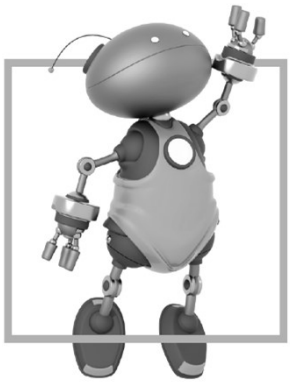
한빛아카데미 Hanbit Academy, Inc.



IT 세상을 만나는

컴퓨터 개론 2판

인공지능, 빅데이터, 확장현실까지

Chapter 09. 운영체제

목차

학습목표

- 운영체제의 정의와 구조를 이해하고 그 역사를 살펴본다.
- 프로그램과 프로세스의 차이를 이해하고
  프로세스의 상태를 파악한다.
- 메모리 관리의 어려움을 이해하고
  가상 메모리 시스템에 대해 알아본다.
- 파일과 디렉터리의 특징을 파악하고
  파일 시스템 구조에 대해 알아본다.

01
운영체제의 개요

01. 운영체제의 개요

1. 운영체제의 개념

- 커피 재료, 커피머신과 같은 자원을 여러 사람으로부터 보호하는 방법
  - 관리자를 두기

(a) 커피머신을 자유롭게 사용하는 경우 (b) 바리스타가 커피머신을 관리하는 경우

그림 9-1 커피머신과 자원 관리

01. 운영체제의 개요

1. 운영체제의 개념

▪ 운영체제의 자원 관리

- **컴퓨터 자원** computer resource
  - 컴퓨터에 있는 CPU, 메모리, 하드디스크, 모니터, 키보드, 마우스
- **운영체제** Operating System(OS)
  - 컴퓨터 전체를 관리하고 운영하는 소프트웨어
  - 모든 소프트웨어 위에 존재하는 최고의 소프트웨어
  - 컴퓨터를 관리하기 위한 기본적인 규칙과 절차를 규정한 소프트웨어

01. 운영체제의 개요

1. 운영체제의 개념

▪ **다양한 운영체제 제품**

- 마이크로소프트의 윈도우 Windows, 리눅스 Linux, 애플 맥OS Mac OS
- 스마트폰 전용 운영체제
  구글의 안드로이드 Android와 애플의 iOS iPhone-OS

그림 9-2 **다양한 운영체제 제품**

7 / 65

01. 운영체제의 개요

1. 운영체제의 개념

▪ **다양한 운영체제 제품**

- **임베디드 운영체제** embedded operating system
  - 임베디드 시스템에 사용하도록 만든 운영체제
  - 워치 OS, 웨어 OS
  - CPU의 성능이 낮고 메모리 크기가 작은 기계에 설치되기 때문에 일반 운영체제에 비해 크기가 작고 몇 가지 기능이 빠져 있음

그림 9-3 **갤럭시 워치에 적용된 웨어 OS**

8 / 65

01. 운영체제의 개요

1. 운영체제의 개념

▪ 운영체제의 역할

- 응용 프로그램과 사용자가 자원에 직접 접근하는 것을 막음으로써 컴퓨터 자원을 보호하고 관리함
- 자원을 이용하는 여러 가지 방법을 제공
- 인터페이스: 컴퓨터를 사용하는 것을 도와주고 그 결과를 알려줌

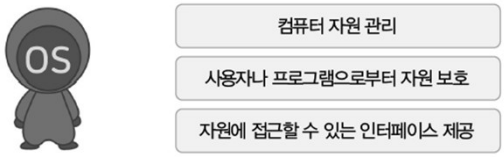


그림 9-4 **운영체제의 역할**

01. 운영체제의 개요

1. 운영체제의 개념

▪ 운영체제의 구성

- **커널** kernel : 운영체제의 핵심적인 기능을 모아 놓은 것
- **인터페이스** : 커널에 명령을 내리고 결과를 사용자, 응용 프로그램에 전달

그림 9-5 **운영체제 관점에서 나타낸 컴퓨터의 구조**

01. 운영체제의 개요

1. 운영체제의 개념

- **인터페이스**
  - 명령을 내리고 결과를 사용자, 응용 프로그램에 전달
  - 응용 프로그램이 컴퓨터 자원을 사용하려면 인터페이스 이용해야 함
  - 운영체제는 함수 형태의 인터페이스를 제공
  - 사용자 기반 인터페이스, 문자 기반 인터페이스,
    그래픽 사용자 인터페이스, 터치스크린 인터페이스

11 / 65

01. 운영체제의 개요

1. 운영체제의 개념

- **자동차의 인터페이스**
  - 자동차의 전자 제어 장치가 독립적이어서
    업그레이드하려면 부품 제조사에서 소프트웨어를 받아 교체해야 했음
  - 자동차에 운영체제를 도입하여 전자 제어 장치를 통합 관리할 수 있음

