

05 운영체제

목차

1. 운영체제의 개요
2. 범용 운영체제
3. 프로세스 관리
4. CPU 스케줄링
5. 기억장치 관리
6. 정보 관리

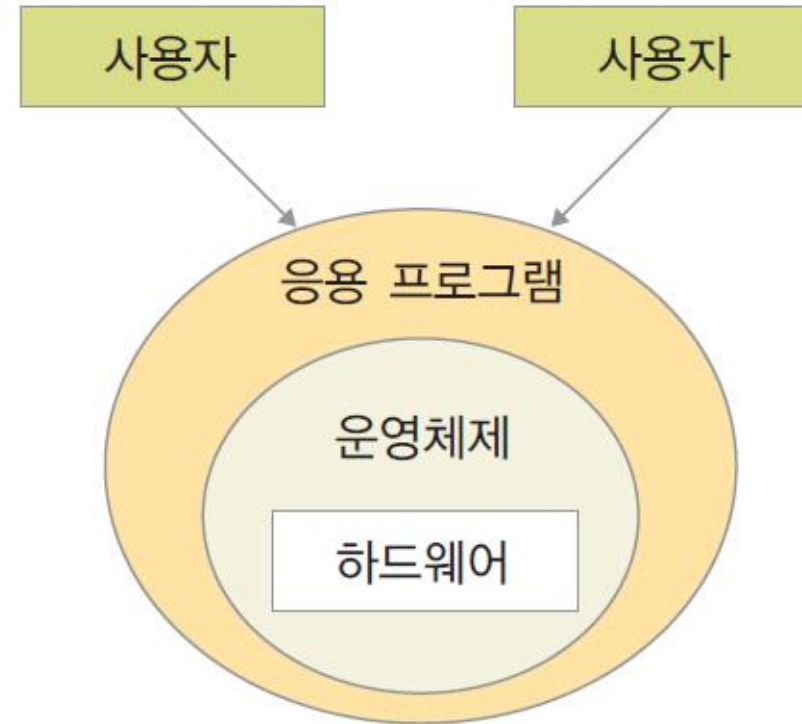
운영체제의 정의와 목적

■ 운영체제의 정의

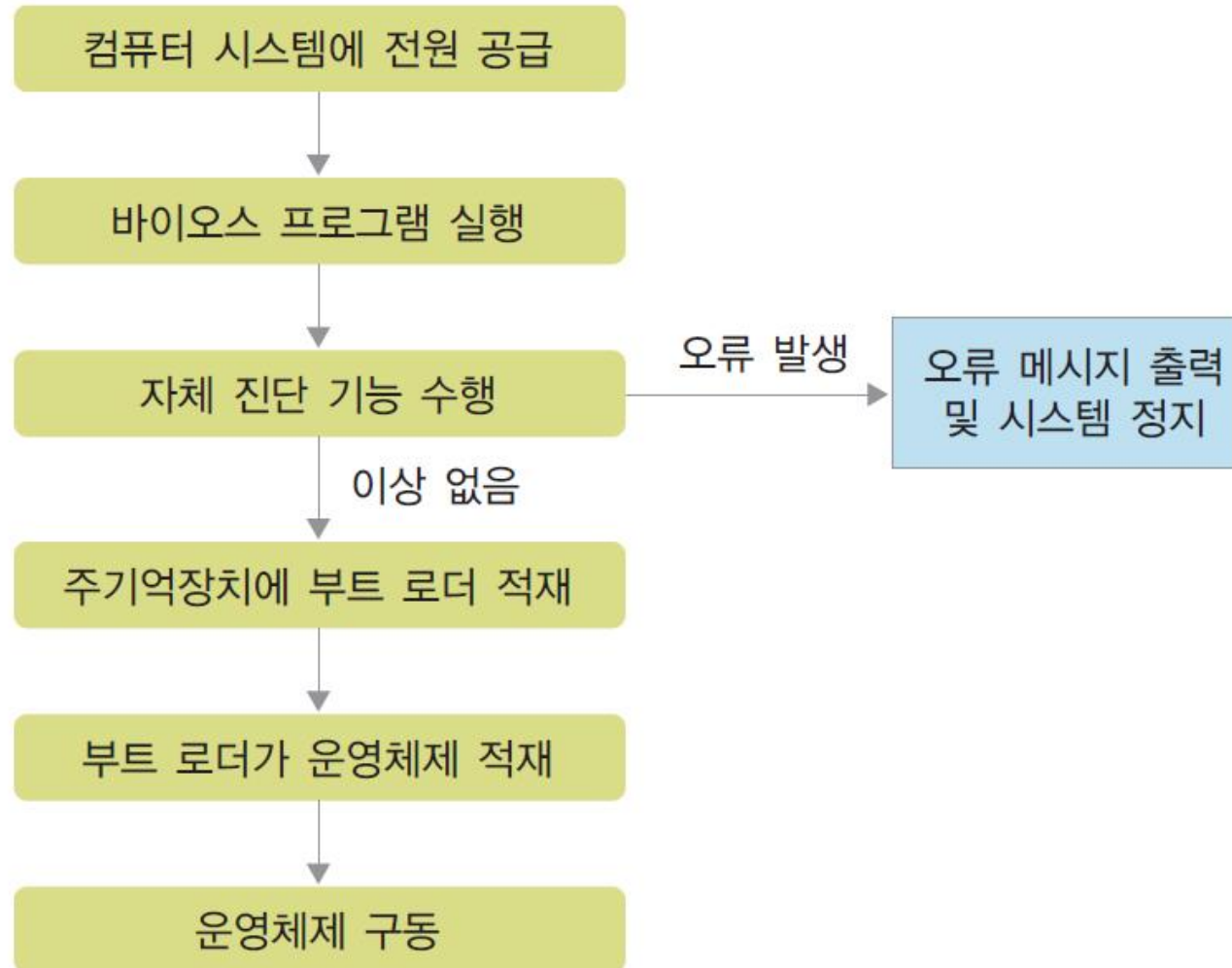
- 하드웨어와 사용자 사이에서 인터페이스를 제공하고, 한정된 시스템 자원을 더욱 효율적으로 관리하여 컴퓨터의 성능을 향상시키는 시스템 소프트웨어

