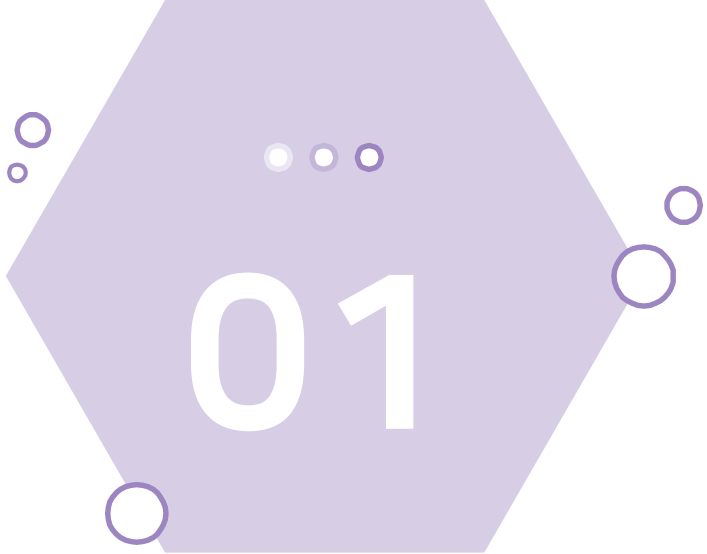
8강. 메모리 관리

방송대 컴퓨터과학과
김진욱 교수

목차

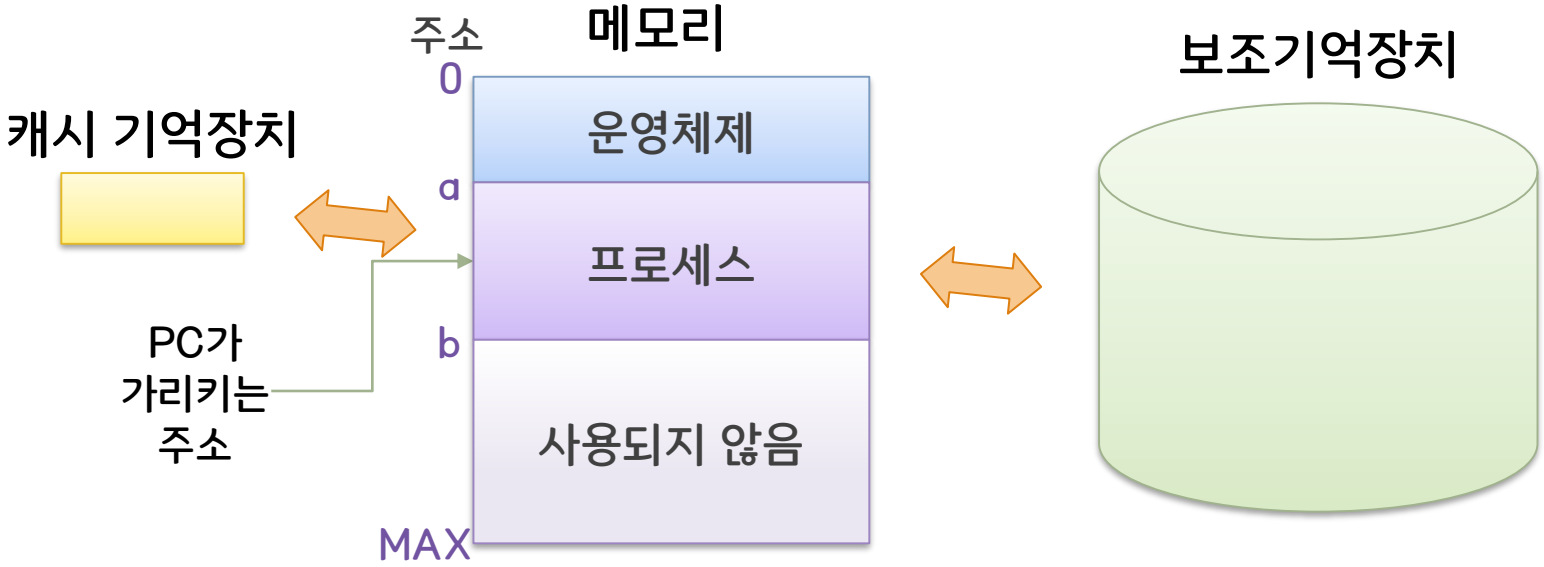


프로세스와 메모리

프로세스와 메모리

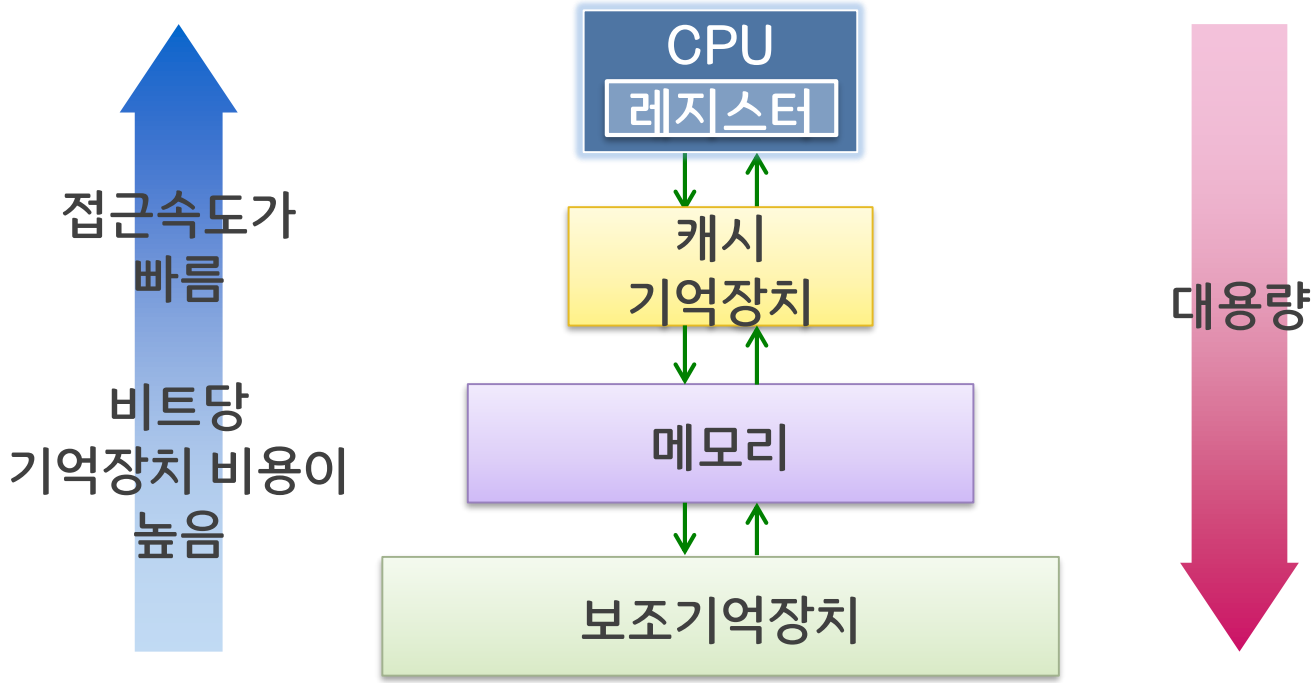
프로세스의 동작

- 프로그램 카운터를 참조하여 메모리로부터 수행될 명령을 읽어 CPU의 해당 명령을 수행



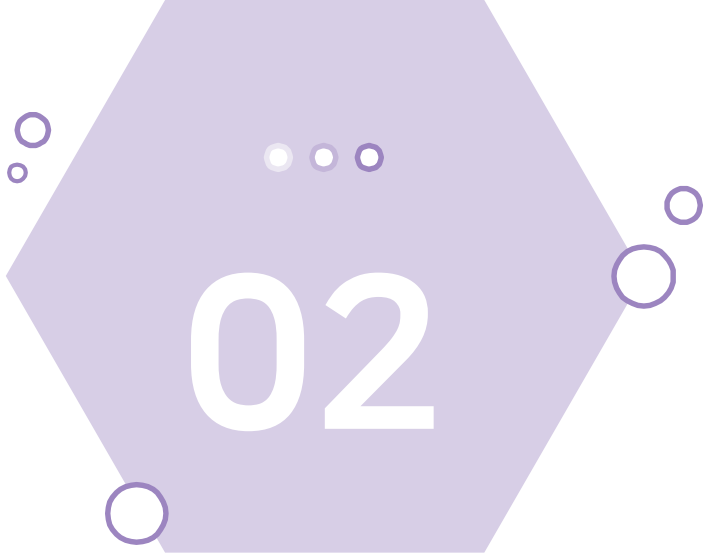
기억장치 계층구조

- 적절한 비용으로 높은 성능을 낼 수 있도록 계층적으로 구성



메모리 관리

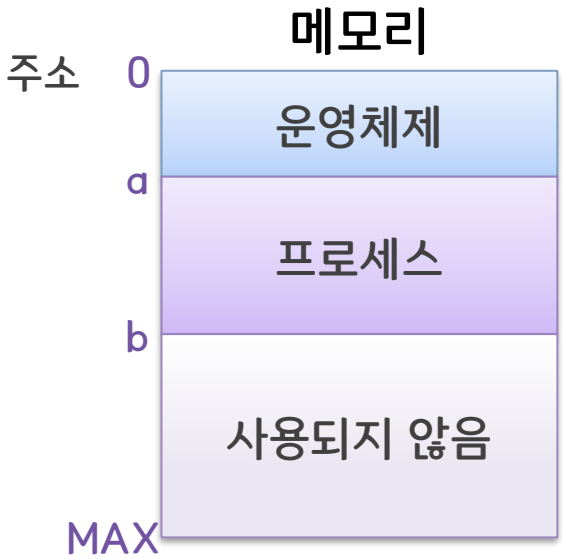
- **메모리 호출**
  - 언제 새로운 프로세스를 메모리에 둘 것인가?
- **메모리 배치**
  - 실행될 프로세스를 메모리 내의 어느 곳에 둘 것인가?
- **메모리 교체**
  - 메모리가 꽉 찬 상태에서 새로운 프로세스를 적재해야 한다면 어떤 프로세스를 제거할 것인가?
- **그 외**