그림 9-6 **테슬라 자동차의 터치스크린 제어판**

12 / 65

01. 운영체제의 개요

1. 운영체제의 개념

▪ **자동차의 인터페이스**

- 자동차에서 운영체제를 사용하면
  - 자동차 제어를 통합할 수 있고
  - 새로운 기능을 구현하기 쉬움
  - 업그레이드할 때 무선통신을 이용하여 자동 다운로드
- **OTA** Over-The-Air Programming
  - 차량 및 무선 기기의 소프트웨어 업그레이드, 설정 변경을 무선 배포하는 기술

13 / 65

01. 운영체제의 개요

2. 운영체제의 주요 작업

- 커널의 주요 작업
  - 프로세스 관리
  - 메모리 관리
  - 파일 시스템
  - 입출력 관리
  - 프로그램 간 통신 환경 제공

14 / 65

01. 운영체제의 개요

2. 운영체제의 주요 작업

- **장치 드라이버** device driver
  - 커널과 하드웨어 사이의 인터페이스
  - 표준을 따르는 주변 장치는 컴퓨터에 꽂기만 하면 바로 사용할 수 있음

그림 9-7 **커널과 장치 드라이버**

15 / 65

01. 운영체제의 개요

2. 운영체제의 주요 작업

- **장치 드라이버** device driver
  - 표준 이외 기능은 드라이버 소프트웨어를 설치해야 함

그림 9-8 **엔비디아의 그래픽카드 드라이버 다운로드 페이지**

16 / 65

01. 운영체제의 개요

2. 운영체제의 주요 작업

▪ 운영체제를 시작하는 부팅

- **부팅** booting 운영체제를 메모리로 올리는 과정
  - 운영체제도 프로그램이기 때문에 메모리에 올라와야 실행됨

(a) 윈도우 부팅 화면　　(b) 유닉스 부팅 화면

그림 9-9 **부팅**

01. 운영체제의 개요

2. 운영체제의 주요 작업

▪ 운영체제를 시작하는 부팅

- 전원을 켜면 가장 먼저 바이오스가 하드웨어를 점검함

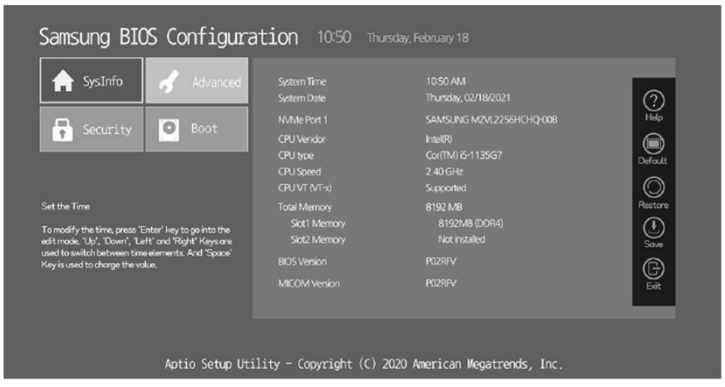


그림 9-10 **바이오스 설정 화면**

01. 운영체제의 개요

2. 운영체제의 주요 작업

- **운영체제를 시작하는 부팅**
  - **부트스트랩 코드** bootstrap code 실행
  - 운영체제의 필수 프로그램을 메모리에 올려 실행

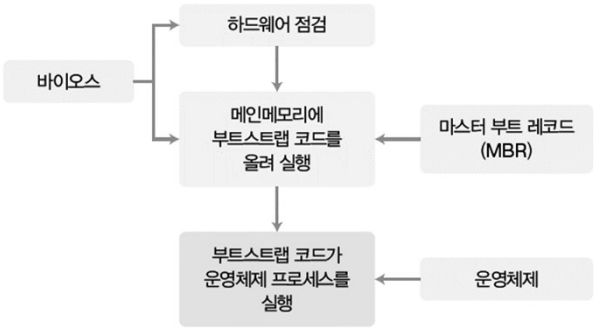


그림 9-11 **부팅 과정**

01. 운영체제의 개요

2. 운영체제의 주요 작업

- **API와 SDK**
  - **응용 프로그램 인터페이스** Application Programming Interface(API)
  - **시스템 개발자용 키트** system developer's kit(SDK)