■ 운영체제의 목적

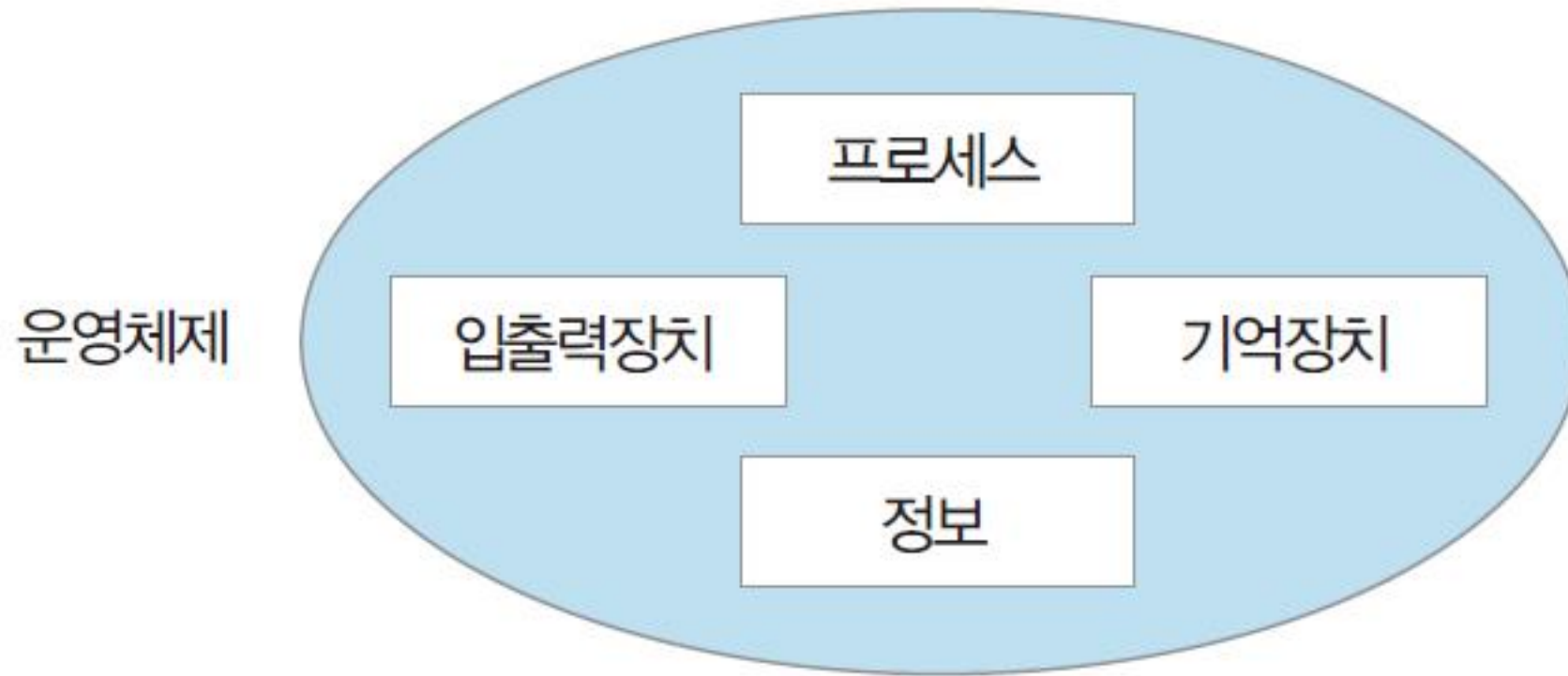
- 처리량 향상
- 응답 시간 단축
- 신뢰성 향상
- 유용성 증대



컴퓨터 시스템의 부팅 과정

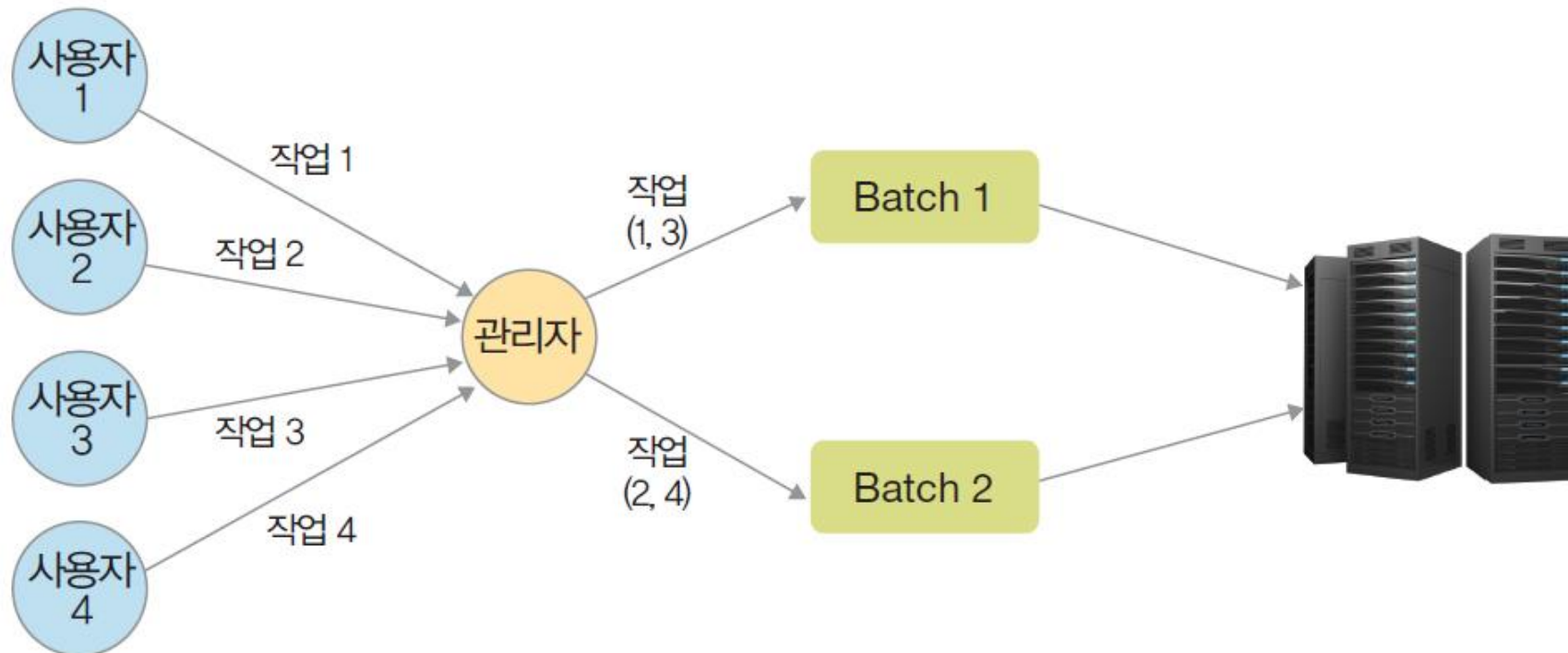


운영체제의 역할



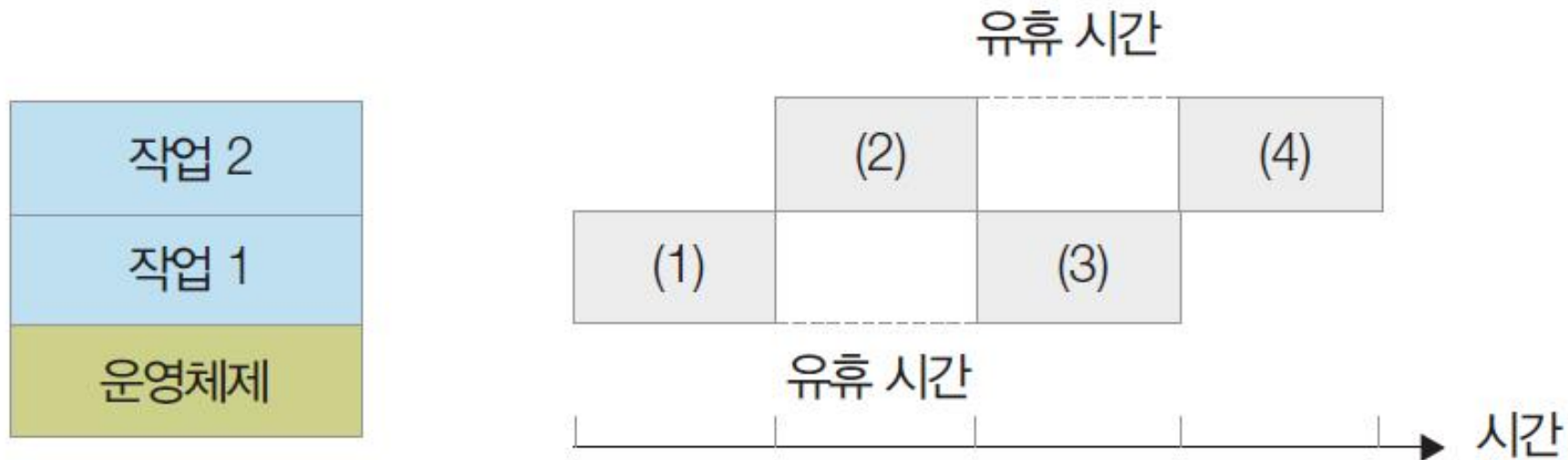
일괄 처리 시스템

- 자료가 발생할 때마다 즉시 처리하지 않고 입력되는 자료를 일정 기간 또는 일정량이 될 때까지 모아두었다가 한꺼번에 처리
- (예) 전기, 수도, 가스 요금의 계산, 수학능력시험 채점, 급여 계산 등



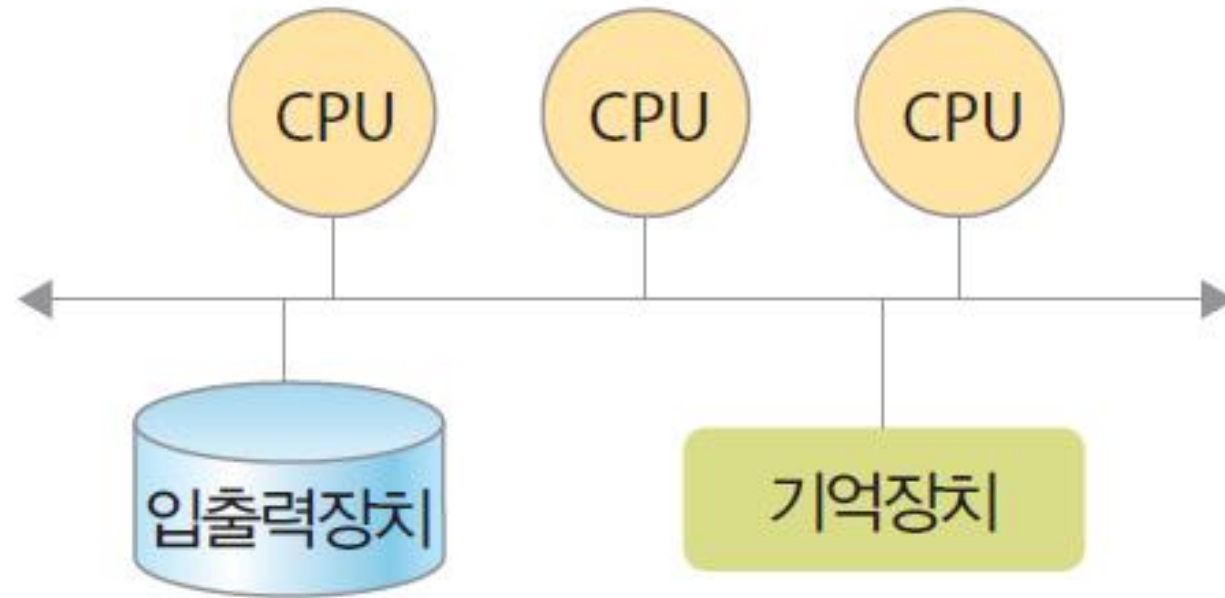
다중 프로그래밍 시스템

- 다수의 프로그램을 주기억장치에 적재시킨 후 CPU를 번갈아 가며 사용
- 하나의 프로그램이 CPU를 사용하다가 입출력연산을 수행하기 위해 입출력장치를 사용하면, 그 동안 다른 프로그램이 CPU를 사용하는 방법



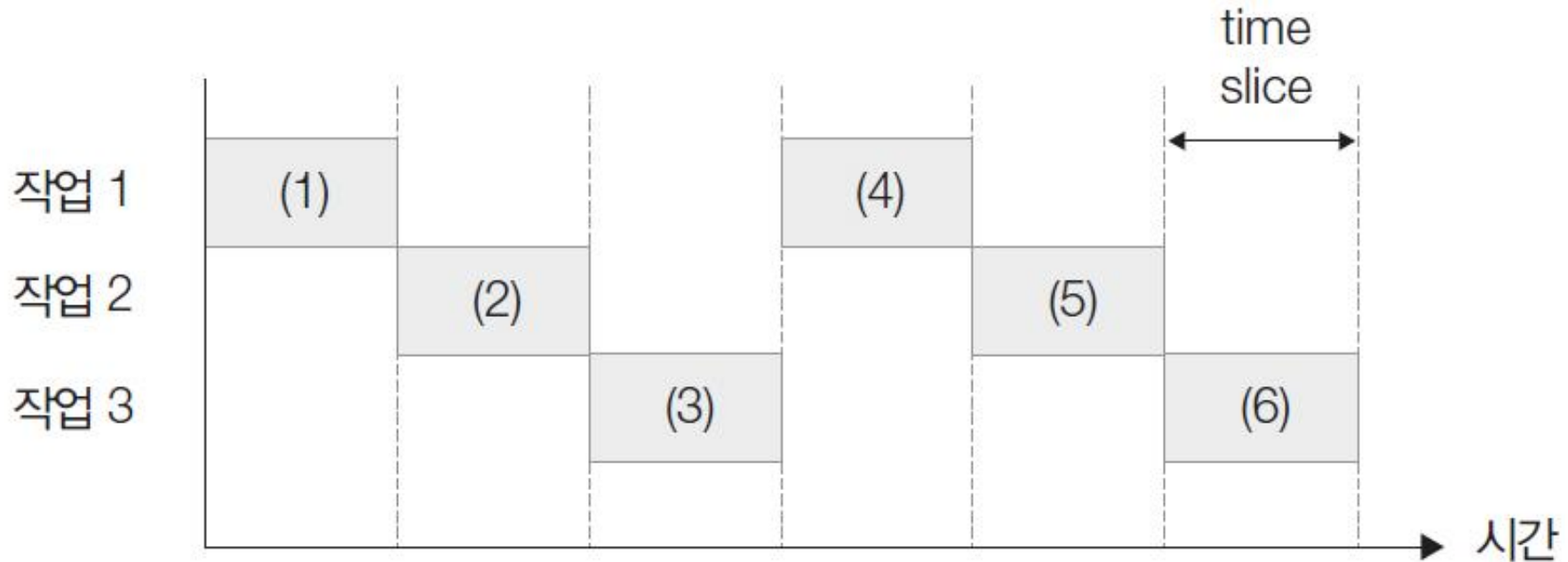
다중 처리 시스템

- 컴퓨터에 두 개 이상의 CPU를 탑재하여 동시에 처리하는 방식
- 목적 : 컴퓨터 시스템의 처리 능력 증대와 신뢰성 향상



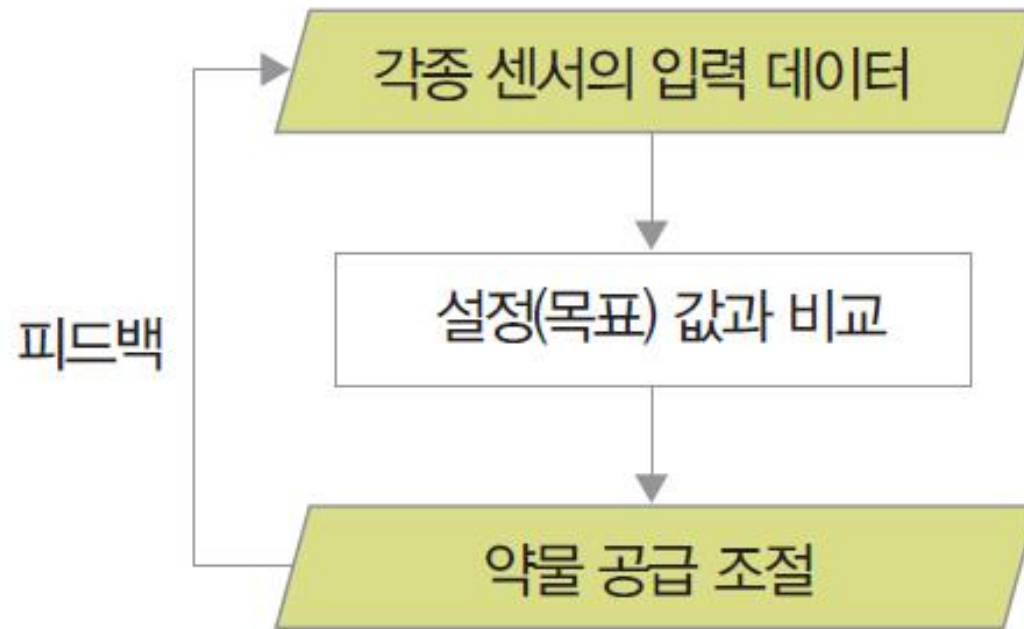
시분할 시스템

- 각 사용자에게 일정한 CPU 시간을 할당
- 주어진 시간 동안 컴퓨터와 대화 형식으로 프로그램을 실행하는 방식
- 각 사용자에게 컴퓨터 시스템을 독점적으로 사용하고 있다는 느낌을 줌



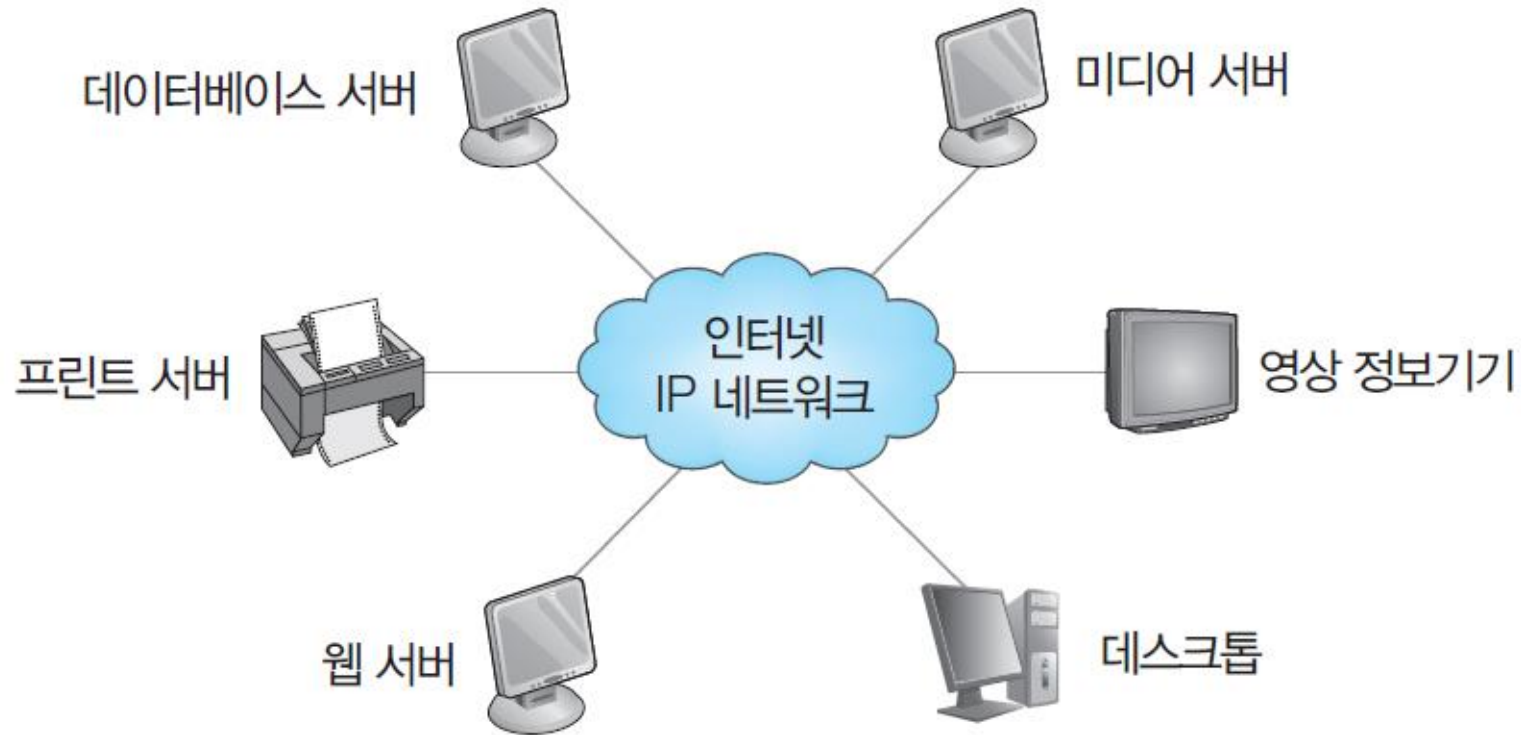
실시간 시스템

- 입력된 데이터를 즉시 처리한 후 바로 결과를 보냄
- 작업의 요청에서부터 수행 결과를 얻기까지 시간제약이 존재
- 제약의 엄격함에 따라 경성 실시간 시스템과 연성 실시간 시스템으로 구분