  - 메모리를 고정 분할할 것인가 동적 분할할 것인가?
  - 프로세스의 적재 영역이 고정적인가 유동적인가?



단일 프로그래밍 환경

단일 프로그래밍 환경

- 초기의 시스템
  - 오직 하나의 프로세스만 메모리를 전용으로 사용
  - 프로세스는 하나의 연속된 블록으로 메모리에 할당(연속 메모리 할당)

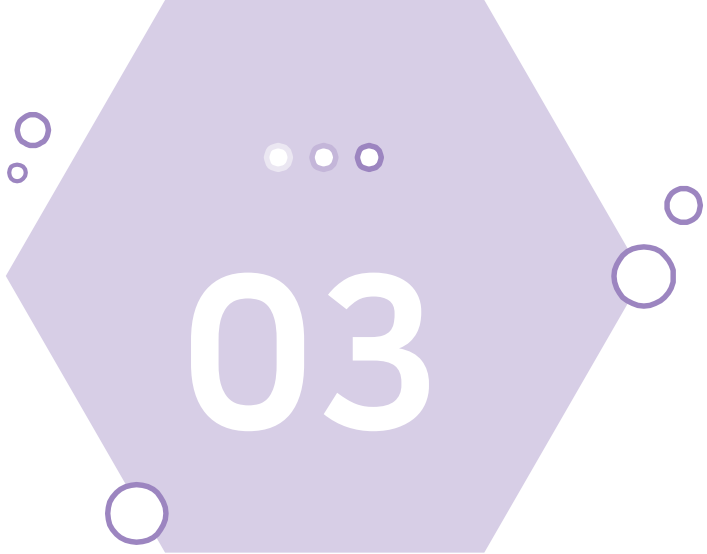


단일 프로그래밍 환경

- 문제점
  - 메모리 용량을 초과하는 프로세스는 실행 불가
  - 메모리 낭비
    - » 당장 사용되지 않는 프로세스의 영역도 계속 적재
  - 자원의 낭비

계산 위주의 사용자 프로그램

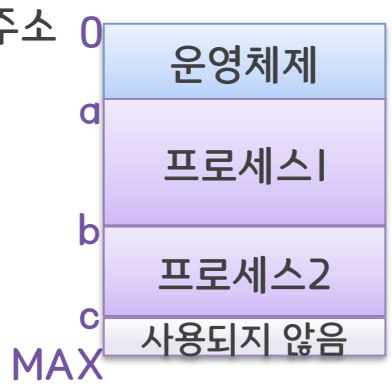
입출력 위주의 사용자 프로그램

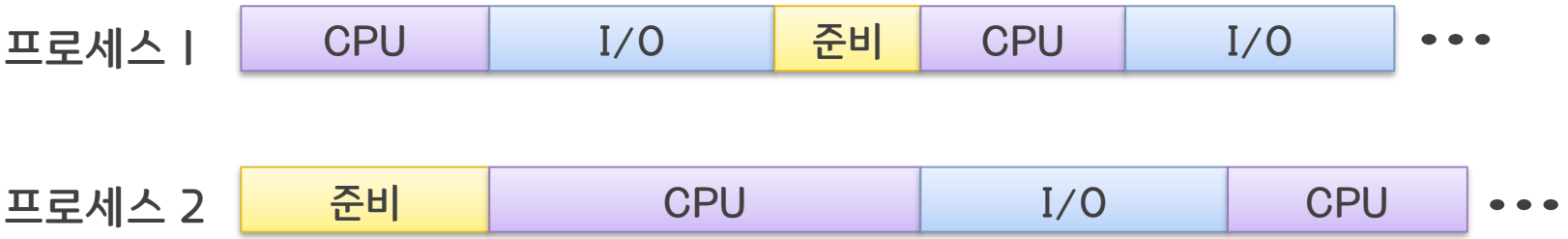


다중 프로그래밍 환경

다중 프로그래밍 환경

- 다중 프로그래밍(멀티프로그래밍)
  - 여러 개의 프로세스가 메모리에 동시에 적재되는 것
  - CPU 연산과 입출력을 동시에 함으로써 CPU 이용도와 시스템 처리량 증가