그림 9-12 **구글 지도에서 제공하는 SDK와 API**

01. 운영체제의 개요

3. 운영체제의 역사

- 운영체제는 컴퓨터를 여러 사람이 다양한 방법으로 이용하도록 발전함

▪ 유닉스 계열의 운영체제

- 유닉스: 1969년 켄 톰프슨이 개발한 단순한 운영체제
- BSD 유닉스: 1978년 캘리포니아대학교에서 유닉스를 수정하여 배포함

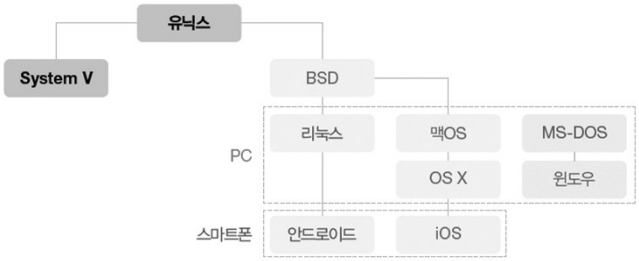


그림 9-14 주요 운영체제의 버전

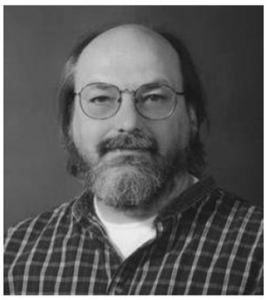


그림 9-13 켄 톰프슨

01. 운영체제의 개요

3. 운영체제의 역사

▪ 안드로이드가 무료인 이유

- 리처드 스톨먼이 **GNU**를 창설
  - 소프트웨어를 유료로 팔지 말고 누구나 자유롭게 실행·복사·수정·배포할 수 있게 하자고 주장함
- 소프트웨어에 라이선스로 GPL 부여
  - copyleft
- 오픈 소스 소프트웨어는 모두 공짜

01. 운영체제의 개요

3. 운영체제의 역사

- **안드로이드가 무료인 이유**
  - 리눅스: 1991년 리누스 토르발스가 PC용 유닉스 호환 커널을 GPL로 배포
  - 유닉스는 안정적인 운영체제로 발전함

그림 9-15 리누스 토르발스

23 / 65

01. 운영체제의 개요

3. 운영체제의 역사

- **개인용 컴퓨터 운영체제**
  - 애플 II 이후 개인용 컴퓨터 대중화
  - IBM XT 호환 컴퓨터가 다수 생산됨
    → MS-DOS 운영체제를 판매한 마이크로소프트 성장
  - MS에서 그래픽 사용자 인터페이스 적용한 윈도우 운영체제 출시

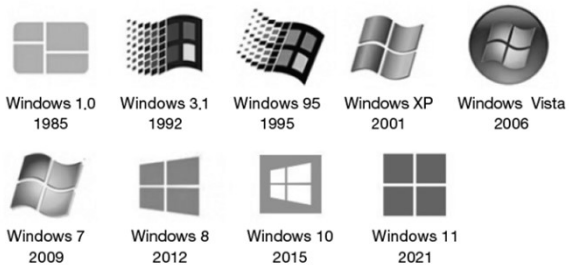


그림 9-16 윈도우의 발전사

24 / 65

01. 운영체제의 개요

3. 운영체제의 역사

▪ **개인용 컴퓨터 운영체제**

- 유닉스 운영체제(오픈 소스)는 크기가 작고 안정적인 운영체제로 구현됨
- ↔ 윈도우 운영체제는 안정화에 오랜 시간이 걸렸음

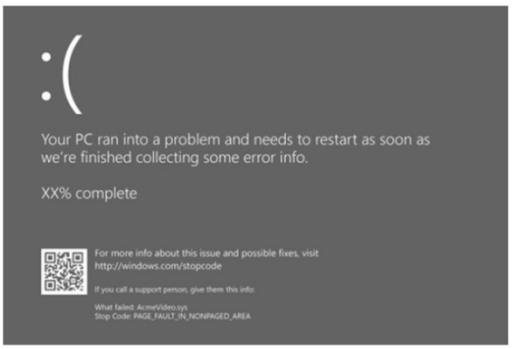


그림 9-17 **윈도우의 블루 스크린**

25 / 65

01. 운영체제의 개요

3. 운영체제의 역사

▪ **스마트폰 운영체제**

- 애플은 처음부터 아이폰에 iOS 탑재
- 구글의 모바일용 운영체제 안드로이드를 대부분의 스마트폰에서 사용함

26 / 65

02
프로세스 관리

02. 프로세스 관리

1. 프로세스

- **프로세스** process : 운영체제에서 작업의 단위
  - 크롬 브라우저(프로그램) 실행 → 크롬 프로세스

그림 9-18 **프로그램 실행과 프로세스 종료**

28 / 65

02. 프로세스 관리

1. 프로세스

- **프로그램과 프로세스의 차이**
  - 프로그램: 어떤 데이터를 사용하여 어떤 작업을 할지 그 절차를 적은 것
    - 저장 장치에 작업 내용이 저장된 정적인 상태
  - 프로세스: 프로그램으로 작성된 작업 절차를 실제로 실행에 옮김
    - 실행을 위해 작업 내용이 메모리에 올라온 동적인 상태

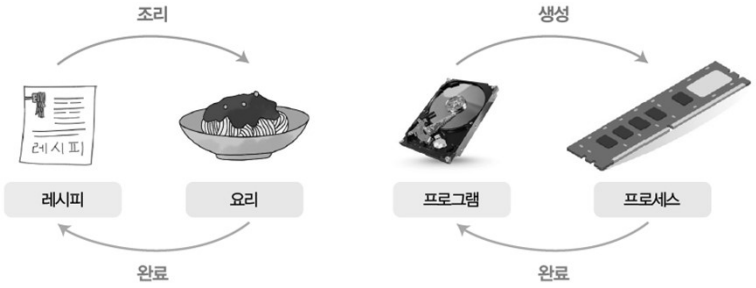


그림 9-19 **프로그램과 프로세스**

02. 프로세스 관리

1. 프로세스

- **프로그램과 프로세스의 차이**
  - 사용 중인 컴퓨터의 프로세스 확인: 윈도우 [Ctrl] + [Alt] + [Delete]