분산 처리 시스템

- 네트워크를 통해 연결된 다수의 컴퓨터 시스템에 작업과 자원을 분산시켜 처리
- 설계 목적은 자원 공유, 연산 속도와 신뢰성 향상, 통신 기능 활용



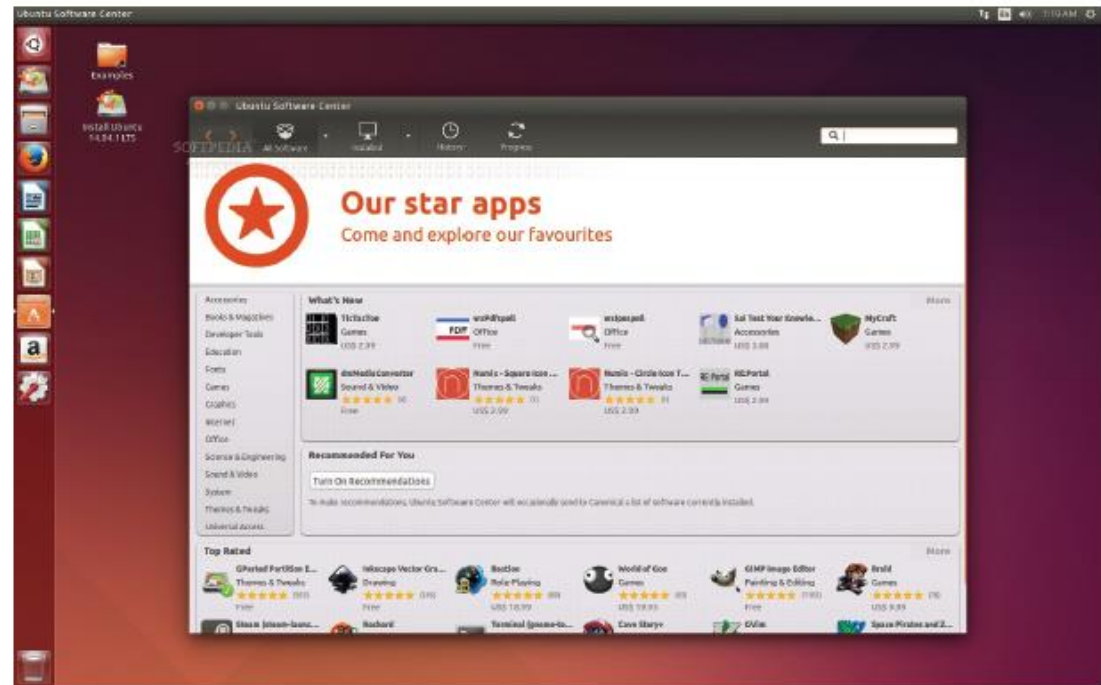
유닉스

- 1969년 미국 벨 연구소의 켄 톰슨에 의해 PDP-7 시스템의 운영체제로 처음 개발
- 프로그램의 소규모 모임으로 시작
- 현재는 소규모 내장 컨트롤러에서 메인 프레임 컴퓨터까지 다양한 영역에서 수많은 응용 프로그램을 구동시키는 개방형 표준 운영체제로 발전



리눅스

- 1991년 핀란드 헬싱키 대학의 학생 리누스 토발즈에 의해 유닉스 호환 운영체제로 개발
- 오픈 소스 정책을 따름



윈도우

- 애플의 그래픽 사용자 인터페이스 운영체제인 맥 OS에 대항하기 위해 개발한 운영체제
- 자사의 MS-DOS에 다중 작업 기능과 GUI 환경을 추가



윈도우 버전별 특징

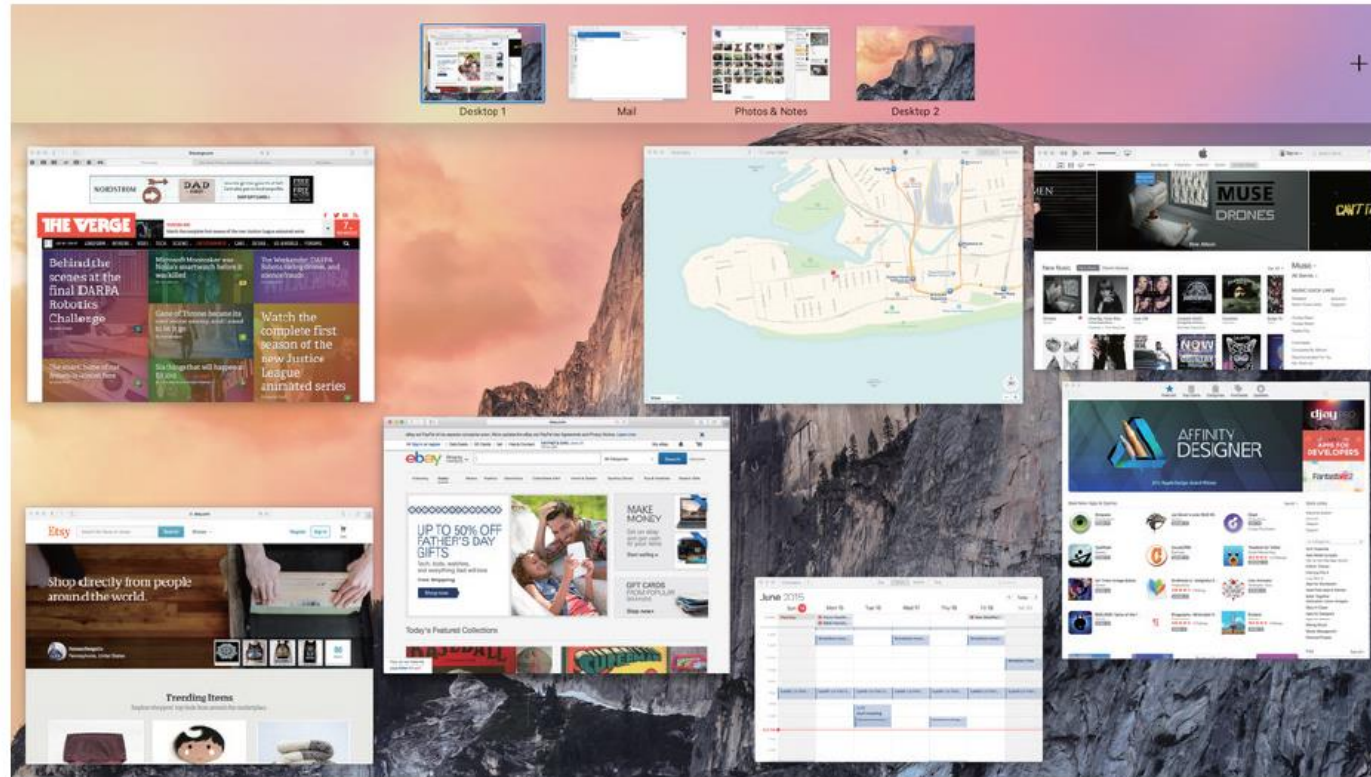
버전 명	출시 연도	특징
윈도우 1.0	1985	· MS-DOS의 확장판
윈도우 3.0/3.1	1990/1992	· 사용자 인터페이스 개선 · 강력한 다중 작업 기능 제공
윈도우 NT	1993	· NT 커널 기반의 전문가용 운영체제 · NTFS 파일 시스템, 강력한 보안 기능, 네트워킹 모듈 제공
윈도우 95	1995	· 진정한 의미의 32비트 운영체제
윈도우 98	1998	· FAT32 파일 시스템 지원 · 개선된 PnP(Plug and Play) 기능 제공 · USB, IEEE1394, AGP(Accelerated Graphics Port) 등 지원 · 다중 모니터 및 표준 전원 관리 기능(ACPI) 제공
윈도우 me	2000	· UPnP(Universal PnP)와 시스템 복원 기능 추가 · 9x 기반의 개인용 윈도우의 마지막 버전
윈도우 2000	2000	· 기업/전문가용 운영체제

윈도우 버전별 특징

윈도우 XP	2001	· 소비자 지향형 운영체제(NT 커널 기반+윈도우 me 편의성) · 홈 에디션과 프로페셔널 버전으로 구분
윈도우 XP 미디어 센터	2003	· DVD/TV 기능(프로그램 녹화와 원격 조정 기능 지원)
윈도우 서버 2003	2003	· 윈도우 2000 서버 제품군에 새로운 기능과 강력한 보안 기능 추가
윈도우 비스타	2006/2007	· 기업용 버전과 개인용 버전을 별도로 출시 · 윈도우 XP보다 강화된 보안 기능 제공
윈도우 서버 2008	2008	· 윈도우 서버 2003의 후속작(윈도우 비스타 기반)
윈도우 7	2009	· 다양한 에디션 및 32/64비트 버전 출시 · 윈도우 XP와 비스타 기반의 프로그램 실행 가능
윈도우 서버 2008 R2	2009	· 윈도우 서버 2008의 후속작(윈도우 7 기반)
윈도우 8/8.1	2012/2013	· 인텔/AMD 외에도 ARM 프로세서 지원(윈도우 RT) · 터치스크린 기능 탑재
윈도우 서버 2012	2012	· 윈도우 서버 2008 R2의 후속작(윈도우 8 기반) · 네 가지 에디션(파운데이션/에센셜/스탠다드/데이터 센터)
윈도우 서버 2012 R2	2013	· 윈도우 서버 2012의 후속작(윈도우 8.1 기반)

맥 OS

- 애플의 개인용 컴퓨터인 매킨토시용으로 개발된 GUI 운영체제
- 애플의 자체 코드에 기반을 둔 클래식 시리즈와 BSD 코드를 사용하는 유닉스 계열 운영체제인 NEXTSTEP 기반의 맥 OS X로 나눔

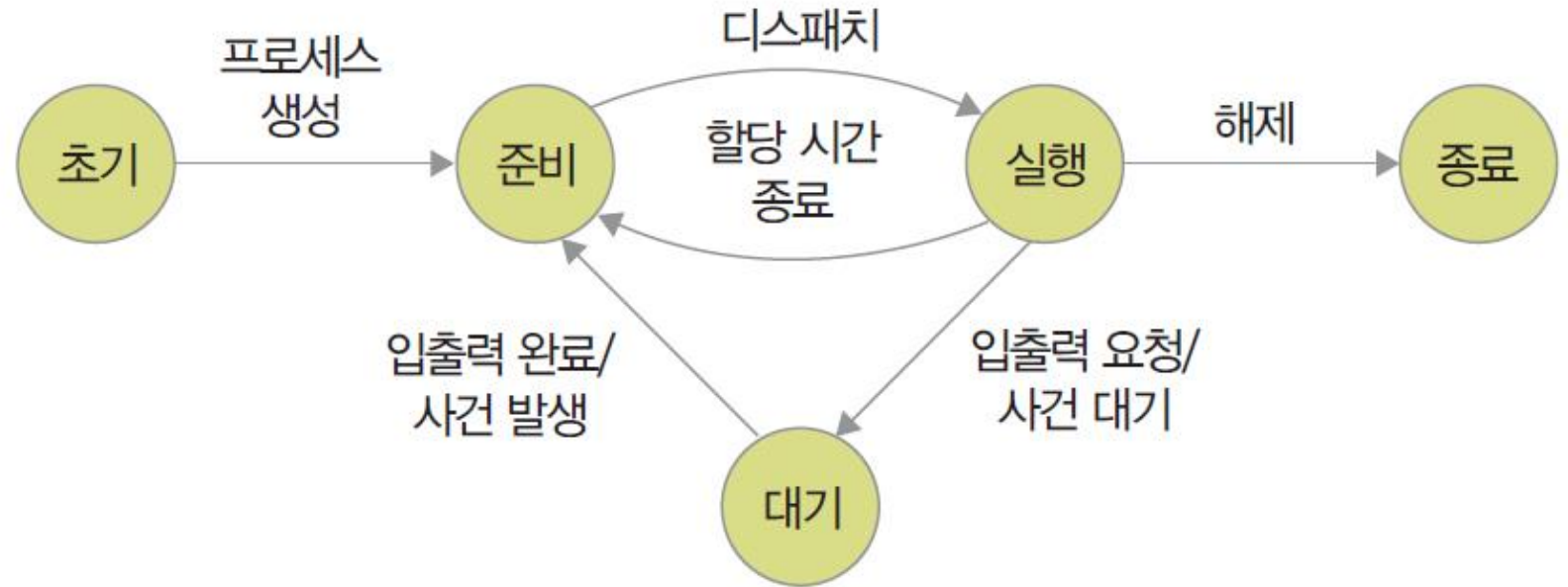


프로세스의 개념

- 실행중인 프로그램
- 프로그램은 디스크에 저장된 파일과 같은 수동적인 개체
- 프로세스는 프로그램 카운터(PC)를 가진 능동적인 개체
- 프로그램이 주기억장치에 적재되어 처리를 시작할 때 비로소 프로세스 라는 활동적인 개체가 됨