다중 프로그래밍 환경

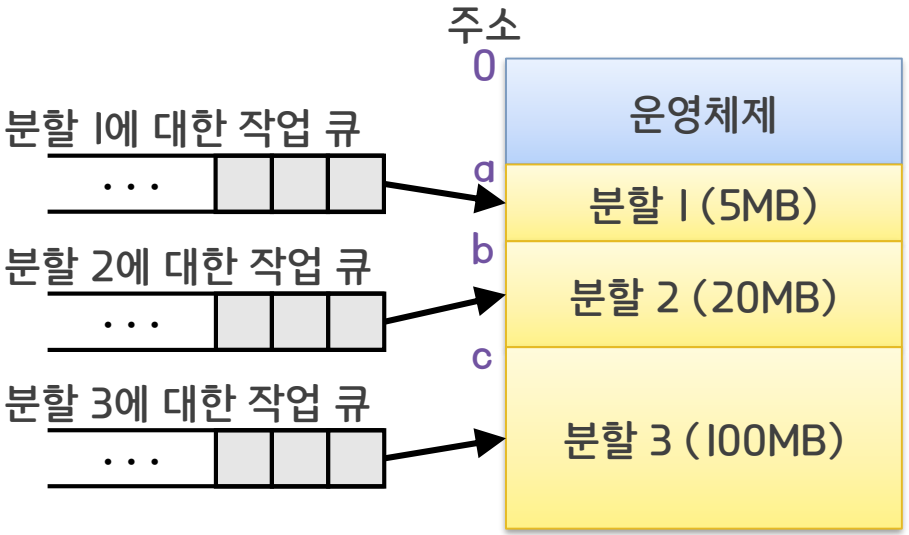
메모리 분할

- 여러 프로세스를 메모리에 적재하기 위하여 고안된 방법
- 하나의 분할에 하나의 프로세스가 적재
- 두 가지 방식: 고정 분할, 동적 분할

메모리 분할

고정 분할

- 메모리를 여러 개의 고정된 크기의 영역으로 분할
- 프로세스 배치 방법 I

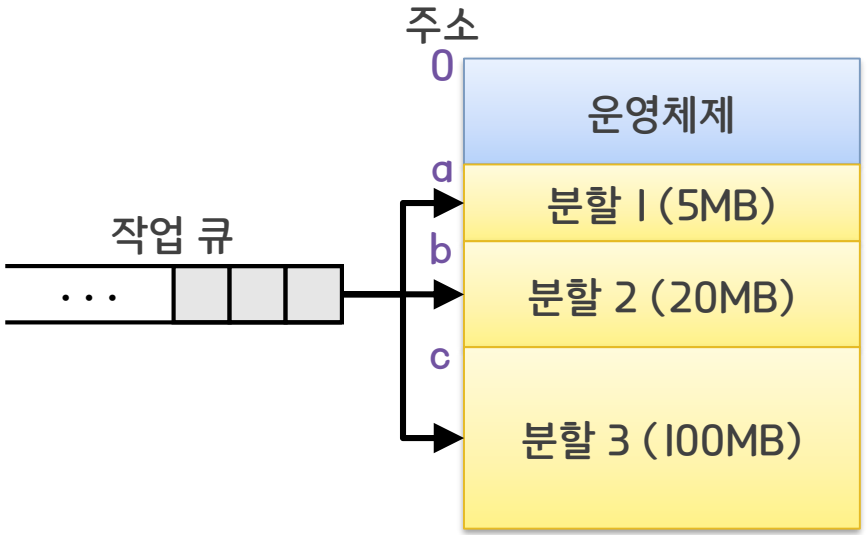


- 분할 영역마다 큐를 두고 큐에 들어온 프로세스는 해당 분할 영역에만 적재
- 절대 번역 및 적재
- 효율성 낮음

메모리 분할

고정 분할

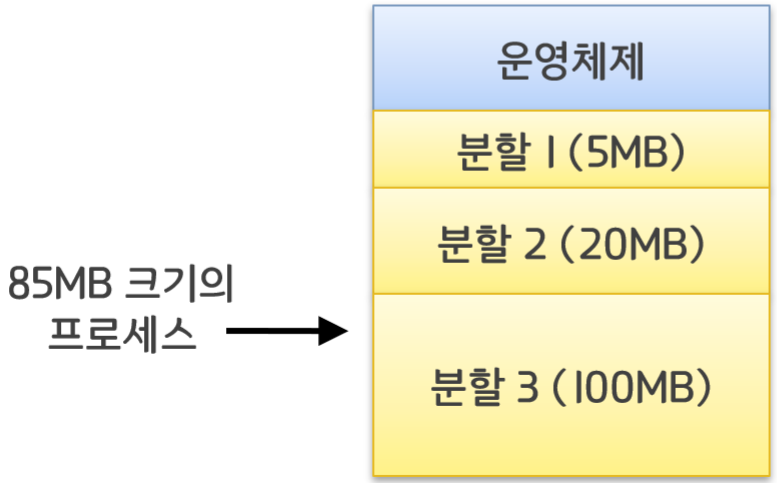
- 메모리를 여러 개의 고정된 크기의 영역으로 분할
- 프로세스 배치 방법 2



- 하나의 큐만 두고
  큐에 들어온 프로세스는
  어느 분할 영역에든 적재
- 재배치 가능 번역 및 적재
- 복잡함

메모리 분할

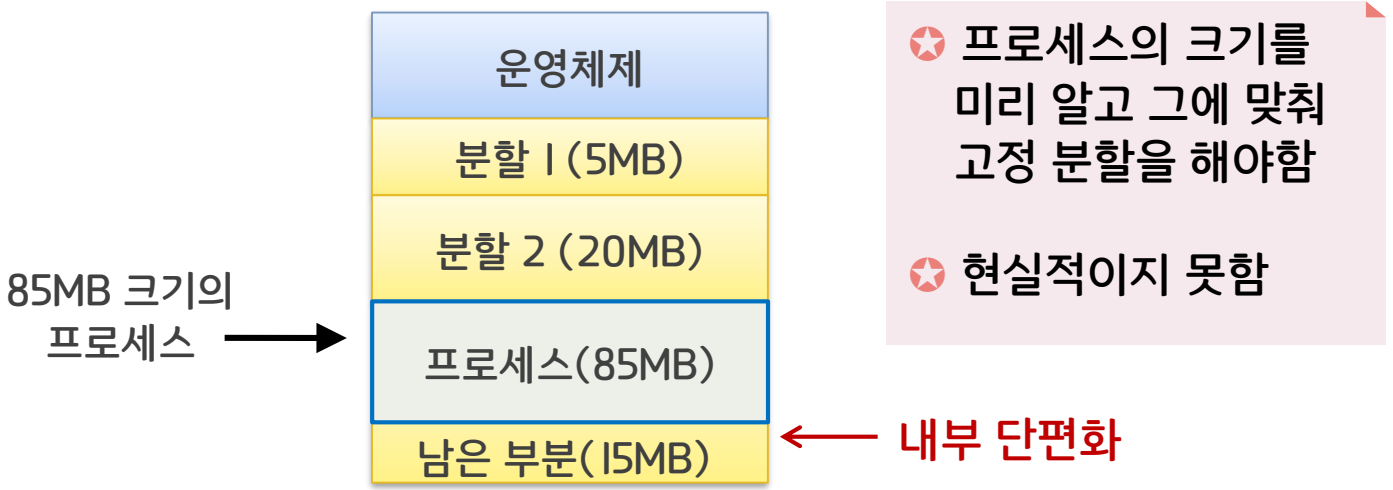
- 고정 분할
  - 문제점: 내부 단편화
    - » 프로세스의 크기가 적재된 분할 영역의 크기보다 작아서 분할 영역 내에 남게 되는 메모리는 낭비됨



메모리 분할

고정 분할

- 문제점: 내부 단편화
  - » 프로세스의 크기가 적재된 분할 영역의 크기보다 작아서 분할 영역 내에 남게 되는 메모리는 낭비됨

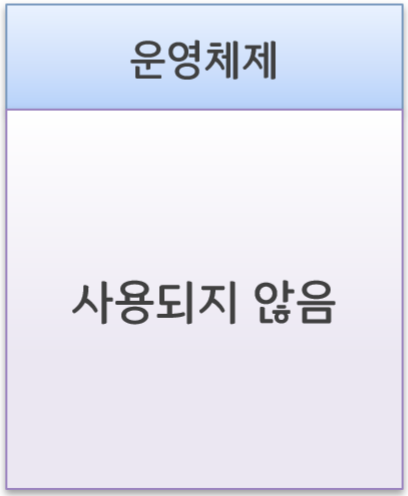


메모리 분할

동적 분할

- 메모리의 분할 경계가 고정되지 않음
- 각 프로세스에게 필요한 만큼의 메모리만을 할당

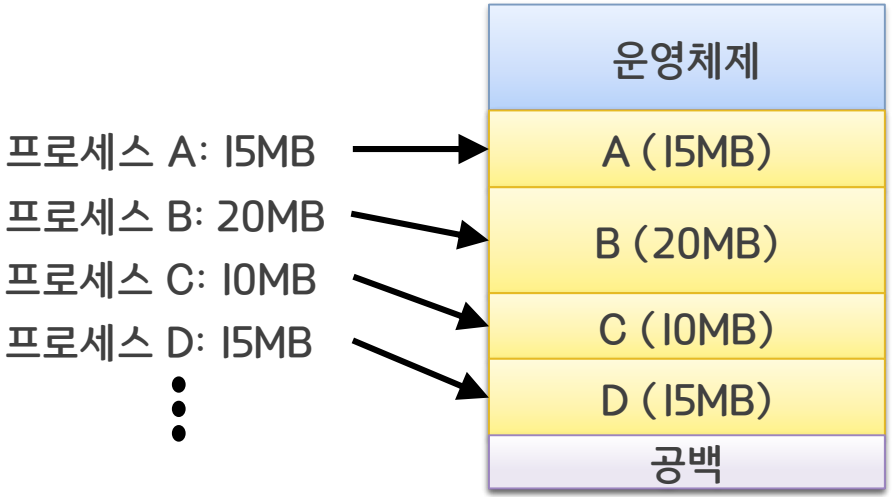
프로세스 A: 15MB
프로세스 B: 20MB
프로세스 C: 10MB
프로세스 D: 15MB
⋮