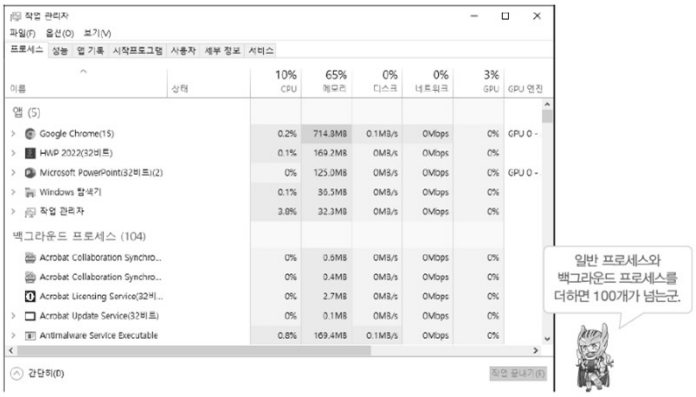


그림 9-20 **작업 관리자**

02. 프로세스 관리

2. 프로세스의 상태

- [요리 준비]에서 가장 앞에 있는 주문서부터 조리함
  - 한 테이블 주문서의 요리가 모두 만들어지면 주문서 폐기
  - 주문서에 처리할 요리가 남아있으면 주문서를 [요리 준비] 맨 뒤에 삽입

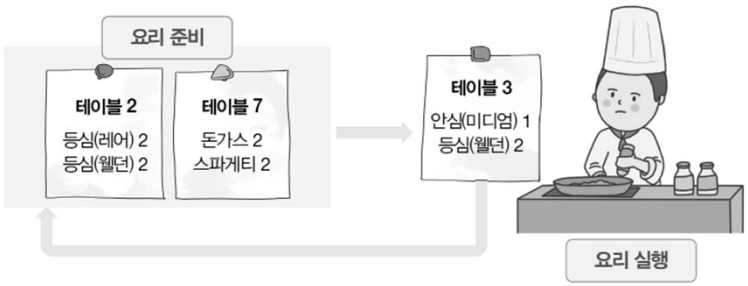


그림 9-21 요리 준비와 실행

02. 프로세스 관리

2. 프로세스의 상태

- 재료가 아직 준비되지 않은 경우 주문서를 [요리 대기]로 보내고 다른 주문서부터 처리

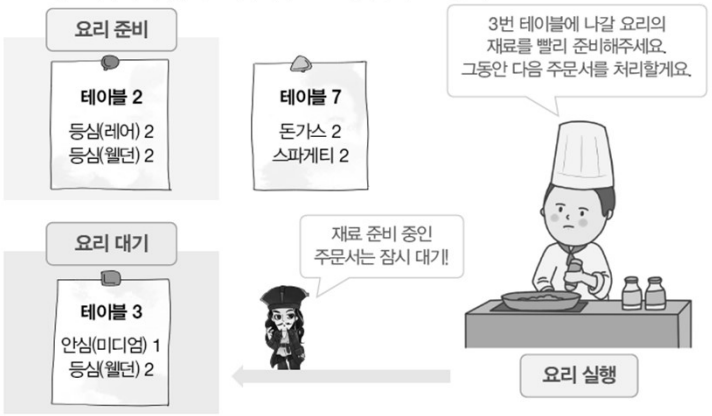


그림 9-22 요리 대기

02. 프로세스 관리

2. 프로세스의 상태

- 프로세스 상태: 생성, 준비, 실행, 대기, 완료

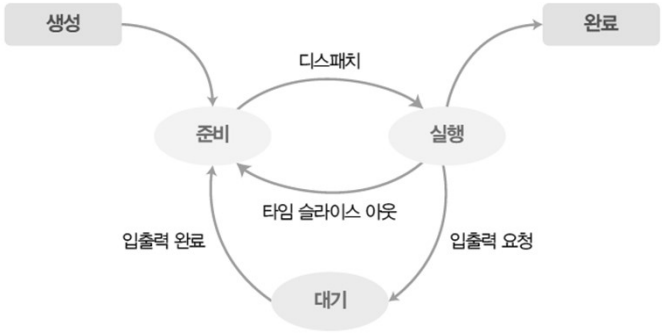


그림 9-23 **프로세스 상태도**

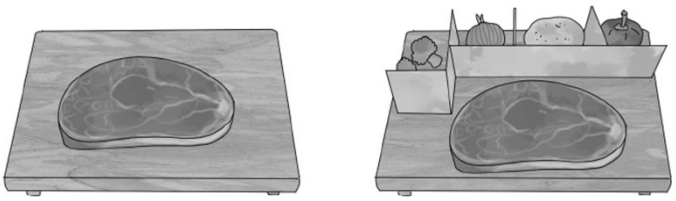
33 / 65

03
메모리 관리

03. 메모리 관리

1. 메모리 관리의 개요

- 도마에서 손질할 재료가 많아지면
  - 재료가 섞이지 않도록 칸막이를 만들어야 함
  - 칸막이를 옮겨서 도마를 나눈 공간의 크기를 조절해야 함
- 도마=메모리, 재료=프로세스