프로세스 상태

- 초기 상태(new state)
- 준비 상태(ready state)
- 실행 상태(running state)
- 대기 상태(blocked state)
- 종료 상태(exit state)



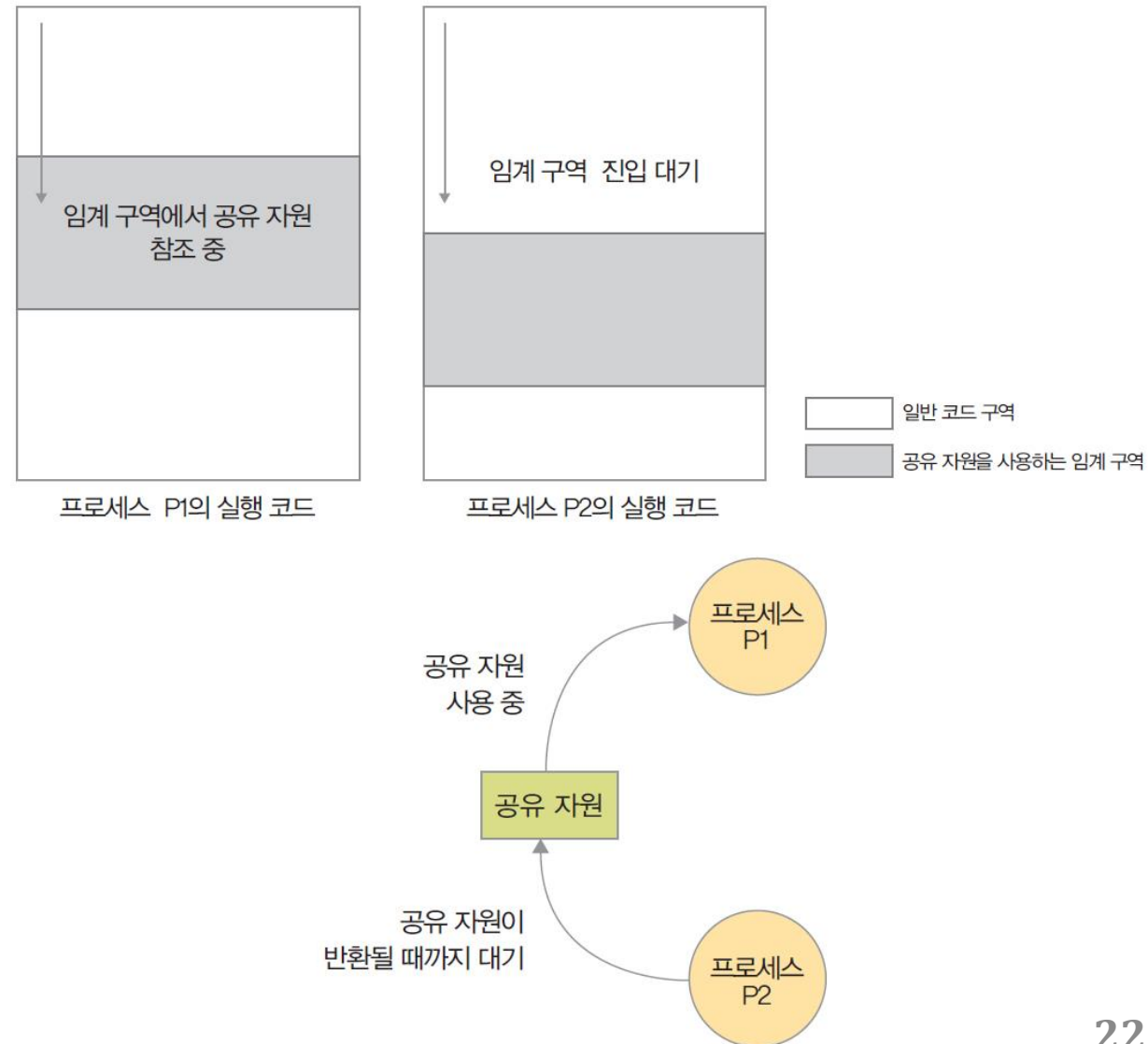
프로세스 제어 블록

- 프로세스에 대한 정보를 운영체제에 제공하는 자료 구조
- 프로세스가 생성될 때 만들어졌다가 프로세스가 실행을 마치면 삭제됨

포인터	프로세스 상태
프로세스 식별자	
프로그램 카운터	
레지스터	
기억장치 관리 정보	
개방된 파일 리스트	
기타 정보	

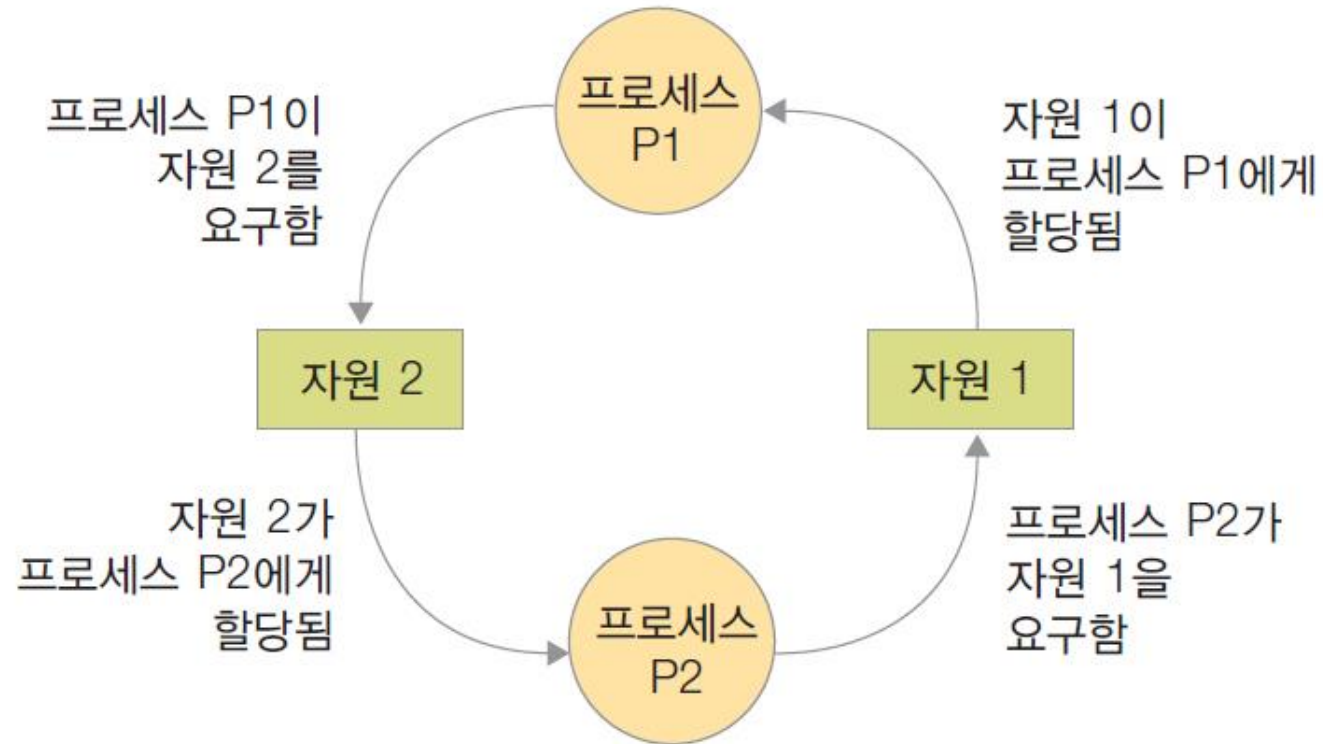
병행 프로세스

- 두 개 이상의 연관된 프로세스가 동시에 실행되는 것
- 두 개 이상의 프로세스들이 병행 처리 상태에 있으면 예측 불가능한 결과가 발생하는데, 이러한 오류를 막기 위해 다음과 같은 기법을 사용
 - 동기화(synchronization)
 - 임계 구역(critical section 또는 critical region)
 - 상호 배제(mutual exclusion)



교착 상태

- 다중 프로그래밍 시스템에서 하나 이상의 프로세스가 절대 일어나지 않을 특정 사건을 기다리고 있는 상태



교착 상태

- 교착 상태의 발생 조건
 - 상호 배제(mutual exclusion)
 - 보유와 대기(hold and wait)
 - 비선점(nonpreemption)
 - 환형 대기(circular wait)
- 교착 상태의 해결 방안
 - 교착 상태 예방(deadlock prevention)
 - 교착 상태 회피(deadlock avoidance)
 - 교착 상태 탐지(deadlock detection)
 - 교착 상태 회복(deadlock recovery)

CPU 스케줄링의 목적

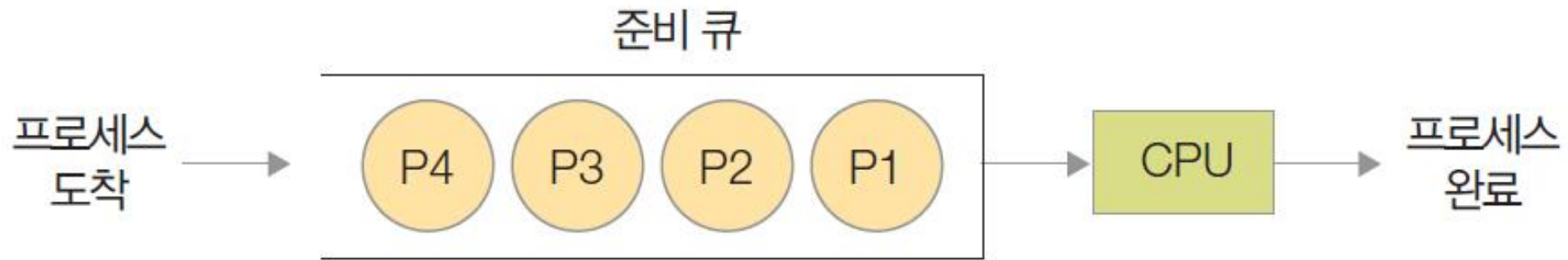
- 스케줄링의 공정성
- 처리량 최대화
- 응답 시간 최소화
- 반환 시간 예측 가능
- 자원의 균형적인 사용
- 응답 시간과 자원 활용도의 조화
- 프로세스 실행의 무한 연기 배제
- 우선순위에 따른 실행 보장
- 시스템의 과도한 부하 방지

CPU 스케줄링 기법

- FCFS
- 라운드 로빈
- SJF
- SRT
- HRN
- 우선 순위
- MLQ
- MFQ

FCFS

- 준비 큐에 도착한 프로세스 순서대로 CPU를 할당하는 방식
- 비선점 방식



라운드 로빈

- FCFS 기법과 동일하게 준비 큐에 도착한 프로세스 순서대로 CPU를 할당하지만, 프로세스마다 CPU 사용 시간을 일정하게 할당하여 시간이 지나면 다음 프로세스에게 CPU 사용권을 넘김
- 선점 방식



SJF

- 준비 큐에서 대기하고 있는 프로세스 중 예상 실행 시간이 가장 짧은 것을 우선 처리
- 비선점 방식



SRT

- 실행 중인 프로세스의 잔여 실행 시간과 준비 큐에서 대기 중인 프로세스의 잔여 실행 시간을 비교하여 가장 짧은 프로세스에게 CPU를 할당
- 선점 방식