메모리 분할

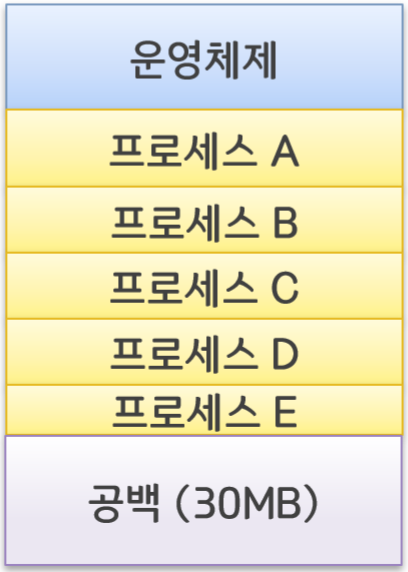
동적 분할

- 메모리의 분할 경계가 고정되지 않음
- 각 프로세스에게 필요한 만큼의 메모리만을 할당



메모리 분할

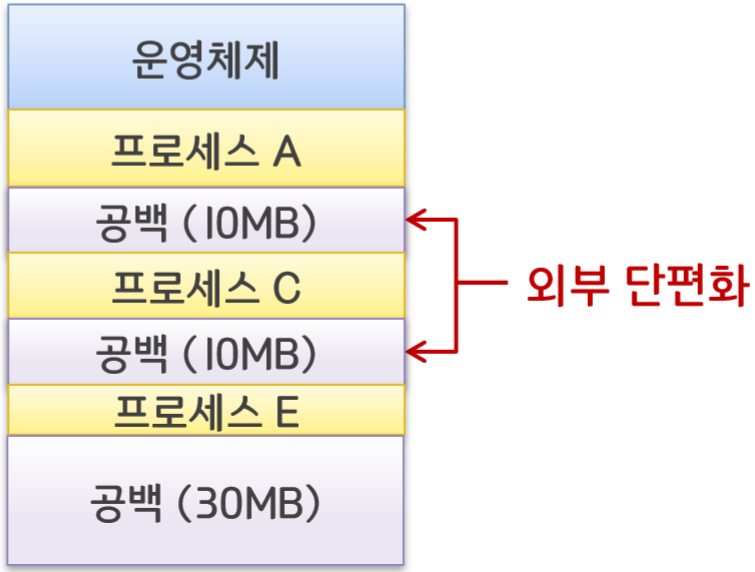
- 동적 분할
  - 문제점: 외부 단편화
    - » 메모리의 할당과 반환이 반복됨에 따라
      작은 크기의 공백이 메모리 공간에 흩어져 생김



메모리 분할

동적 분할

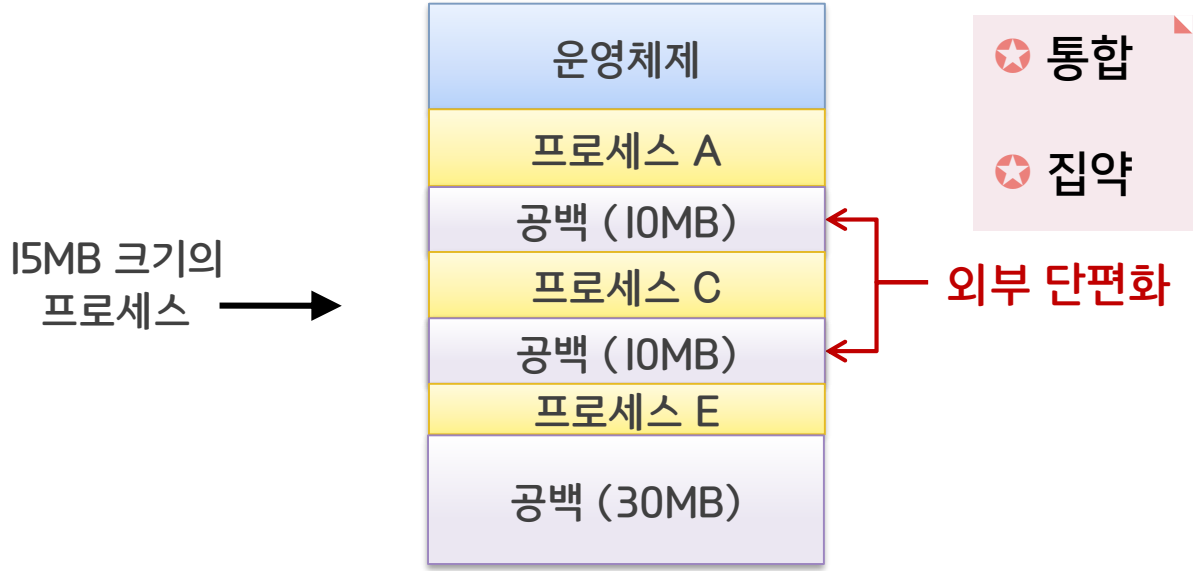
- 문제점: 외부 단편화
  - » 메모리의 할당과 반환이 반복됨에 따라
    작은 크기의 공백이 메모리 공간에 흩어져 생김



메모리 분할

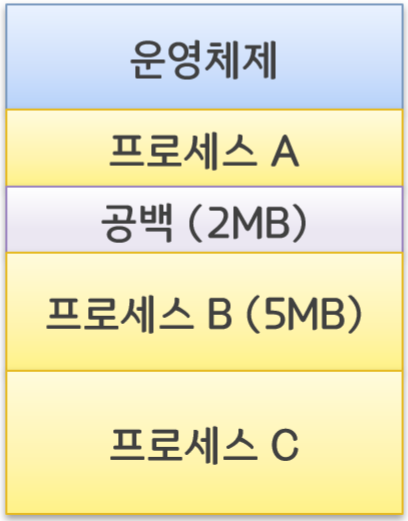
동적 분할

- 문제점: 외부 단편화
  - » 메모리의 할당과 반환이 반복됨에 따라 작은 크기의 공백이 메모리 공간에 흩어져 생김



메모리 분할

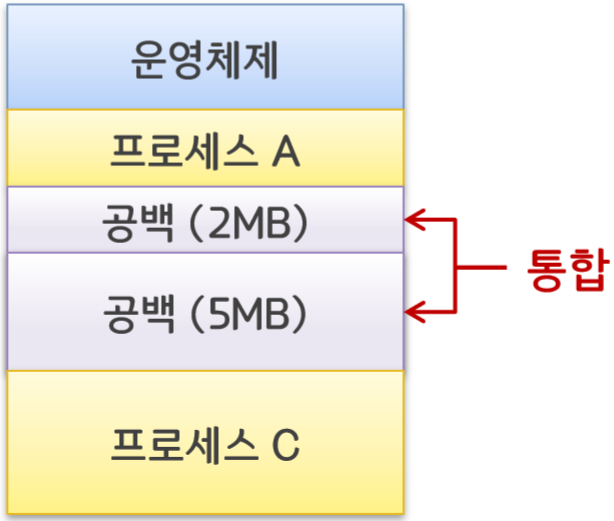
- 외부 단편화 해결 방법
  - 통합: 인접된 공백을 더 큰 하나의 공백으로 만듦



