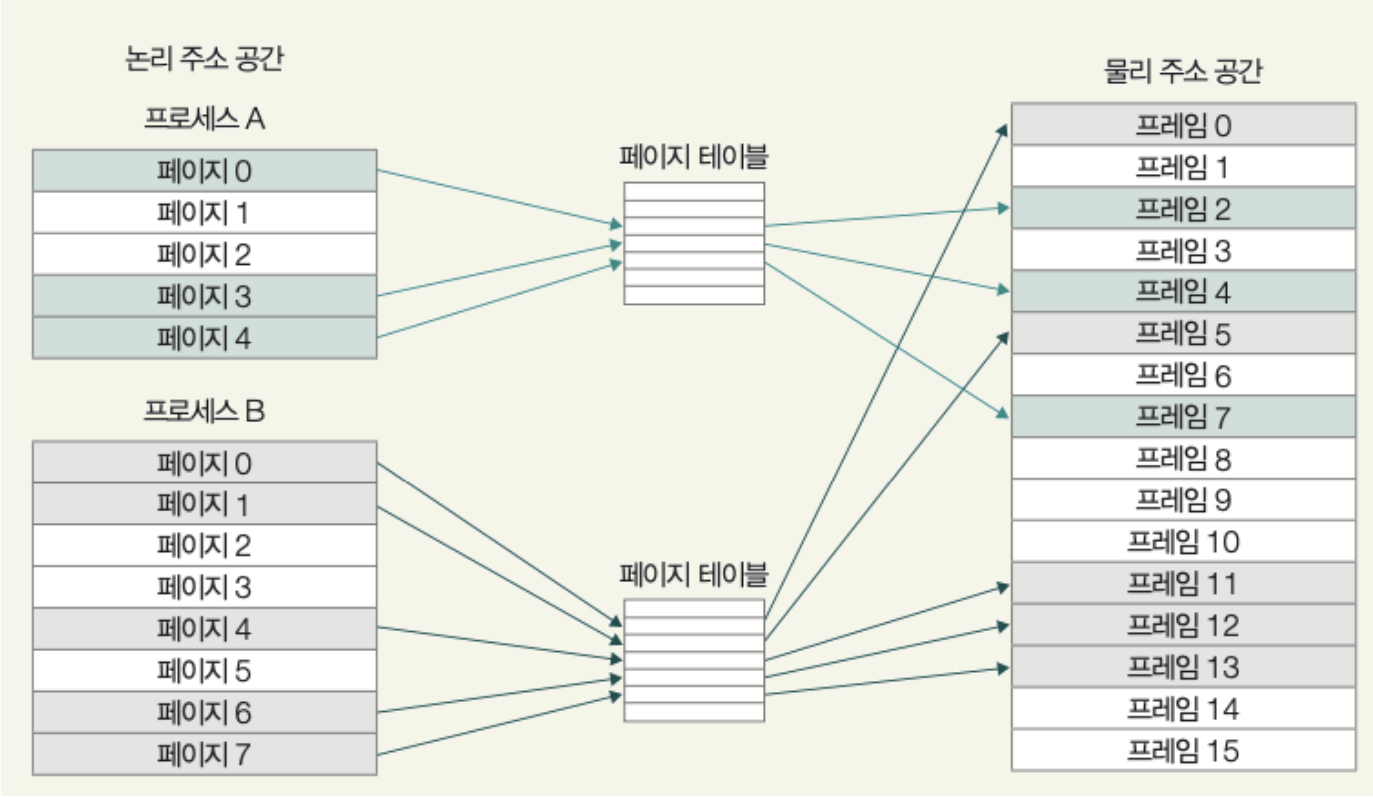
페이지 테이블

- 프로세스의 페이지와 실제로 적재된 프레임을 매핑해주는 역할



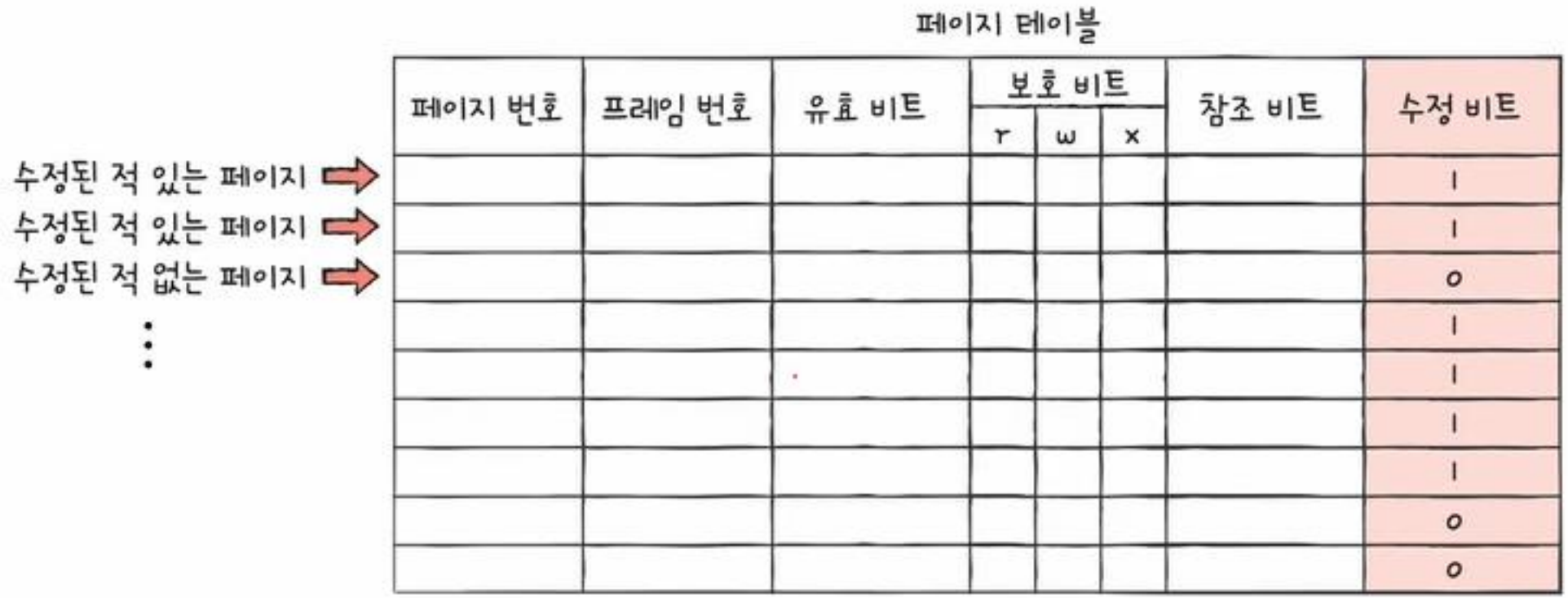