HRN

- 실행 시간이 길더라도 대기를 오래 했다면 우선 처리될 수 있도록 스케줄링할 때 매번 프로세스의 응답률(우선순위)을 계산
- 응답률 공식

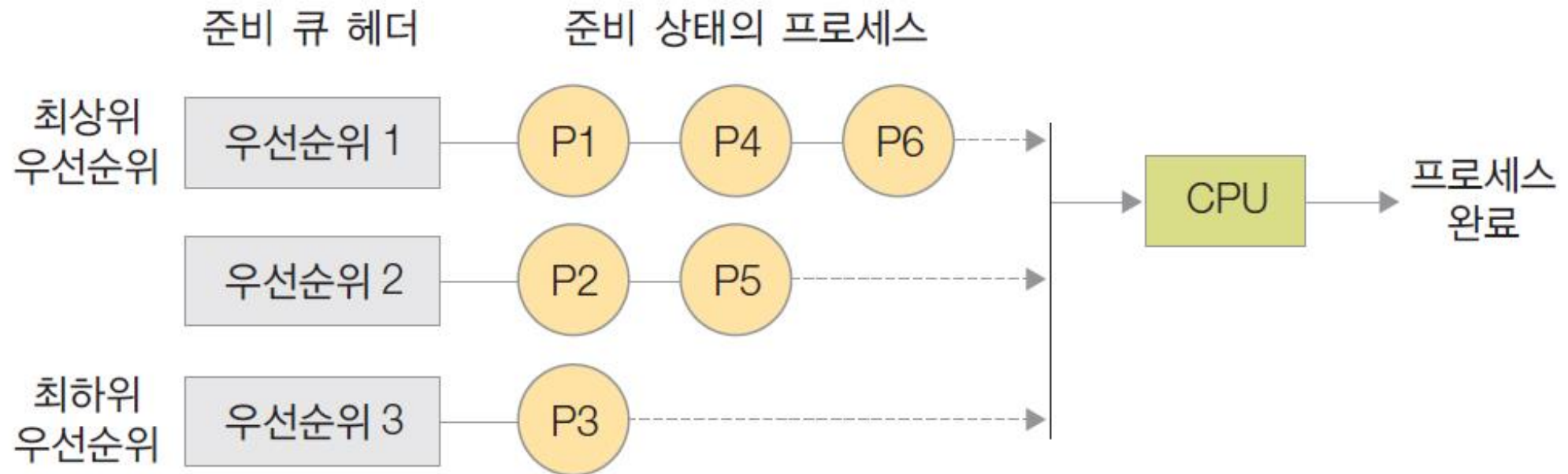
$$\text{응답률} = (\text{대기 시간} + \text{실행 시간}) / \text{실행 시간}$$

- 비선점 방식



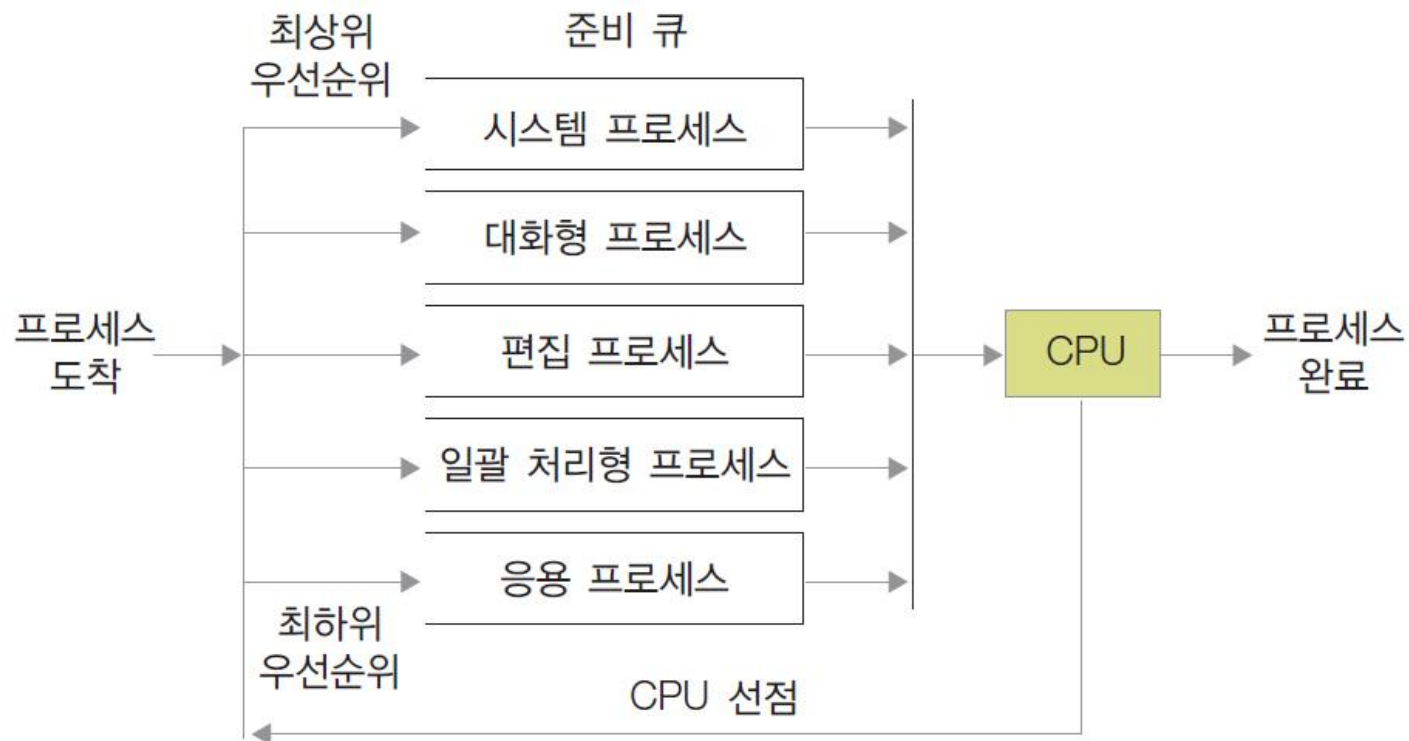
우선 순위

- 프로세스마다 우선순위를 부여해 최상위 우선순위 프로세스부터 처리하는 방식
- 비선점 방식



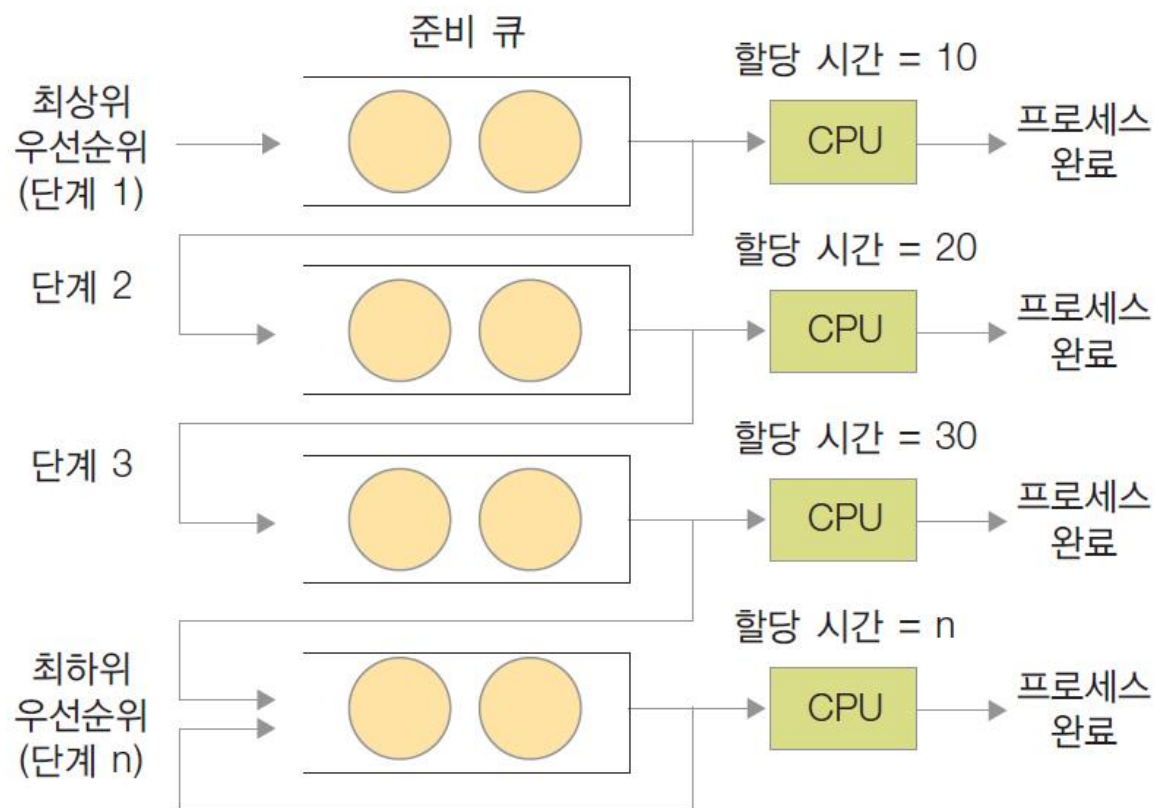
MLQ

- 작업 특성에 따라 별도의 준비 큐를 배정하여 각 큐마다 별도의 스케줄링 기법을 적용하는 방식
- 선점 방식



MFQ

- 입출력 위주 프로세스와 연산 위주 프로세스의 특성에 따라 CPU 사용 시간(할당량)을 다르게 부여하는 스케줄링 기법
- 선점 방식



CPU 스케줄링 기법의 성능 기준

- CPU 활용률
- 처리율
- 반환 시간
- 대기 시간
- 응답 시간

주기억장치 관리 기법

- 단일 프로그래밍 기법
- 고정 분할 다중 프로그래밍 기법
- 가변 분할 다중 프로그래밍 기법

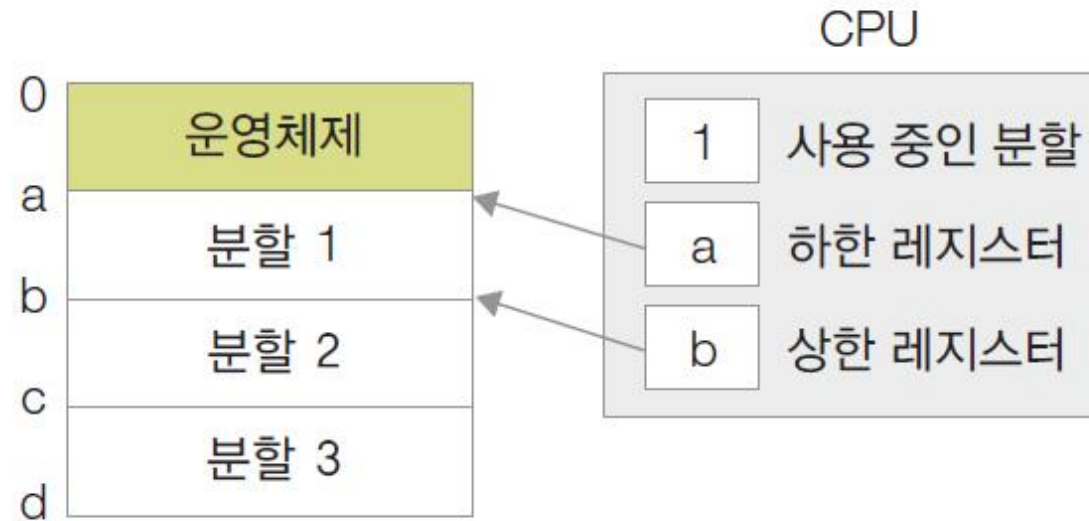
단일 프로그래밍 기법

- 운영체제와 하나의 사용자 프로그램만 주기억장치에 적재
- 주기억장치의 빈 공간이 있어도 다른 사용자 프로그램을 실행할 수 없음
- 자원 낭비가 심함
- 주기억장치의 용량보다 작은 프로그램만 실행



고정 분할 다중 프로그래밍 기법

- 주기억장치를 다수의 고정된 크기로 나눠서 실행 중인 여러 프로세스에게 할당
- 분할은 하나의 작업이 적재될 수 있는 일정한 크기의 기억 공간



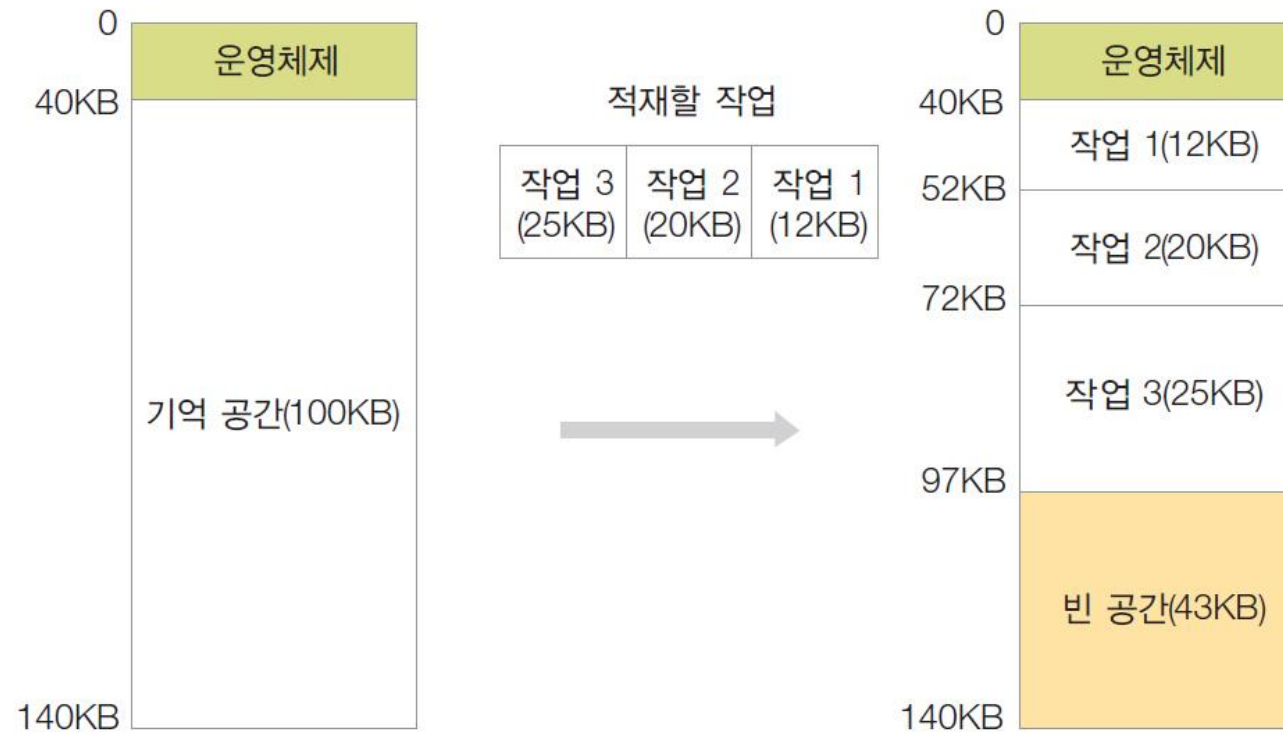
고정 분할 다중 프로그래밍 기법

- 작업과 분할의 크기가 일치하지 않아 사용되지 않는 빈 공간이 생기거나, 분할이 너무 작아서 작업을 적재하지 못하는 단편화 현상이 발생
 - 내부 단편화 : 분할에 작업을 적재한 후 빈 공간이 남는 현상
 - 외부 단편화 : 적재할 작업보다 분할의 크기가 작아서 분할이 빈 공간으로 남는 현상