메모리 분할

외부 단편화 해결 방법

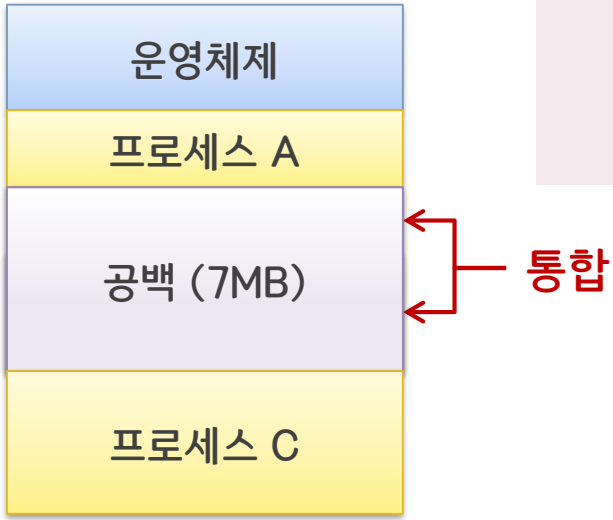
- 통합: 인접된 공백을 더 큰 하나의 공백으로 만듦



메모리 분할

외부 단편화 해결 방법

- 통합: 인접된 공백을 더 큰 하나의 공백으로 만듦



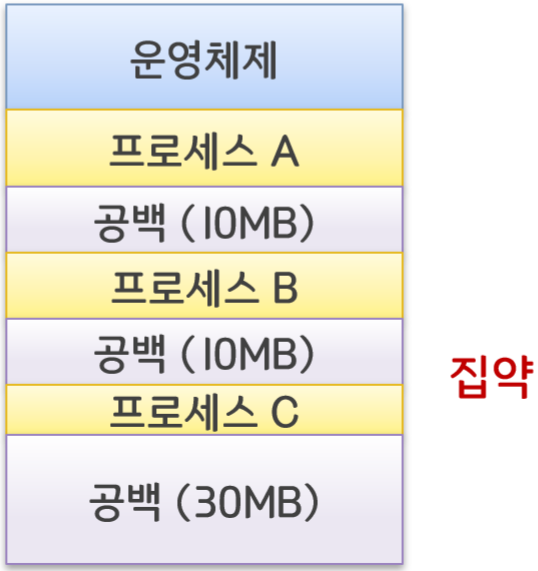
✪ 통합이 되어도 여전히 여러 공백이 메모리 내에서 여기저기 분산되어 있을 수 있음

메모리 분할

- 외부 단편화 해결 방법
  - 집약: 메모리 내의 모든 공백을 하나로 모음

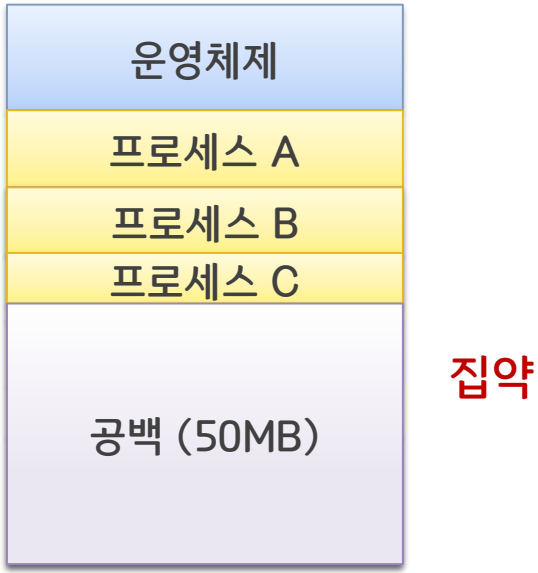
메모리 분할

- 외부 단편화 해결 방법
  - 집약: 메모리 내의 모든 공백을 하나로 모음



메모리 분할

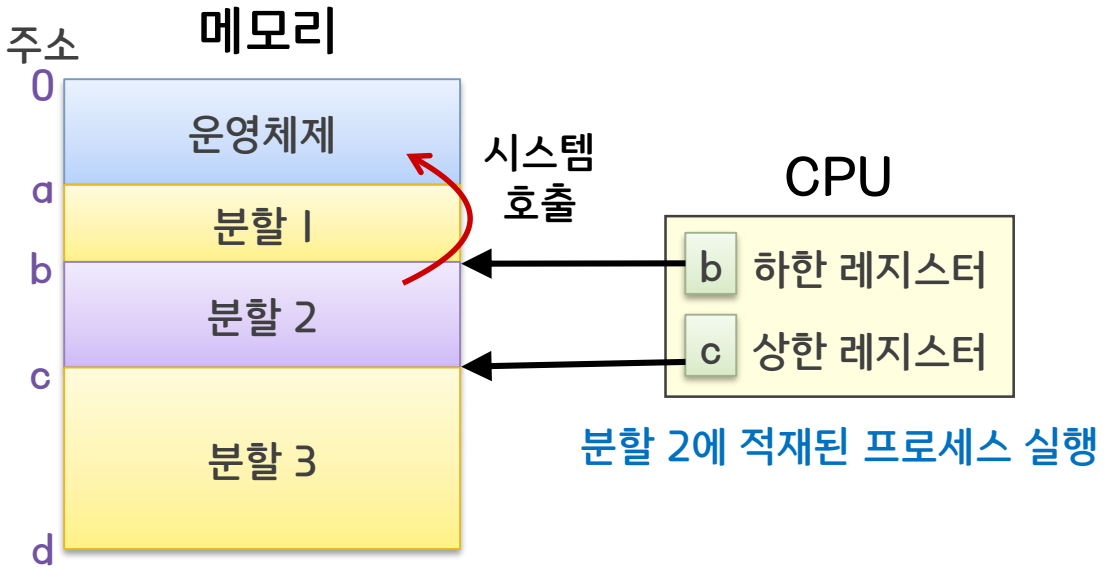
- 외부 단편화 해결 방법
  - 집약: 메모리 내의 모든 공백을 하나로 모음

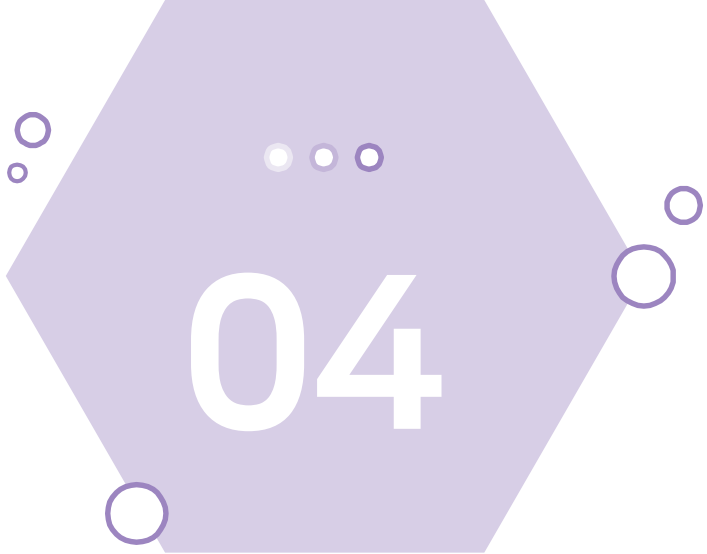


다중 프로그래밍 환경

메모리 보호

- 여러 프로세스가 동시에 메모리에 상주하므로 프로세스가 다른 할당영역을 침범하지 않게 하는 것





메모리 배치기법

메모리 배치기법

메모리 배치기법

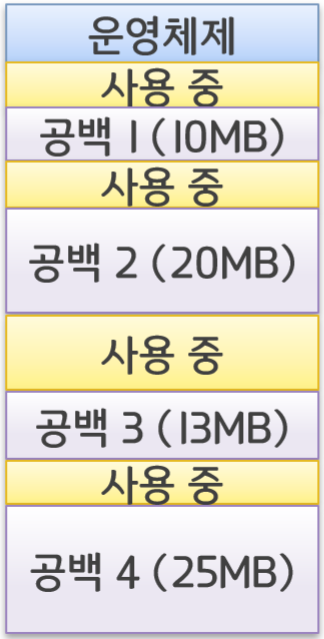
- 새로 반입된 프로그램이나 데이터를
  메모리의 어느 위치에 배치할 것인가를 결정