(a) 일괄 처리 시스템 (b) 시분할 시스템

그림 9-24 메모리 관리

03. 메모리 관리

1. 메모리 관리의 개요

- 운영체제도 프로세스로 메모리에 올라와야 실행 가능
- 메모리 영역 → 운영체제 영역 / 일반 영역

그림 9-25 일괄 처리 시스템과 시분할 시스템의 메모리 구조

03. 메모리 관리

1. 메모리 관리의 개요

- 일괄 처리 시스템의 메모리 관리자
  - 일반 프로세스가 운영체제 영역으로 침범하지 못하게 막기만 하면 됨
- 시분할 시스템의 메모리 관리자
  - 프로세스가 다른 프로세스 작업 영역으로 침범하는 것까지 막아야 함
  - 작업 영역이 부족하면 확보해야 함
  - 프로세스가 끝나면 작업 영역을 치워야 함

37 / 65

03. 메모리 관리

2. 메모리 오버레이

- 과거에 작은 메모리로 큰 프로그램을 실행해야 했음
- **메모리 오버레이** memory overlay
  - 프로그램의 크기가 물리 메모리보다 클 때
    프로그램을 적당한 크기로 잘라서 메모리에 가져오는 기법

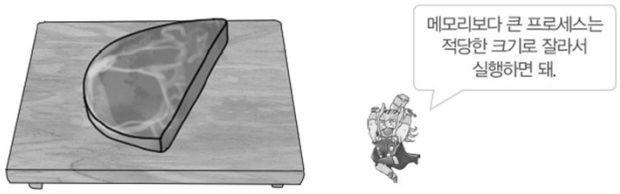


그림 9-26 **조리대보다 큰 고깃덩어리를 손질하는 경우**

38 / 65

03. 메모리 관리

2. 메모리 오버레이

- 프로그램을 몇 개의 모듈로 나누고 필요한 모듈을 메모리에 가져와 실행

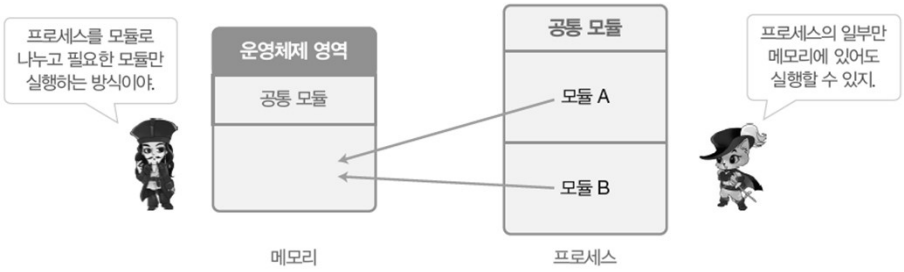


그림 9-27 **메모리 오버레이**

- 한정된 메모리에서 메모리보다 큰 프로그램 실행 가능
- 프로그램의 일부만 메모리에 올라와도 실행 가능

39 / 65

03. 메모리 관리

3. 스왑

- 프로그램에서 메모리에 올리지 않은 부분을 별도의 공간에 저장함

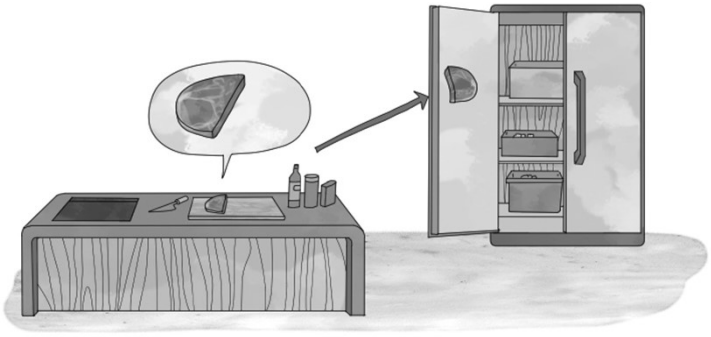


그림 9-28 **도마에서 손질 중인 고깃덩어리 일부를 다시 보관 창고에 가져다 놓는 경우**

40 / 65

03. 메모리 관리

3. 스왑

- 사용 중이 아닌 모듈 A를 하드디스크가 아닌 다른 공간에 보관해야 함
- **스왑 영역** swap area
  - 메모리가 모자라서 쫓겨난 프로세스를 모아두는, 저장 장치의 특별한 공간
  - 스왑 인 ↔ 스왑 아웃