가변 분할 다중 프로그래밍 기법

- 고정 분할 다중 프로그래밍 기법의 단점을 보완한 기법
- 고정된 분할의 경계를 없애고 각 작업에게 필요한 만큼의 기억 공간을 할당
- 작업이 완료되면 사용되지 않는 기억 공간을 회수하여 관리



가변 분할 다중 프로그래밍 기법

- 새로운 작업의 적재와 완료된 작업의 기억 공간 회수가 반복됨에 따라 단편화 현상이 발생



가변 분할 다중 프로그래밍 기법

- 통합 : 인접한 빈 공간을 합병하여 하나의 큰 공백을 형성하는 기법
- 집약 : 주기억장치의 한쪽 끝에 하나의 큰 공백을 형성하는 기법



주기억장치 관리 전략

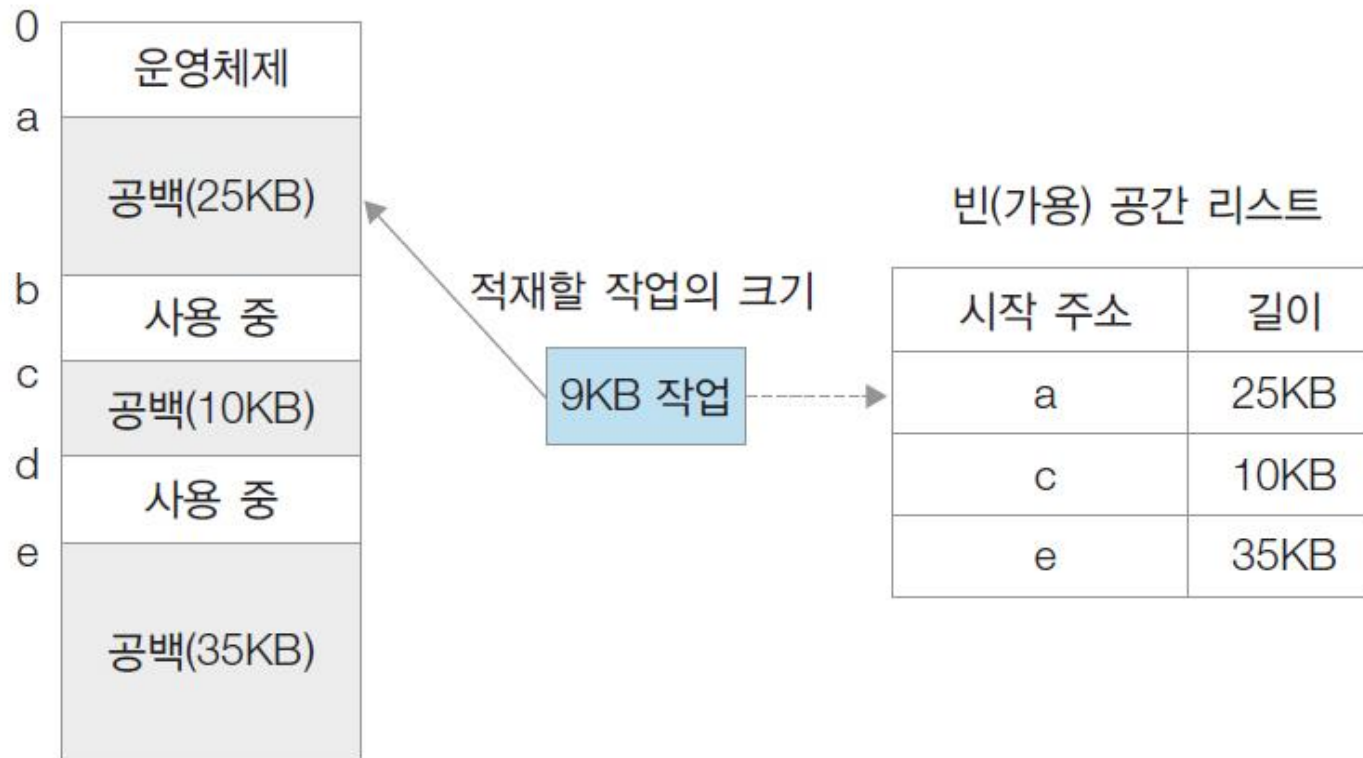
- 반입 전략
- 배치 전략
- 교체 전략

반입 전략

- 다음에 실행할 프로그램이나 참조할 데이터를 보조기억장치에서 주기억장치로 언제 가져올지 결정하는 전략
- 요구 반입 전략
- 예상 반입 전략

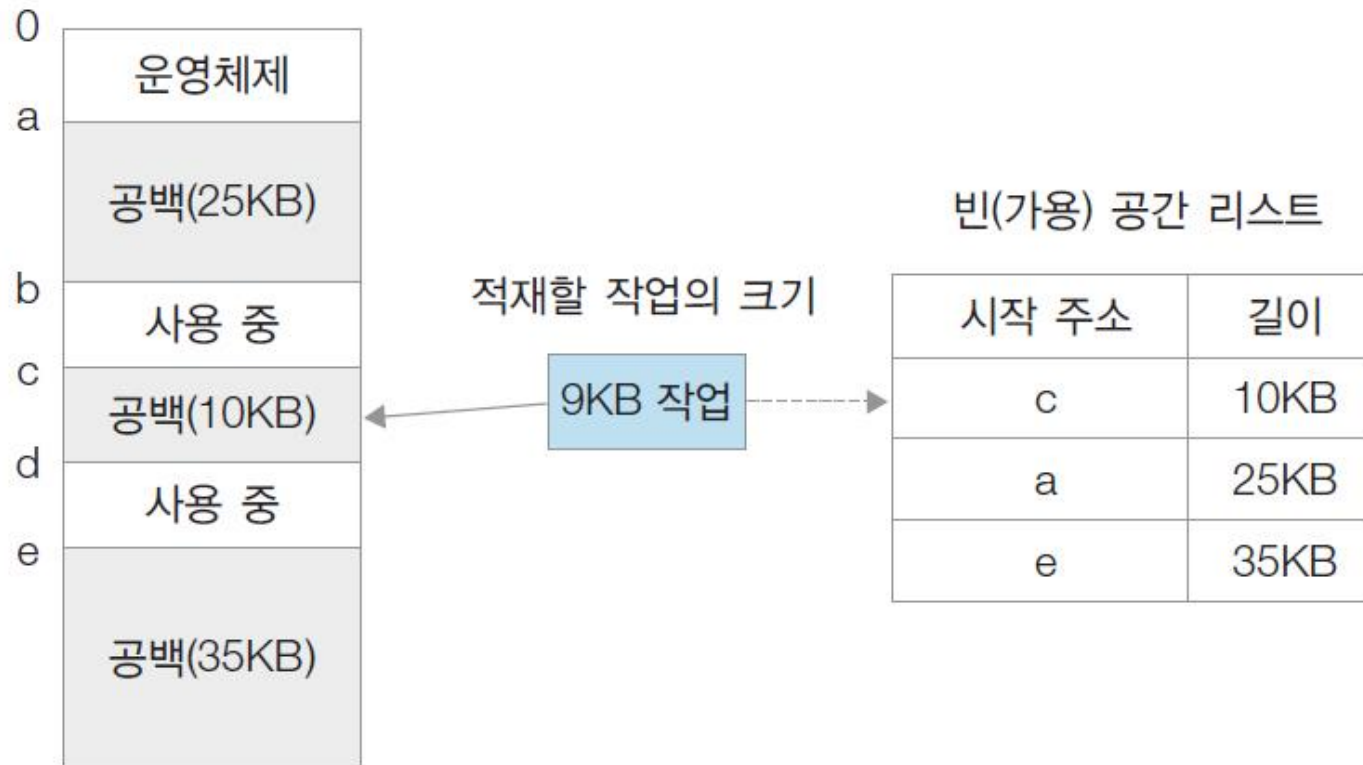
배치 전략

- 새로 가져온 프로그램이나 데이터를 주기억장치의 어디에 배치할지 결정하는 전략
- 최초 적합



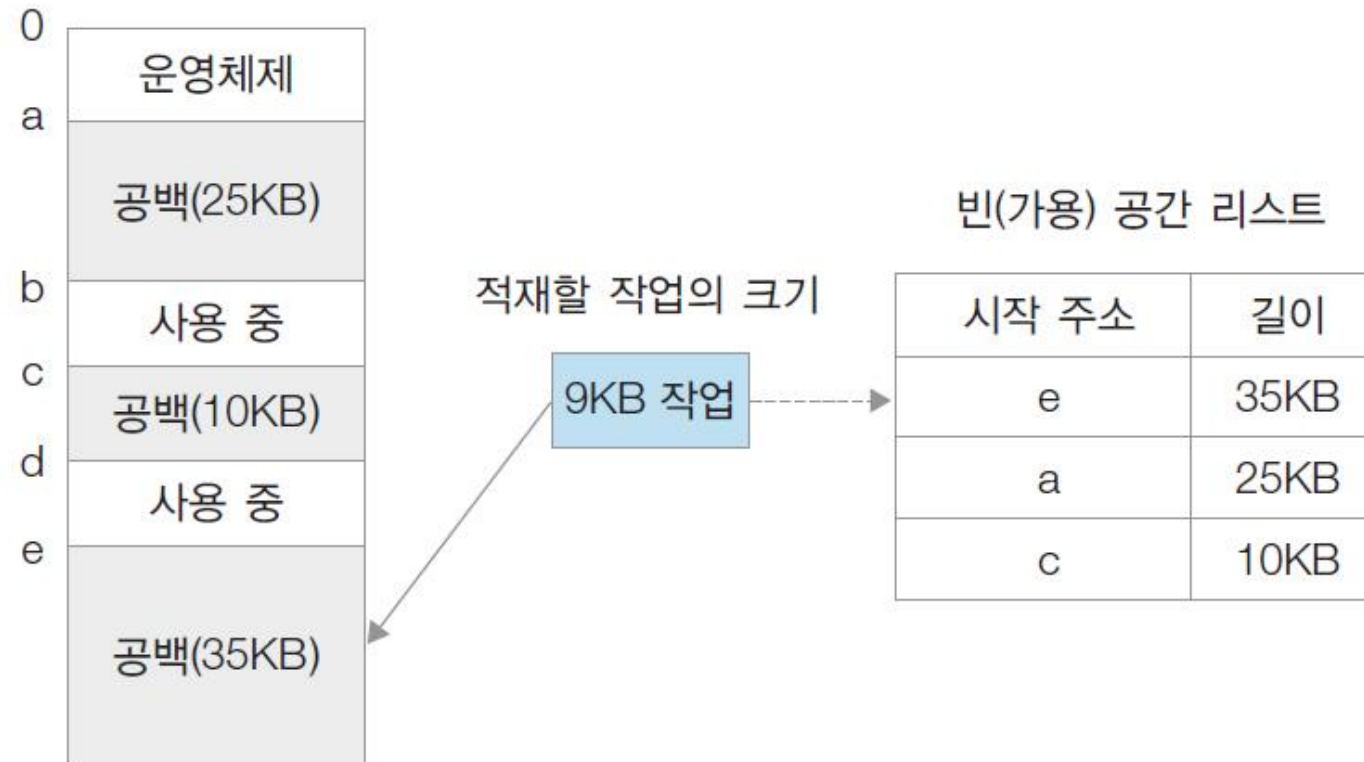
배치 전략

- 새로 가져온 프로그램이나 데이터를 주기억장치의 어디에 배치할지 결정하는 전략
- 최적 적합



배치 전략

- 새로 가져온 프로그램이나 데이터를 주기억장치의 어디에 배치할지 결정하는 전략
- 최악 적합



교체 전략

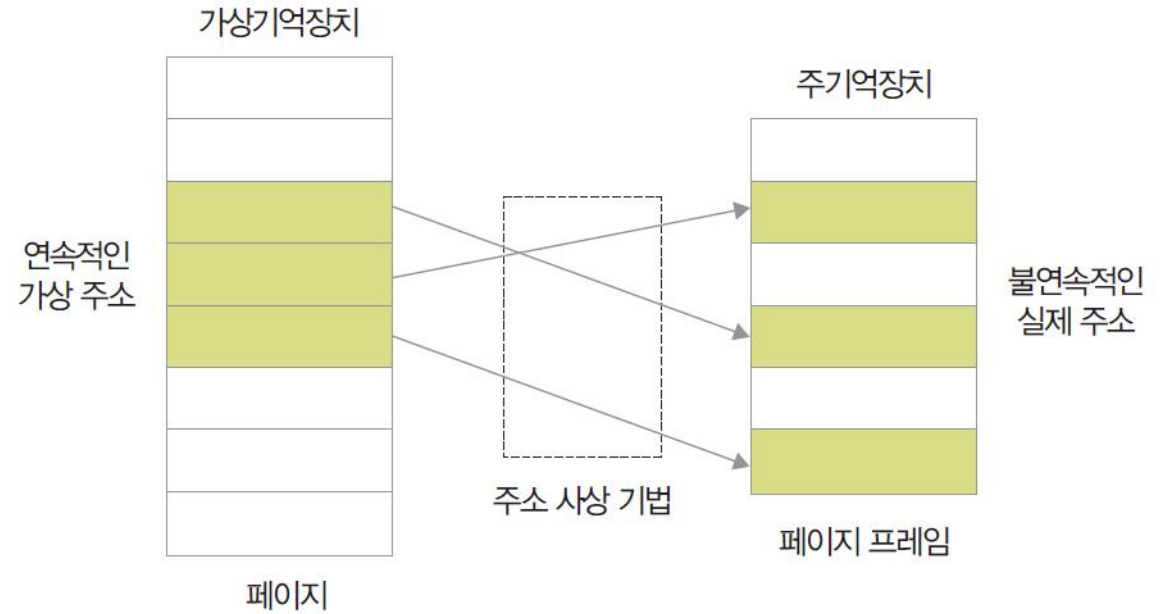