- 종류: 최초 적합, 후속 적합, 최적 적합, 최악 적합

메모리 배치기법

최초 적합

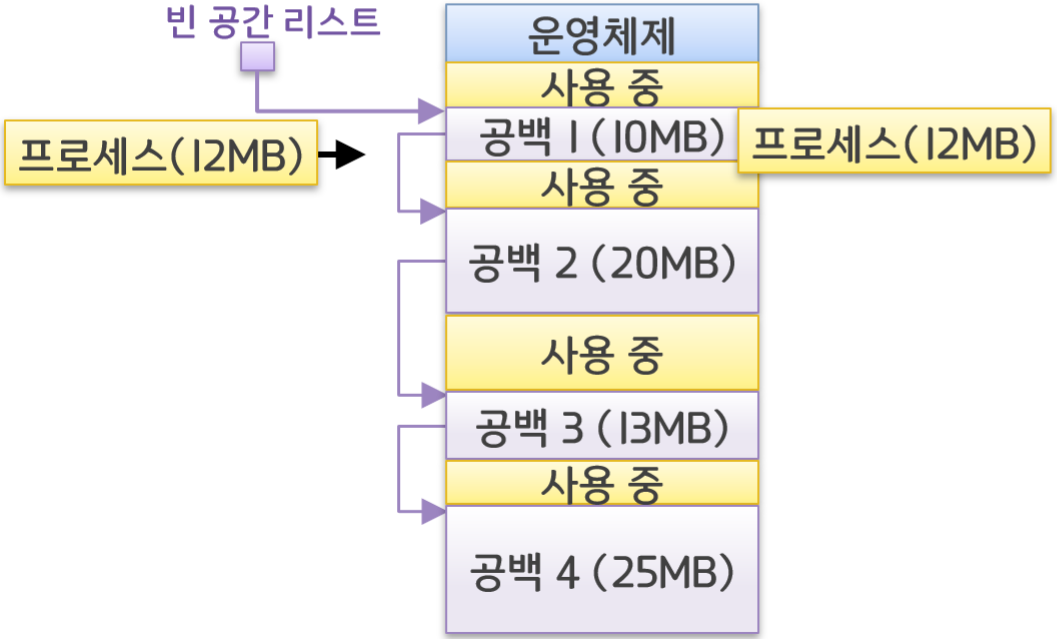
- 프로세스가 적재될 수 있는 빈 공간 중에서 가장 먼저 발견되는 곳을 할당



메모리 배치기법

최초 적합

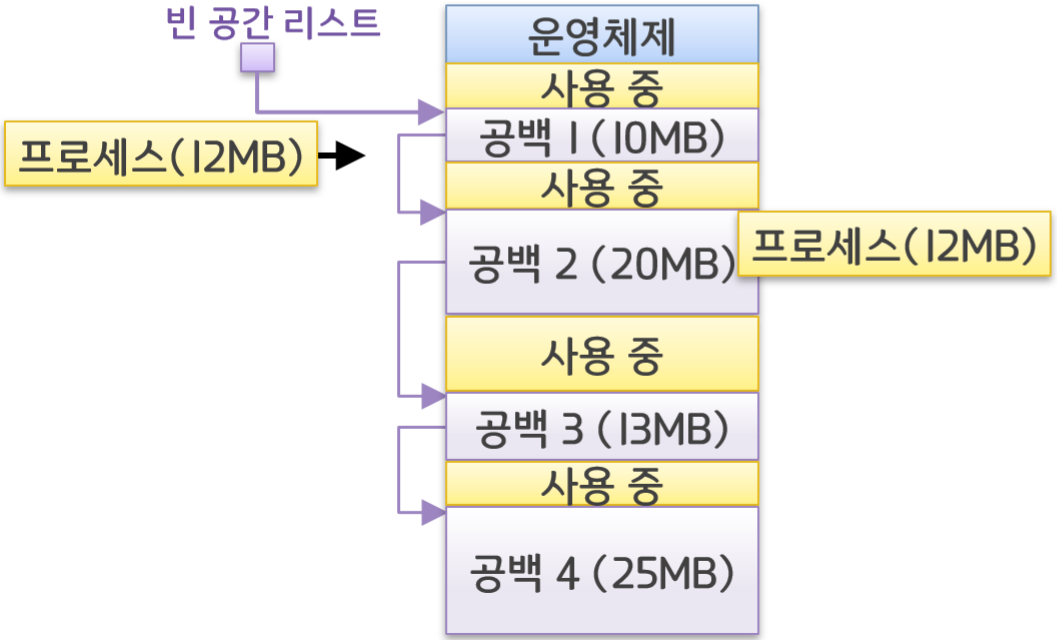
- 프로세스가 적재될 수 있는 빈 공간 중에서 가장 먼저 발견되는 곳을 할당



메모리 배치기법

최초 적합

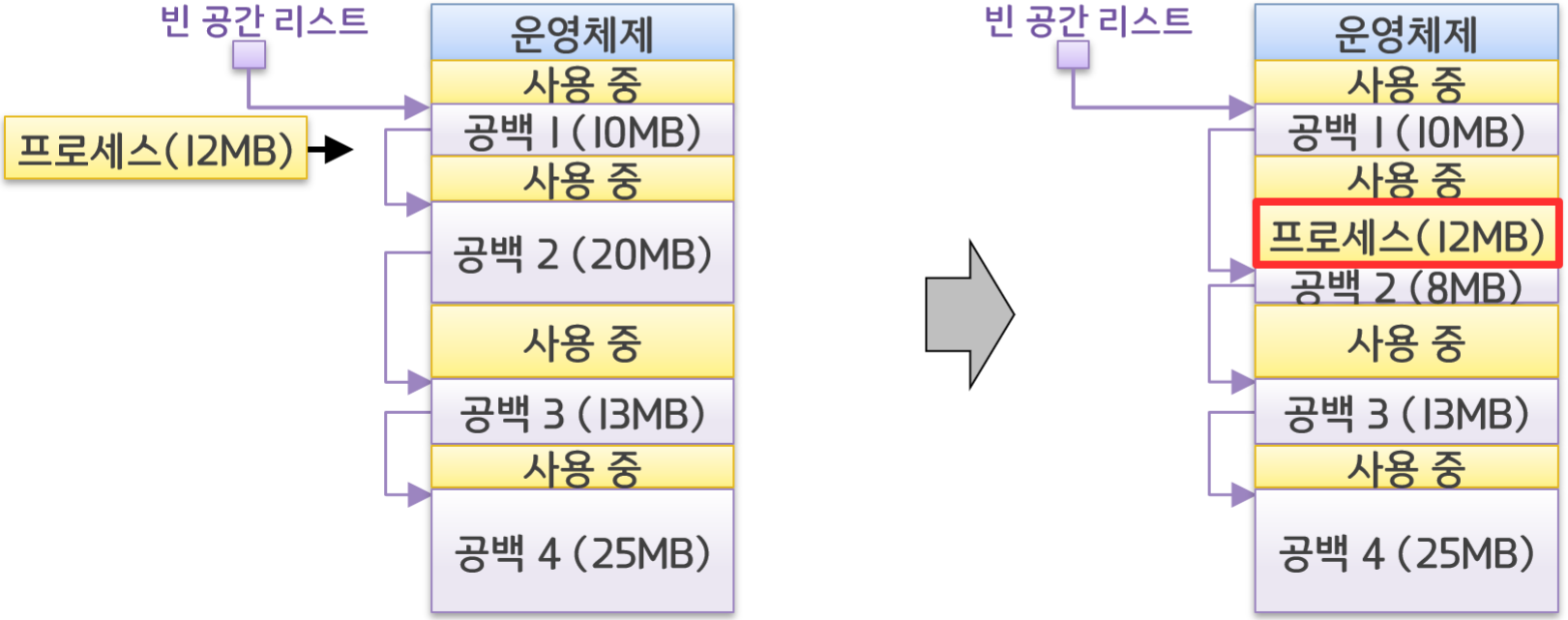
- 프로세스가 적재될 수 있는 빈 공간 중에서 가장 먼저 발견되는 곳을 할당