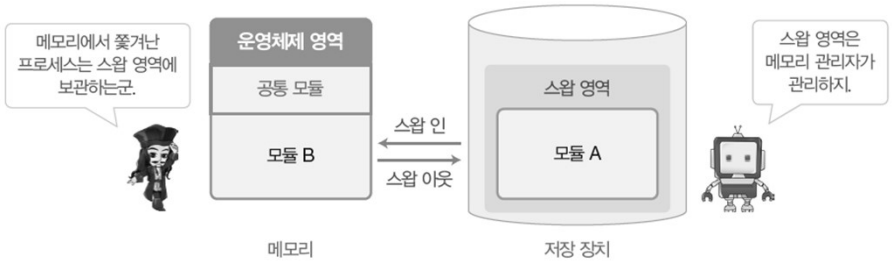


그림 9-29 **스왑 영역**

41 / 65

03. 메모리 관리

3. 스왑

- 스왑 영역은 저장 장치에 있지만 메모리 관리자가 관리함
  - ↔ 원래 저장 장치는 저장 장치 관리자가 관리함
- 스왑 영역의 크기도 메모리의 크기로 인식됨
  - 사용할 수 있는 메모리 크기 = 실제 메모리의 크기 + 스왑 영역의 크기

42 / 65

03. 메모리 관리

4. 가상 메모리 시스템

- 레시피를 개발할 때 주방장은 레스토랑의 주방 사정까지 고려할 수 없음

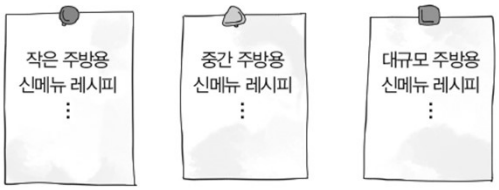


그림 9-30 **주방 크기를 고려한 레시피 개발**

03. 메모리 관리

4. 가상 메모리 시스템

- 프로그램을 만들 때 프로그래머가 사용자의 메모리까지 고려하기 어려움

그림 9-31 **다양한 물리 메모리의 크기**

- 가상 메모리 시스템
  - 메모리 크기에 신경 쓰지 않고 프로그램을 만들도록 지원하는 메모리 관리 시스템
  - 물리 메모리의 크기와 상관없이 커다란 논리주소 공간을 제공함

03. 메모리 관리

4. 가상 메모리 시스템

▪ 가상 메모리의 구성

- 프로세스 입장에서는 시스템이 허용하는 최대 크기의 메모리를 혼자서 독차지하는 것처럼 보임
- 메모리 관리자 입장에서는 메모리의 부족한 부분을 스왑 영역으로 보충
- 현대의 모든 컴퓨터는 가상 메모리를 지원함

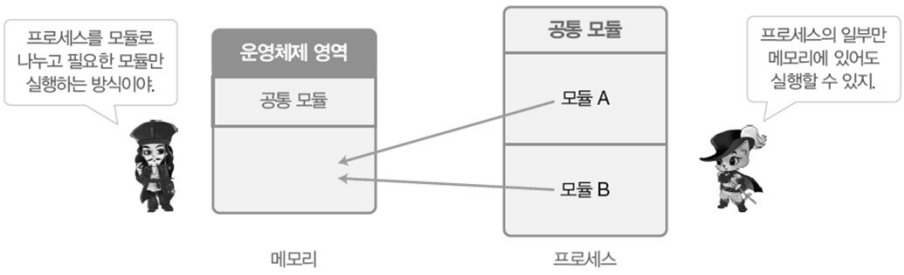


그림 9-27 **메모리 오버레이**

03. 메모리 관리

4. 가상 메모리 시스템

▪ 최대 절전 모드

- 컴퓨터를 잠시 껐다가 현재 작업 상태를 복구함
  - CPU와 메모리의 전력 공급이 끊겨 메모리 내용이 모두 사라지기 전에 메모리에 있는 데이터를 스왑 영역에 옮김

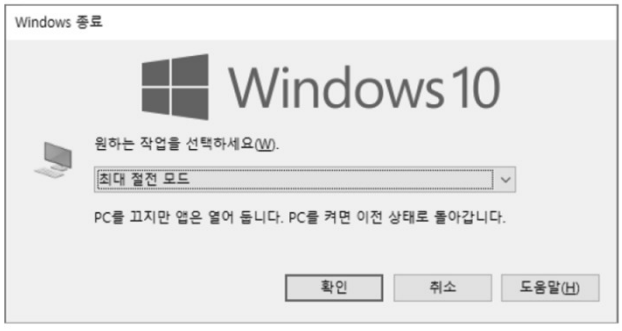


그림 9-33 **최대 절전 모드**

04
저장 장치 관리

04. 저장 장치 관리

1. 파일과 디렉터리

- 운영체제는 사용자가 저장 장치 특정 위치에 파일을 보관하는 것을 막고, 접근 권한을 확인하여 권한이 없는 사용자가 파일에 접근하는 것을 막음
- 윈도우는 NTFS, 유닉스는 아이노드 파일 시스템 사용
- 파일 시스템의 **파일 테이블** file table에 파일 속성이 명시됨

