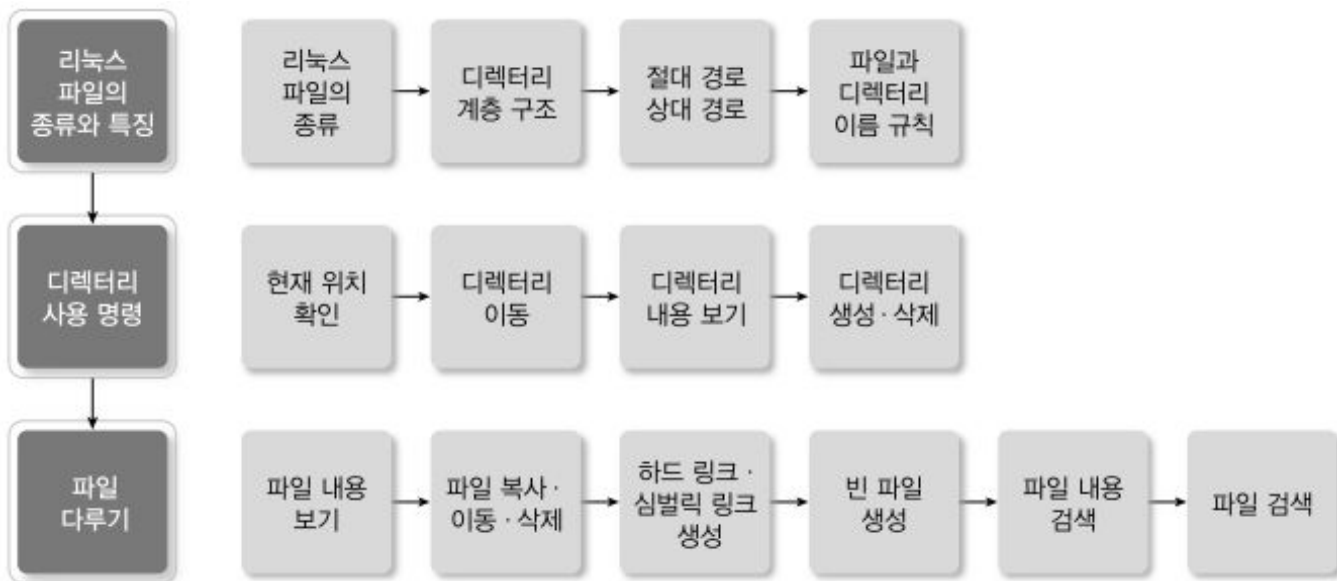











# 시스템프로그래밍 : 명령어 (Cont.)



# 파일의 종류와 특징

디렉터리	기능
dev 	장치 파일이 담긴 디렉터리이다.
home 	사용자 홈 디렉터리가 생성되는 디렉터리이다.
media	시디롬이나 USB 같은 외부 장치를 연결(마운트라고 함)하는 디렉터리이다.
opt	추가 패키지가 설치되는 디렉터리이다.
root 	root 계정의 홈 디렉터리이다. 루트(/) 디렉터리와 다른 것이므로 혼동하지 않도록 한다.
sys 	리눅스 커널과 관련된 파일이 있는 디렉터리이다.
usr 	기본 실행 파일과 라이브러리 파일, 헤더 파일 등 많은 파일이 있다. 참고로 usr은 Unix System Resource의 약자이다.
bin 	실행 파일(명령)을 가지고 있다.
boot 	부팅에 필요한 커널 파일을 가지고 있다.
etc 	리눅스 설정을 위한 각종 파일을 가지고 있다.
lost+found	파일 시스템에 문제가 발생하여 복구할 경우, 문제가 되는 파일이 저장되는 디렉터리로 보통은 비어있다.
mnt	파일 시스템을 임시로 마운팅 하는 디렉터리이다.
proc	프로세스 정보 등 커널 관련 정보가 저장되는 디렉터리이다.
run	실행 중인 서비스와 관련된 파일이 저장된다.
srv	FTP나 Web 등 시스템에서 제공하는 서비스의 데이터가 저장된다.
tmp	시스템 사용 중에 발생하는 임시 데이터가 저장된다. 이 디렉터리에 있는 파일들은 재부팅 하면 모두 삭제된다.
var 	시스템 운영 중에 발생하는 데이터나 로그 등이 저장되는 디렉터리이다.