- 주기억장치에 새로 반입할 프로그램이나 데이터를 배치할 빈 공간이 없을 때, 배치 공간을 마련하기 위해 주기억장치에서 어떤 프로그램이나 데이터를 제거할지 결정하는 전략
- 최적화 원리, 무작위 교체, 선입 선출 교체, LRU 교체, NUR 교체, LFU 교체, 2차 기회 교체 등

가상기억장치 구현 기법

- 가상기억장치
- 페이징 기법
- 세그멘테이션 기법
- 페이징/세그멘테이션 혼합 기법

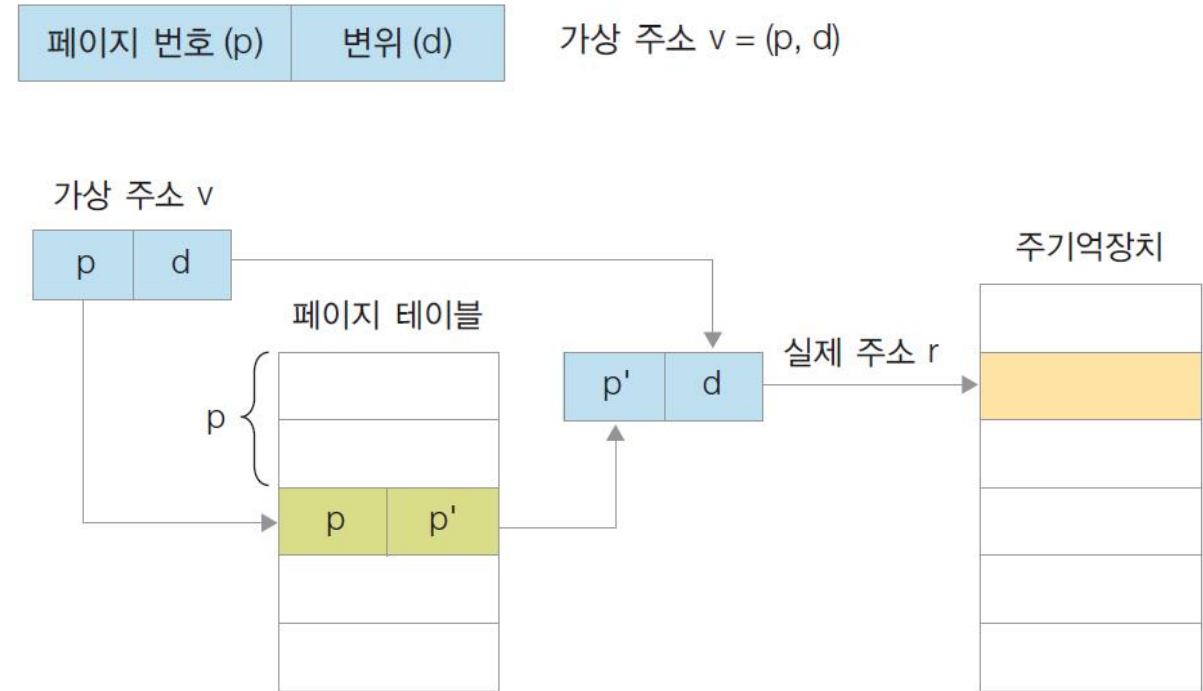
가상기억장치

- 현재 실행 중인 프로그램의 일부는 주기억장치에 적재하고, 나머지는 보조기억장치에 유지
- 프로세스가 참조하는 가상 주소는 주소사상 함수에 의해 실행 중에 주기억장치에서 사용할 수 있는 실제 주소로 변환하는 동적 주소 변환 과정을 거쳐야 함



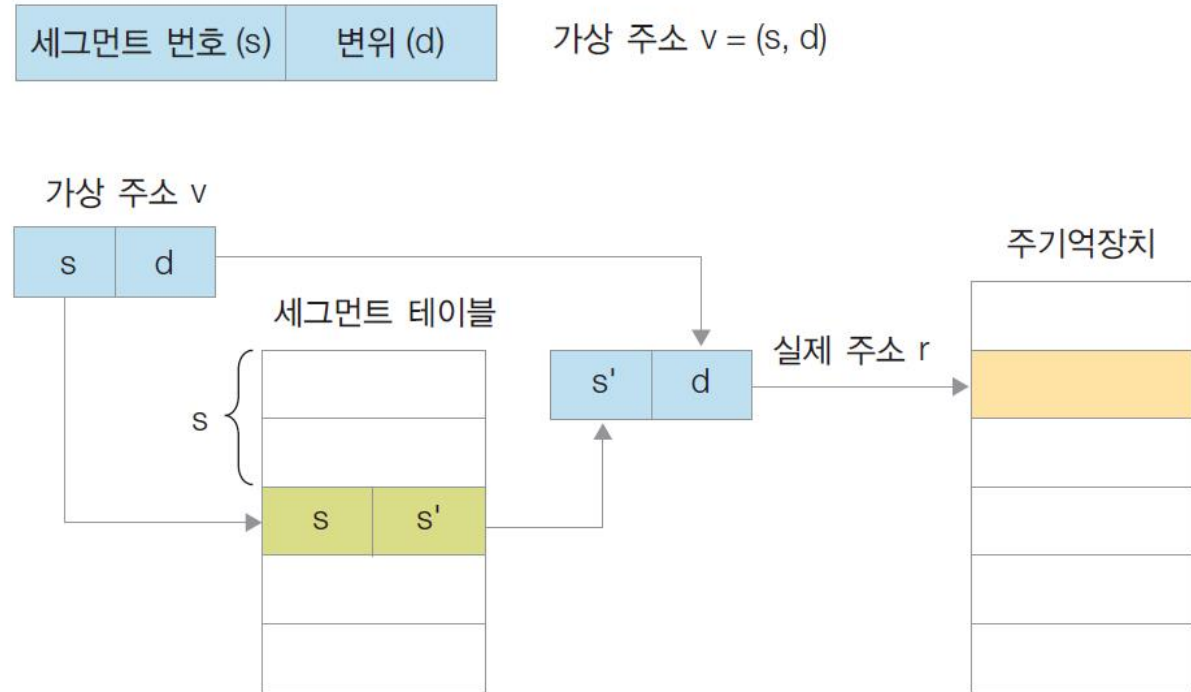
페이징 기법

- 가상기억장치를 일정한 크기의 페이지로 나눠서 관리
- 주기억장치는 페이지와 크기가 동일한 페이지 프레임으로 분할하여 사용
- 가상 주소는 $V=(p, d)$
- 프로세스마다 별도의 페이지 테이블을 가짐



세그멘테이션 기법

- 프로그램이나 데이터를 용도에 따라 가변적인 크기로 분할하여 관리
- 가변적인 크기로 분할된 프로그램이나 데이터 블록을 세그먼트
- 가상 주소는 $V=(s, d)$
- 프로세스마다 별도의 세그먼트 테이블을 가짐

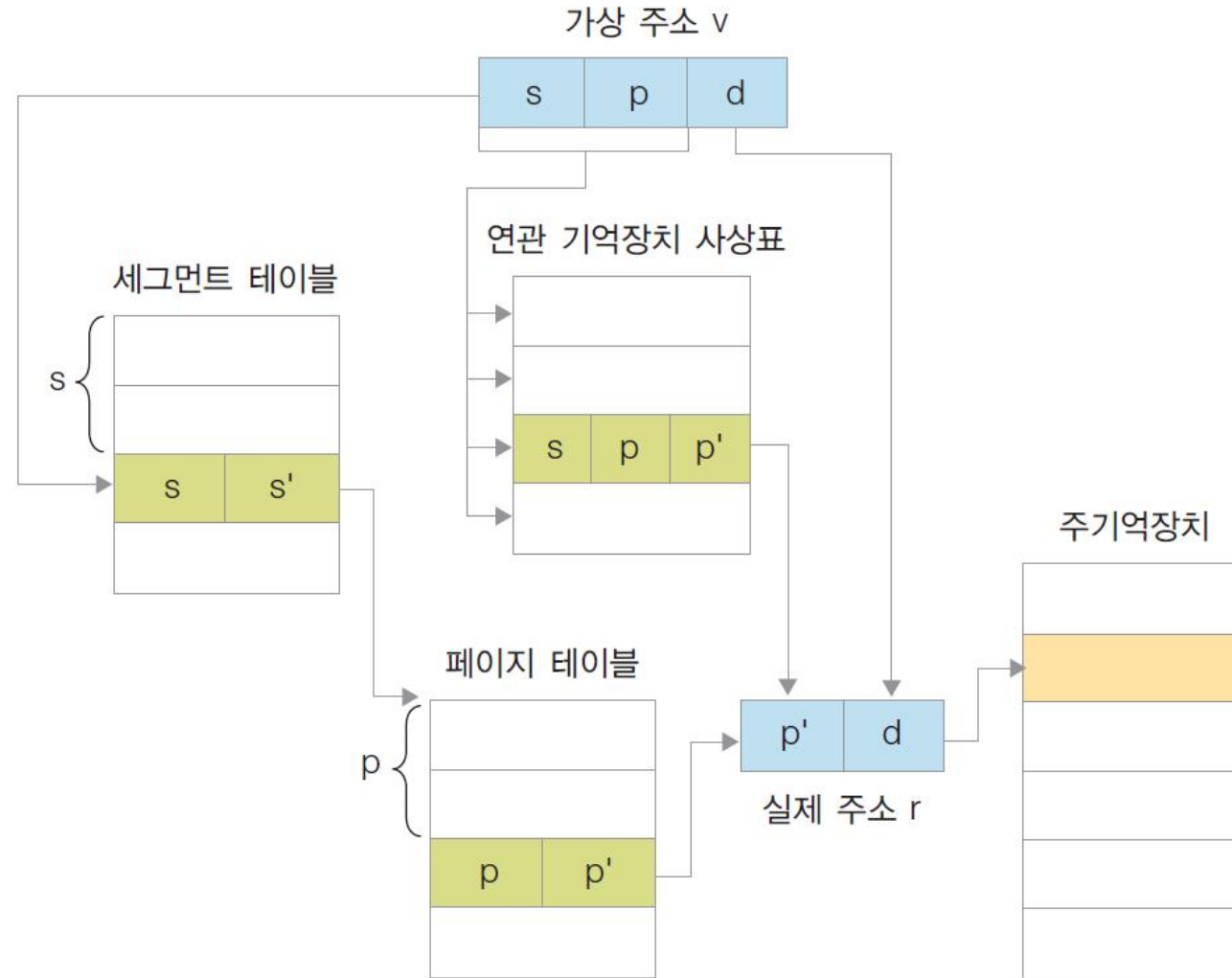


페이징/세그멘테이션 혼합 기법

- 모든 세그먼트를 페이지 단위로 다시 분할해 외부 단편화 현상을 해결
- 세그먼트의 크기는 페이지의 정수 배
- 가상 주소는 $V=(s, p, d)$