메모리 배치기법

최초 적합

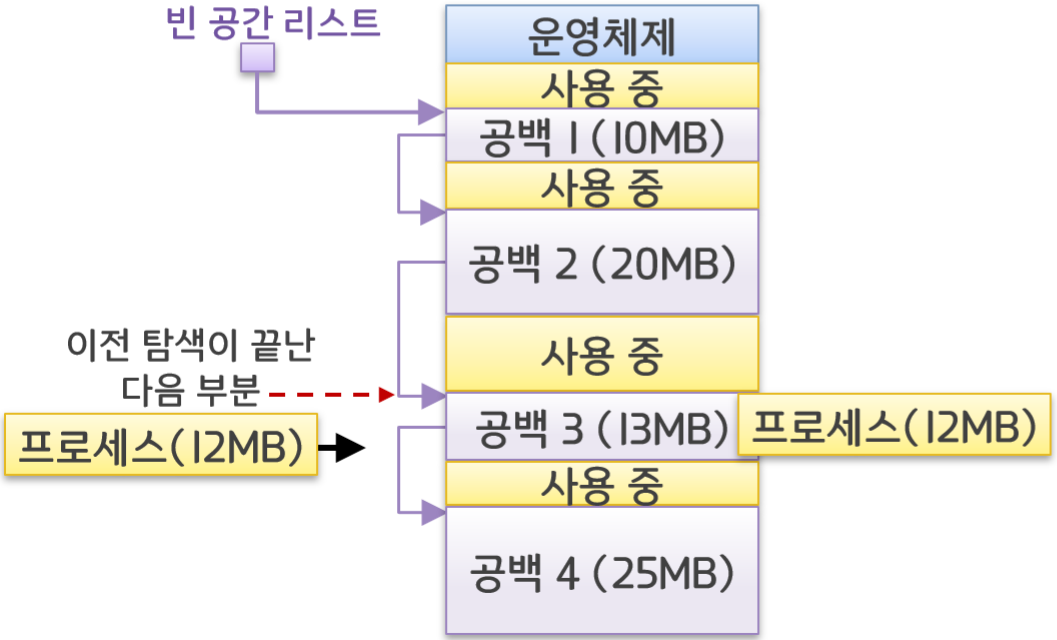
- 프로세스가 적재될 수 있는 빈 공간 중에서 가장 먼저 발견되는 곳을 할당