그림 9-34 **저장 장치 관리자와 파일 시스템**

48 / 65

04. 저장 장치 관리

1. 파일과 디렉터리

▪ 파일의 단위

- **블록** block
  - 저장 장치에서 파일을 묶어서 관리하는 일정 크기
  - 메모리의 단위는 바이트이지만, 저장 장치의 단위는 블록

이름	블록 번호
파일 A	1, 3, 9
파일 B	2, 4
파일 C	13
파일 D	12, 15
파일 E	7, 23

(a) 파일 테이블

0	1	2	3	4	5	6	7	8	9
	A	B	A	B			E		A
		D	C		D				
			E						

(b) 저장 장치

그림 9-35 **파일 테이블과 블록 번호**

04. 저장 장치 관리

1. 파일과 디렉터리

▪ 파일의 종류

- 실행 파일
  - 운영체제가 메모리로 가져와 CPU를 이용하여 작업하는 파일
  - 사용자가 실행 파일의 실행을 요청했을 때 프로세스가 됨
- 데이터 파일
  - 실행 파일이 작업하는 데 필요한 데이터를 모아 놓은 파일

04. 저장 장치 관리

1. 파일과 디렉터리

▪ 파일의 종류

- 확장자
  - 파일에 확장자를 붙이면 파일의 종류를 알 수 있음

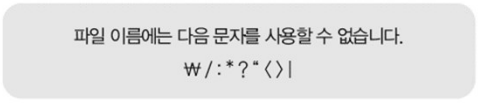


그림 9-36 **윈도우의 파일 이름 오류 메시지**

51 / 65

04. 저장 장치 관리

1. 파일과 디렉터리

▪ 파일 이름 규칙

- **'파일 이름.확장자'**의 형태로 구성됨
  - 확장자는 세 글자 또는 네 글자
- 파일 이름에 마침표(.)를 여러 번 사용 가능
  - 마지막 마침표 다음의 글자를 확장자로 인식
- 파일 이름 유의 사항
  - 파일 이름은 경로를 포함하여 최대 255자
  - 영문자, 숫자, 붙임표(-), 밑줄(_), 마침표(.) 사용 가능
  - 유닉스와 윈도우의 파일 이름 규칙 차이(특수문자 사용, 대소문자 구분)

52 / 65

04. 저장 장치 관리

1. 파일과 디렉터리

- 연결 프로그램
  - 윈도우에서 데이터 파일을 더블클릭했을 때 실행되는 응용 프로그램
  - 데이터 파일이 필요로 하는 응용 프로그램을 운영체제가 실행함
  - ↔ 실행 파일을 더블클릭하면 프로세스가 됨

그림 9-37 **동영상 파일 실행**

04. 저장 장치 관리

1. 파일과 디렉터리

- **디렉터리** directory
  - 서로 관련 있는 파일을 모아 놓은 것 (=폴더)
  - 디렉터리의 계층 구조
    - 루트 디렉터리: 최상위에 있는 디렉터리

그림 9-38 **디렉터리**

04. 저장 장치 관리

1. 파일과 디렉터리

- **디렉터리** directory
  - 그림에서 역슬래시(\ )는 루트 디렉터리

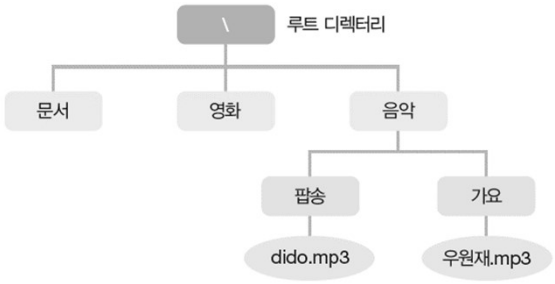


그림 9-39 **디렉터리 계층 구조**

55 / 65

04. 저장 장치 관리

2. 파일 시스템

- **파티션** partition
  - 논리적으로 구분된 저장 장치 영역
  - 하나의 논리적 저장 장치에 하나의 파일 시스템이 탑재됨

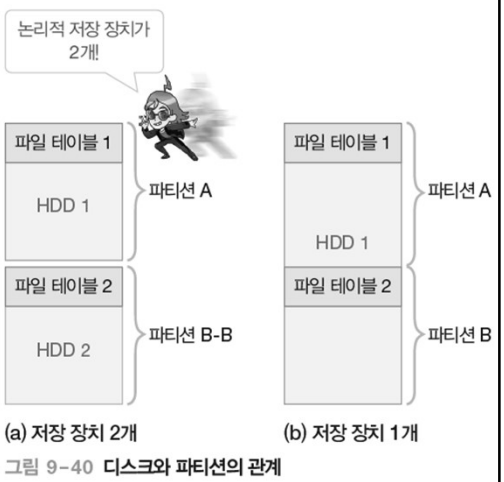


그림 9-40 **디스크와 파티션의 관계**

56 / 65

04. 저장 장치 관리

2. 파일 시스템

- **파티션** partition
  - 유닉스는 여러 개의 파티션을 통합함
  - 마운트
    - 저장 장치의 개수나 파티션 개수에 관계 없이 하나의 파일 시스템만 가짐