세그먼트 번호 (s)	페이지 번호 (p)	변위 (d)	가상 주소 $v = (s, p, d)$
-------------	------------	--------	-----------------------

페이징/세그멘테이션 혼합 기법



파일 시스템의 기능

- 사용자에게 편리한 사용자 인터페이스를 제공
- 사용자가 파일을 생성, 수정, 삭제할 수 있는 기능을 제공
- 적절한 제어 방법을 통해 다른 사람의 파일을 공유할 수 있는 기능을 제공
- 불의의 사고에 대비하기 위한 파일 백업 및 복구 기능을 제공
- 정보를 암호화하고 해독할 수 있는 능력을 제공

파일 구조

- 순차 파일
- 색인 순차 파일
- 직접 파일
- 분할 파일

순차 파일

- 레코드를 순차적으로 저장하고 검색
- 장점 : 레코드 사이에 빈 공간이 없어 기억장치를 효율적으로 이용함
- 단점 : 레코드의 삽입과 삭제가 어렵고 파일 검색에 많은 시간이 소요됨

색인 순차 파일

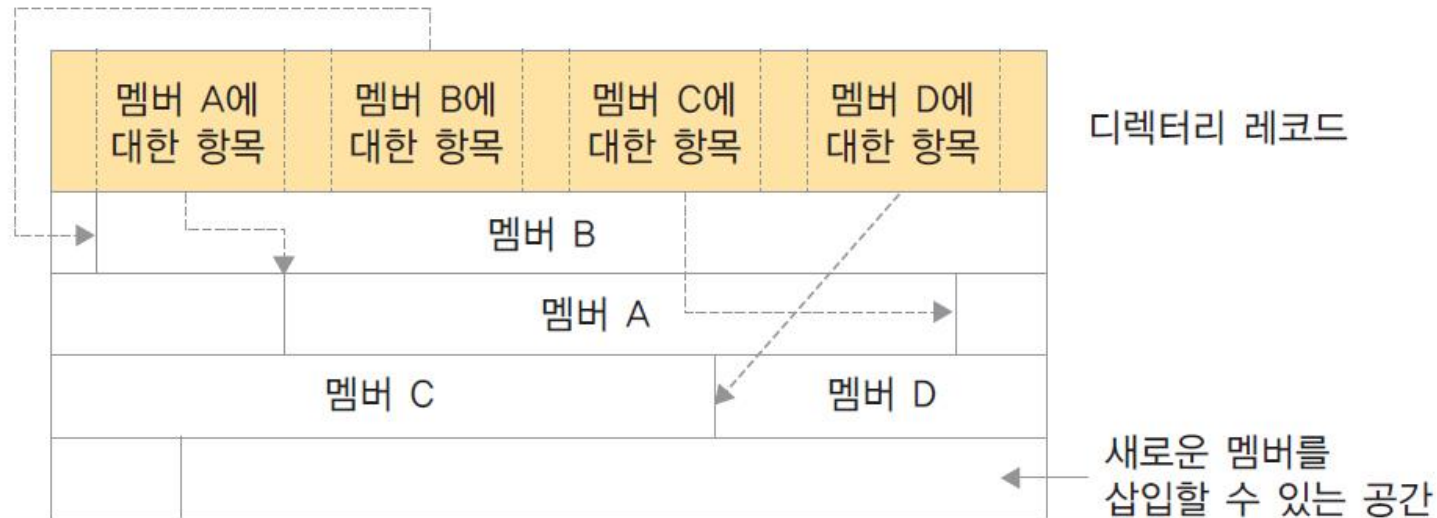
- 키 값에 따라 정렬된 레코드를 순차적으로 접근하는 순차 파일 구조와 키 값에 따라 직접적으로 접근하는 직접 파일 구조가 모두 사용
- 데이터 파일과 색인 파일로 구성
- 장점 : 비교적 레코드에 빠르게 접근할 수 있으며 레코드의 삽입과 수정이 용이
- 단점 : 처리 속도가 늦고, 색인을 저장하는 공간이 별도로 필요함

직접 파일

- 특정 레코드에 직접 접근할 수 있는 파일 형태
- 레코드가 저장된 보조기억장치의 물리 주소를 키 값으로 계산하여 접근
- 장점 : 처리 속도가 빠르고, 다른 레코드에 영향을 주지 않으면서 특정 레코드를 삽입, 삭제, 검색, 수정할 수 있음
- 단점 : 키 값에 의한 순차 검색이 어려움

분할 파일

- 다수의 순차적인 서브 파일로 구성
- 프로그램 라이브러리나 매크로 라이브러리를 저장할 때 유용
- 순차적인 서브 파일의 시작 주소는 디렉터리 레코드에 저장

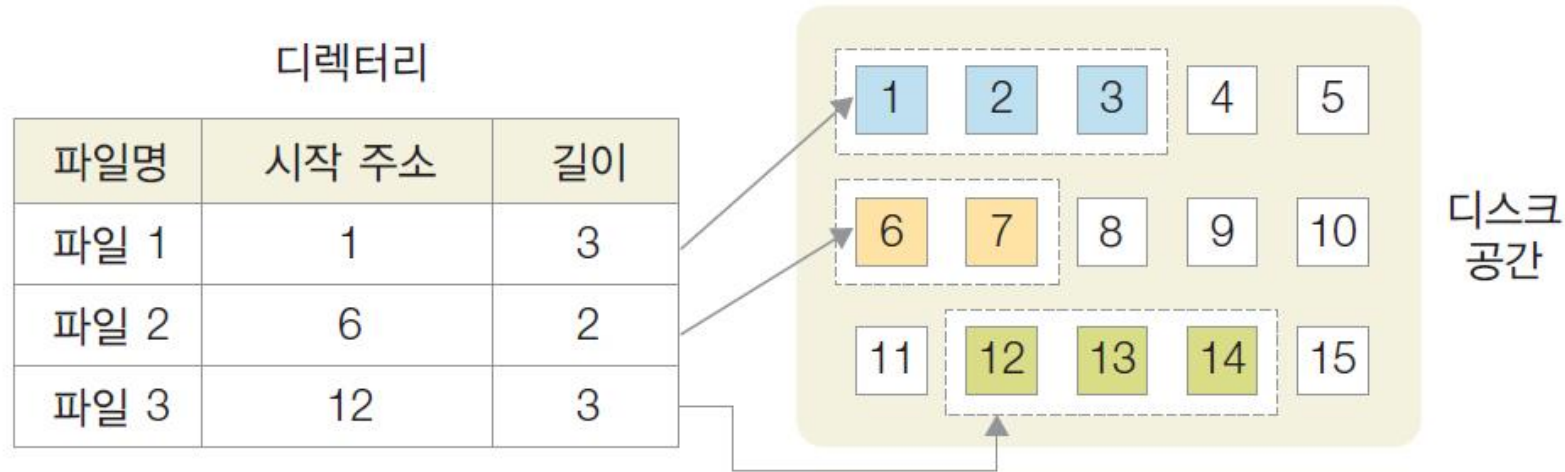


파일의 공간 할당 기법

- 연속 할당
- 불연속 할당

연속 할당

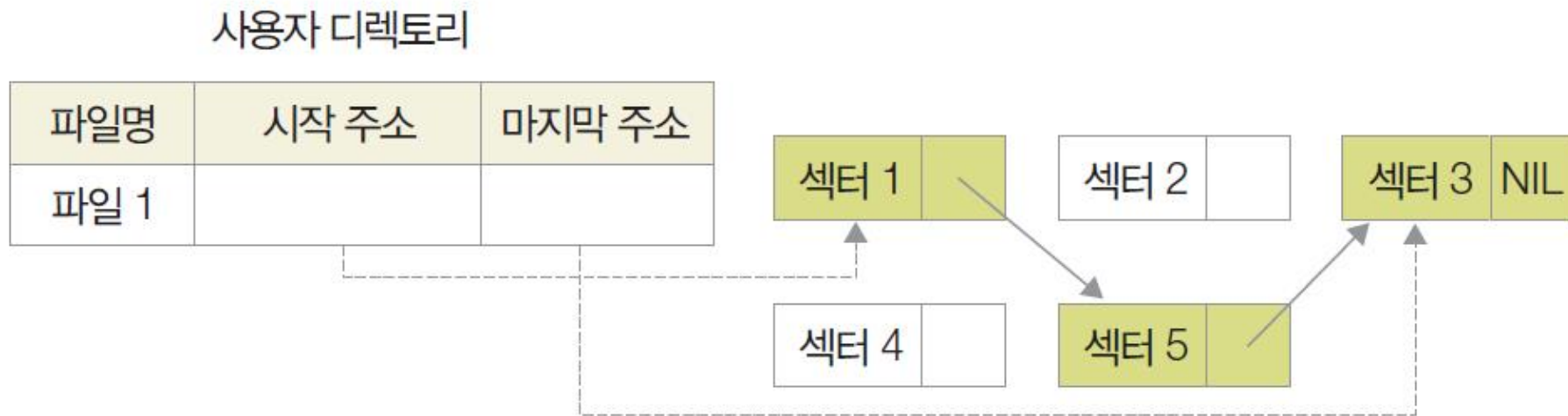
- 파일을 디스크 내의 연속적인 빈 공간에 저장
- 논리적으로 연속된 레코드들이 물리적으로도 인접하게 저장되어 접근 시간이 감소
- 디렉터리에는 파일을 구성하는 첫 번째 블록의 시작 주소와 길이만 유지하면 되므로 구현이 쉬움
- 주기적인 집약이 필요



불연속 할당

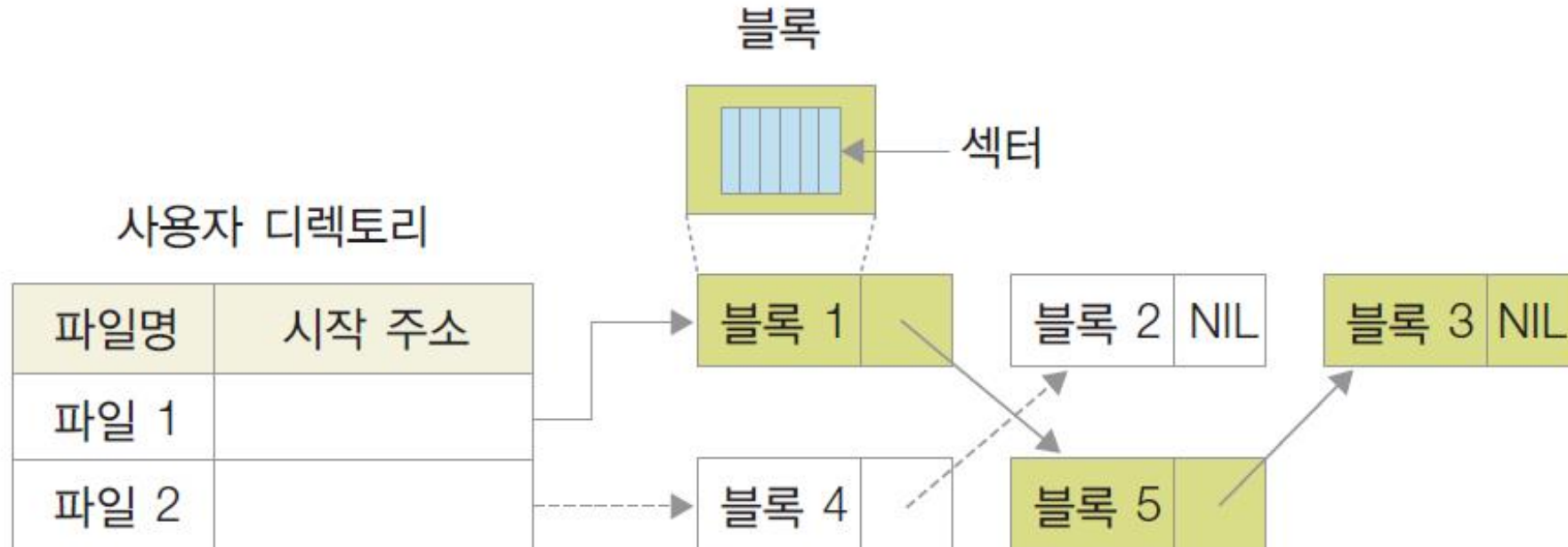
■ 섹터 단위 할당

- 하나의 파일에 속한 다수의 섹터들이 연결 리스트로 구성
- 각 섹터는 다른 섹터와의 연결을 위한 포인터를 가짐
- 디렉터리는 각 파일의 시작 주소와 마지막 주소에 대한 포인터를 보유



불연속 할당

- 블록 단위 할당
 - 다수의 섹터를 묶은 블록 단위로 할당
 - 연속 할당과 불연속 할당의 절충형



파일 보호

- 접근 통제 방법

- 접근 제어 행렬(access control matrix)
- 파일 명명(file naming)
- 암호(password)
- 암호 기법(cryptography)

- 파일 백업

- 백업은 불의의 사고로 인한 데이터의 손상과 파괴를 막기 위해 파일을 복사하여 다른 곳에 보존하는 방법