페이지 테이블

- 프로세스의 페이지와 실제로 적재된 프레임을 매핑해주는 역할



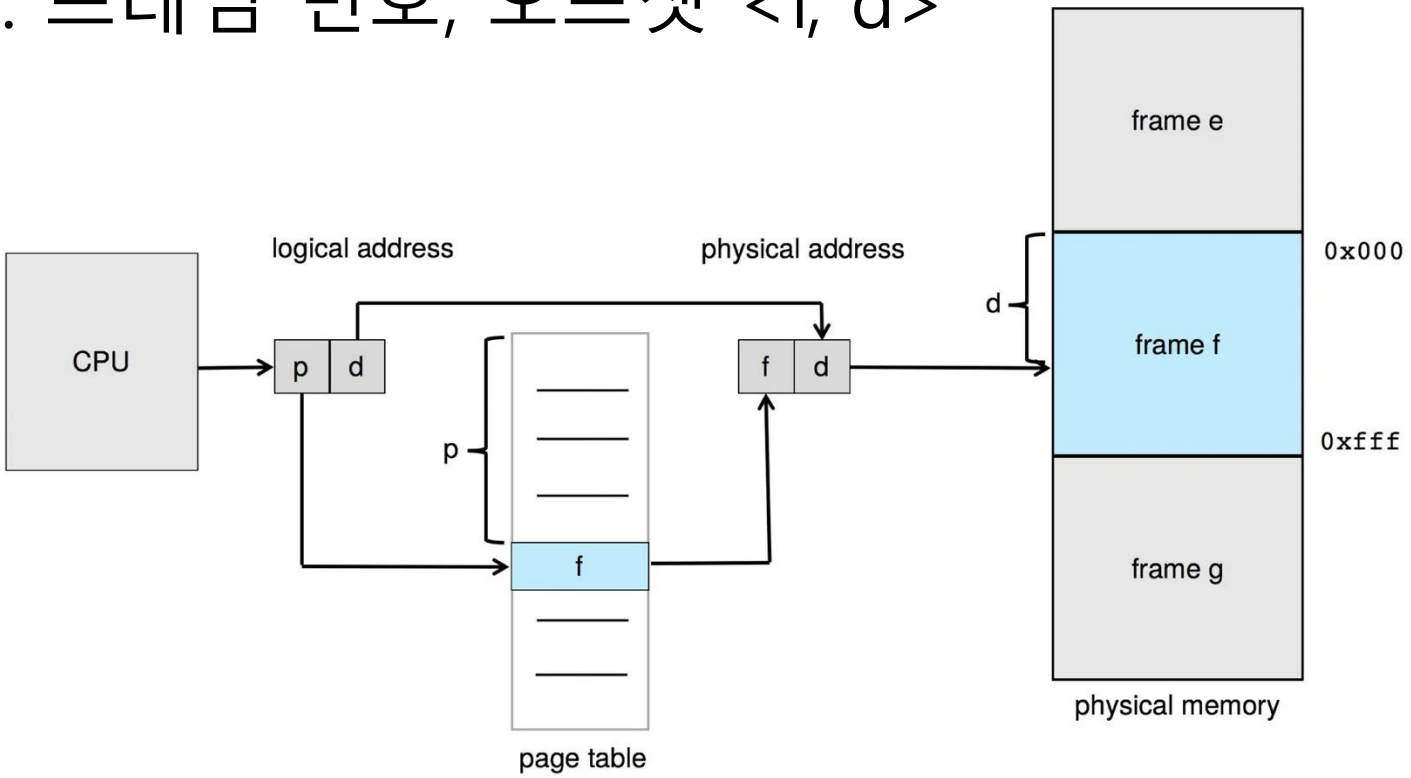
페이지 테이블 엔트리(PTE)