메모리 배치기법

후속 적합

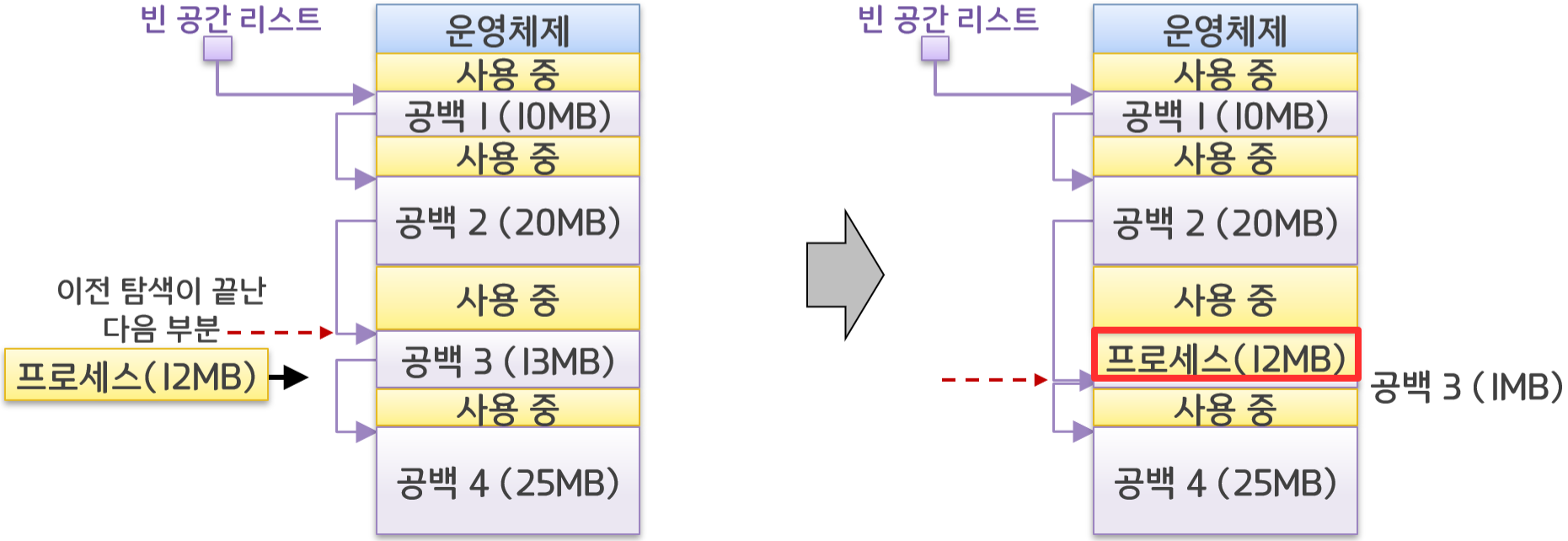
- 최초 적합의 변형으로 이전에 탐색이 끝난 그 다음 부분부터 시작



메모리 배치기법

■ 후속 적합

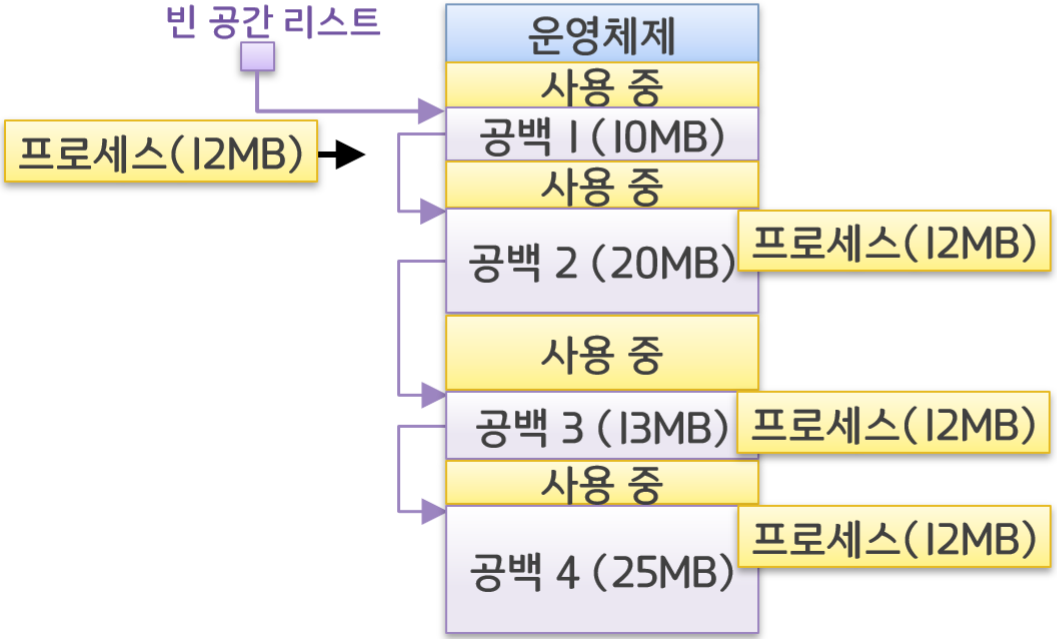
- 최초 적합의 변형으로 이전에 탐색이 끝난 그 다음 부분부터 시작



메모리 배치기법

최적 적합

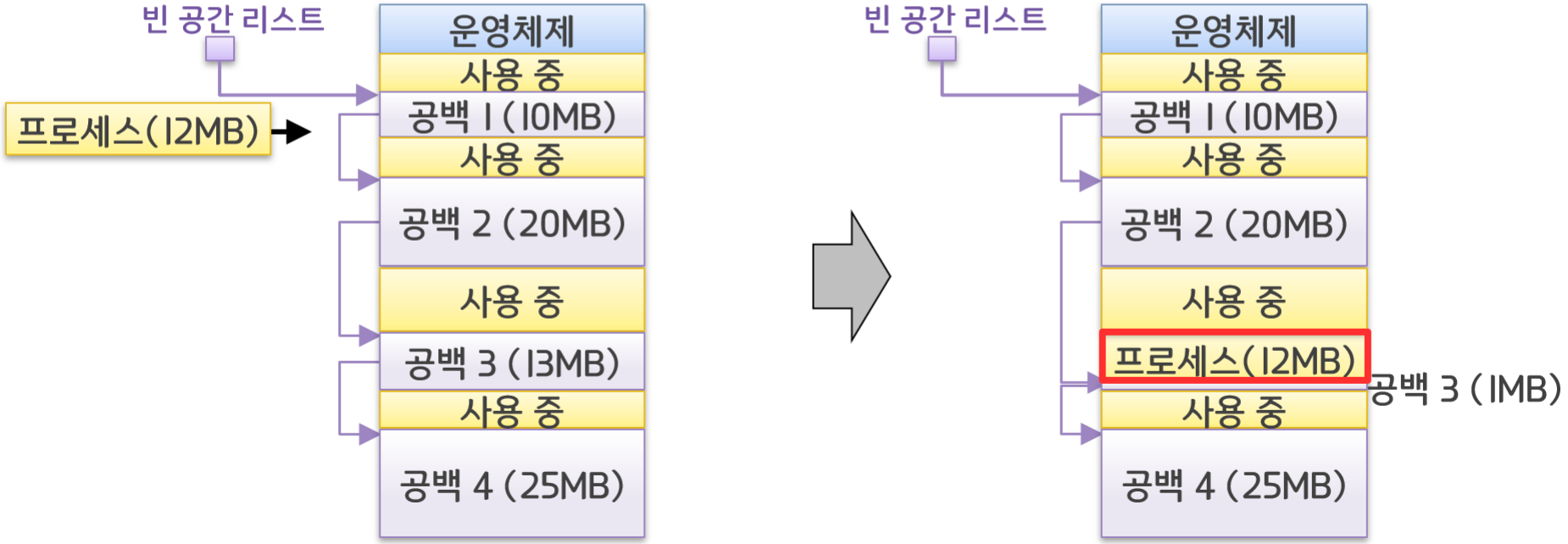
- 필요한 공간을 제공할 수 있는 빈 공간 중에서 가장 작은 곳을 선택하여 할당



메모리 배치기법

최적 적합

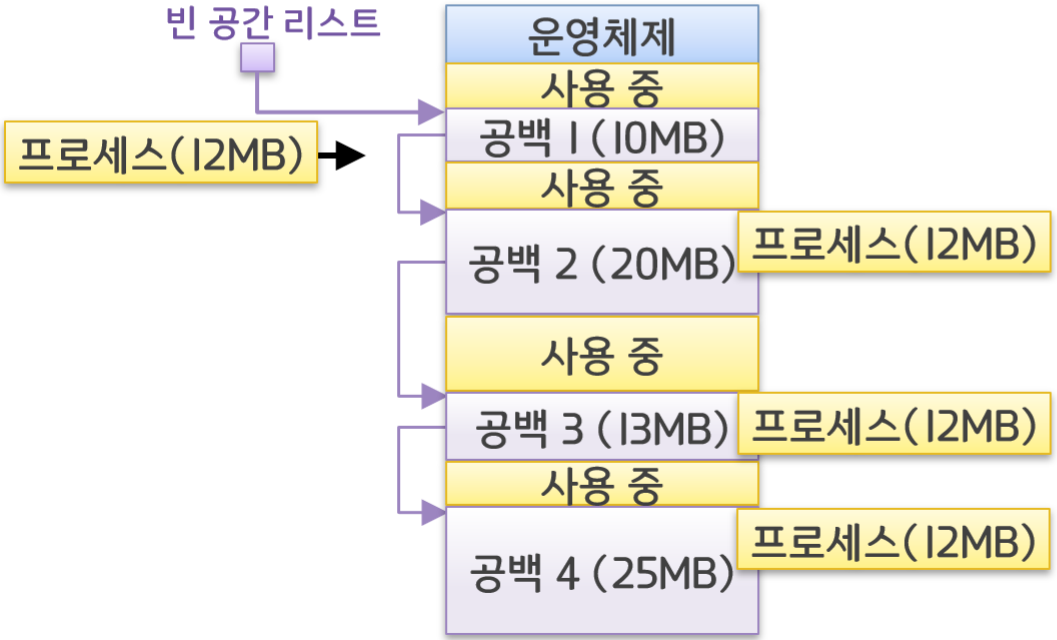
- 필요한 공간을 제공할 수 있는 빈 공간 중에서 가장 작은 곳을 선택하여 할당



메모리 배치기법

최악 적합

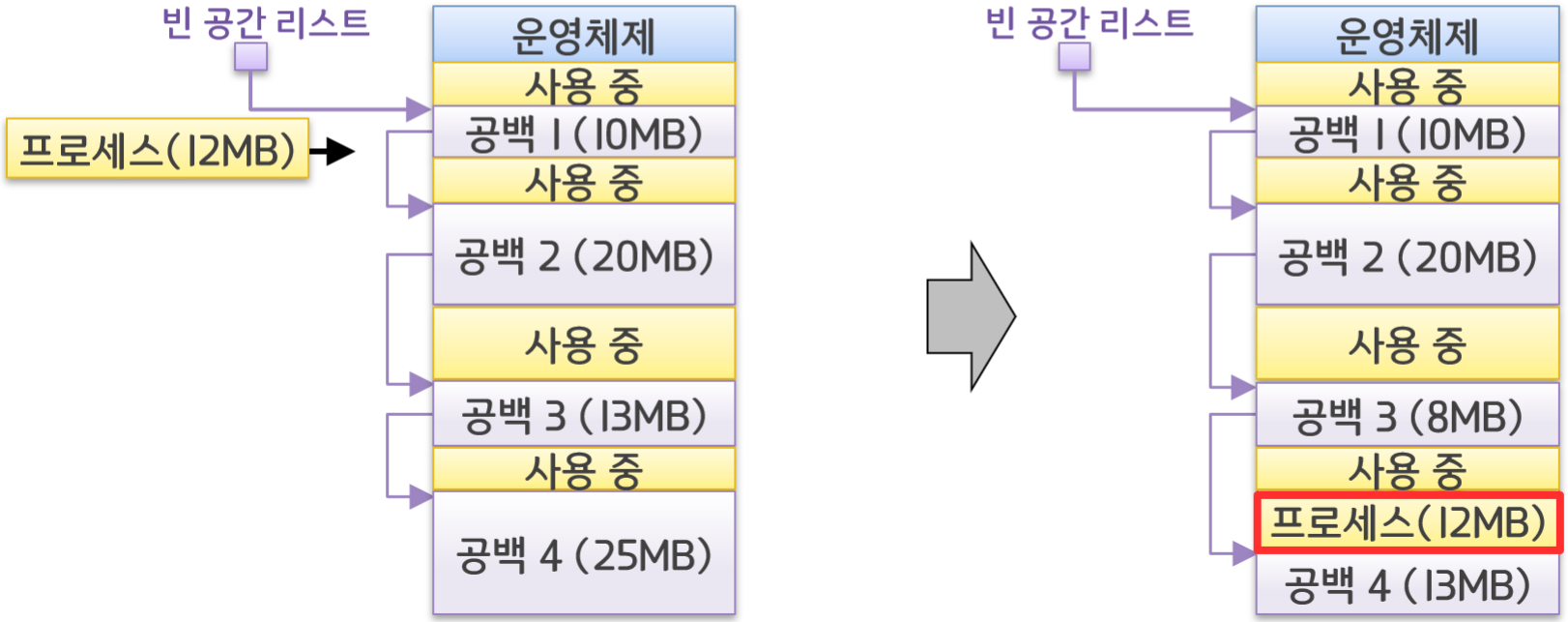
- 필요한 공간을 제공할 수 있는 빈 공간 중에서 가장 큰 곳을 선택하여 할당



메모리 배치기법

최악 적합

- 필요한 공간을 제공할 수 있는 빈 공간 중에서 가장 큰 곳을 선택하여 할당



강의를 마쳤습니다.

다음시간에는

9강. 가상 메모리 I