(c) 저장 장치 2개

57 / 65

04. 저장 장치 관리

2. 파일 시스템

- **빠른 포맷과 느린 포맷**
  - **디스크 포맷** disk formatting
    - 저장 장치의 파일 시스템을 초기화하는 작업
    - 파일 테이블이 없는 저장 장치에는 파일 테이블을 새로 탑재
    - 기존의 파일 테이블이 있는 경우,
      파일 테이블을 초기화하여 아무것도 저장되지 않은 처음 상태로 만듦
  - 빠른 포맷: 데이터는 그대로 둔 채 파일 테이블만 초기화함
  - 느린 포맷: 파일 테이블을 초기화 & 블록의 모든 데이터를 0으로 만듦

58 / 65

04. 저장 장치 관리

2. 파일 시스템

- **저장 장치의 블록 크기**
  - 블록 크기가 클 때
    - 주소의 수 감소
    - 블록 크기보다 작은 파일을 저장했을 때 공간이 낭비됨

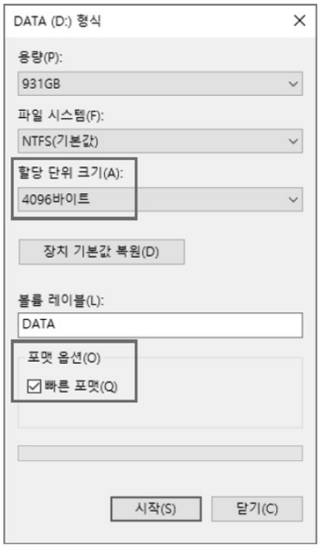


그림 9-41 **윈도우 NTFS 파일 시스템의 포맷 화면**

04. 저장 장치 관리

2. 파일 시스템

- **저장 장치의 블록 크기**
  - 디스크 내부 단편화
    - 전체 파일 크기는 62.4GB
    - 파일에 할당된 디스크의 크기 54.5GB
    - 블록보다 작은 파일을 저장하여 8GB가 낭비됨

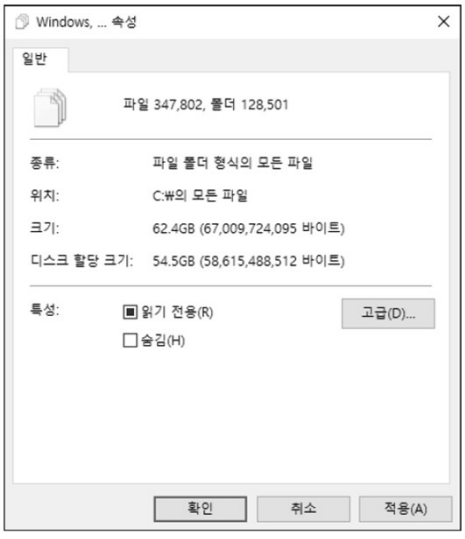


그림 9-42 **디스크 내부 단편화**

60 / 65

04. 저장 장치 관리

2. 파일 시스템

- **파일 테이블의 데이터 저장 방식**
  - 파일 제어 테이블
    - 파일 정보, 파일의 시작 블록 정보가 있음
  - 파일 할당 테이블(FAT)
    - 데이터가 연결된 블록의 번호가 있음

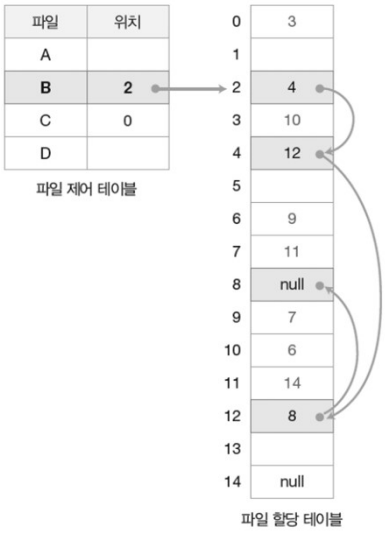


그림 9-43 FAT 파일 시스템의 구조

04. 저장 장치 관리

2. 파일 시스템

- **빈 공간 관리**
  - 디지털 포렌식
    - 디지털 기기의 데이터를 수집·추출하여 범죄 단서와 증거를 찾아내는 기술
  - **빈 공간 리스트** free block list
    - 파일 시스템이 저장 장치 공간을 효율적으로 관리하기 위하여 빈 블록 정보만 모아 놓은 것

그림 9-44 파일 B를 삭제하기 전후 빈 공간 리스트의 변화

63 / 65

04. 저장 장치 관리

2. 파일 시스템

- **빈 공간 관리**
  - 파일을 삭제하거나 빠른 포맷을 할 때 블록의 내용이 지워지지 않아도 빈 공간 리스트에 삽입되면 파일이 삭제된 것으로 간주함
  - 새로운 데이터를 빈 공간 리스트의 맨 앞 블록부터 할당함
- **삭제된 파일을 복구할 수 있는 이유**
  - 파일 삭제 시 파일 내용이 사라지지 않고 파일 테이블의 정보만 삭제됨
  - 악의적으로 파일 복구 프로그램 실행 시 삭제했던 파일을 되살릴 수 있음
  - 파일이 즉시 클라우드로 전송될 수 있음

64 / 65

Thank You!

한빛아카데미 Hanbit Academy, Inc.