- 프로세스의 페이지와 실제로 적재된 프레임을 매핑해주는 역할



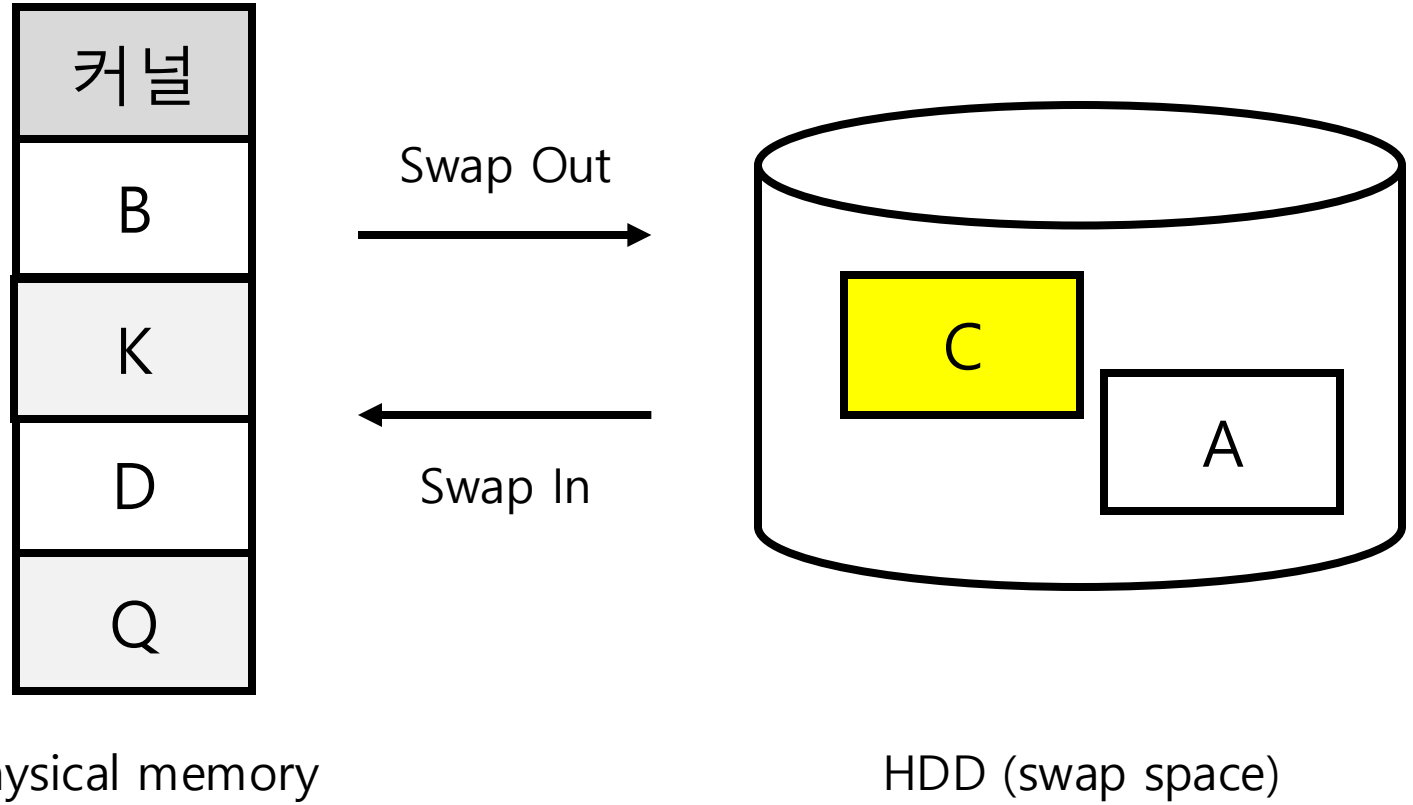
페이징 주소 체계

- 논리 주소: 페이지 번호, 오프셋 <p, d>
- 물리 주소: 프레임 번호, 오프셋 <f, d>



요구 페이징

- 메모리에 필요한 페이지만을 적재하는 기법



페이지 교체 알고리즘

- FIFO (First-In First-Out) -> 전통적인 방법
- 최적 (Optimal) -> 현실적으로 불가능
- **LRU (Least Recently Used) -> 현대 컴퓨터 채택**

참고 자료

- [10분 테코톡] 채채의 가상 메모리
- [10분 테코톡] 배럴의 가상 메모리
- 널널한 개발자 TV 짧게 개념만 이해하는 가상 메모리 구조
- 널널한 개발자 TV Page-in/out, Swap-in/out

감사합니다