# 파일의 종류와 특징

## ■ 경로명

- 파일 시스템에서 디렉터리 계층 구조에 있는 특정 파일이나 디렉터리의 위치 표시
- 경로명에서 각 경로를 구분하는 구분자로 슬래시(/)를 사용
- 경로명에서 가장 앞에 있는 /는 루트 디렉터를 뜻하지만 경로명 중간에 있는 /는 구분자
- 예: `/usr/bin/ls`에서 맨 앞의 /는 루트 디렉터를 의미하고, 중간에 있는 / 두 개는 디렉터리 이름과 파일 이름을 구분하는 구분자

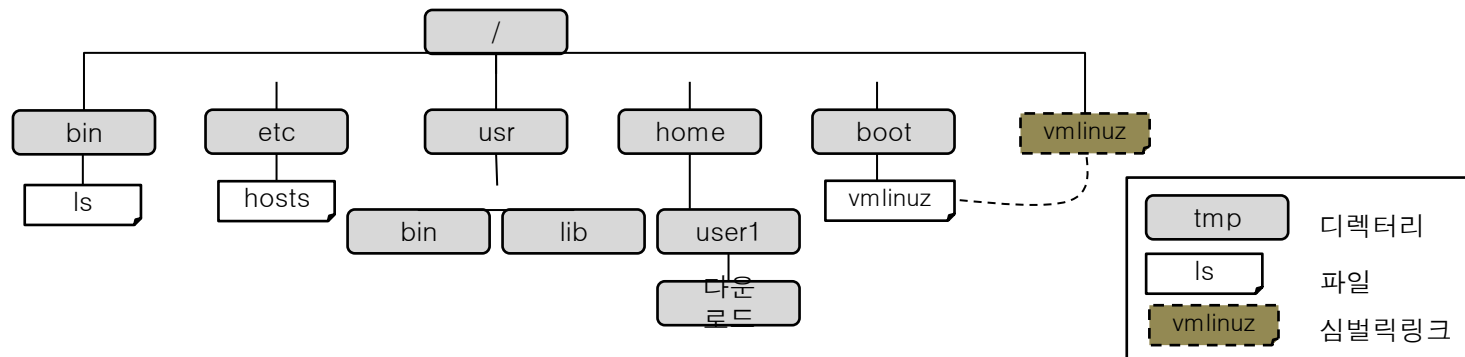
## ■ 절대 경로명

- 항상 루트(/) 디렉터리부터 시작
- 반드시 /로 시작한다.
- / 디렉터리부터 시작하여 특정 파일이나 디렉터리의 위치까지 이동하면서 거치게 되는 모든 중간 디렉터리의 이름을 표시
- 특정 위치를 가리키는 절대 경로명은 항상 동일

## ■ 상대 경로명

- 현재 디렉터를 기준으로 시작
- / 이외의 문자로 시작
- 현재 디렉터를 기준으로 서브 디렉터리로 내려가면 그냥 서브 디렉터리의 이름을 추가
- 현재 디렉터를 기준으로 상위 디렉터리로 가려면 `..`을 추가
- 상대 경로명은 현재 디렉터리가 어디냐에 따라 달라짐

# 파일의 종류와 특징



디렉터리 계층구조 예

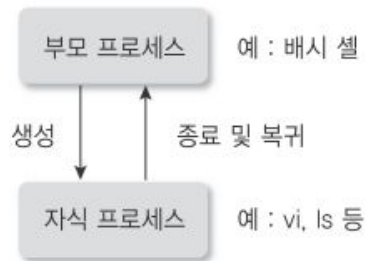
## ■ 현재 디렉터리가 user1일 때

- user1의 절대 경로명: /home/user1
- user1 아래 '다운로드'의 절대 경로명: /home/user1/다운로드
- '다운로드'의 상대 경로명: 다운로드 또는 ./다운로드
- hosts 파일의 상대 경로명: ../../etc/hosts

디렉터리/파일명	절대 경로	상대 경로
/		
home		
tmp		
lib		
ls		

# 프로세스의 개념

- 프로세스: 현재 시스템에서 실행 중인 프로그램
- 프로세스의 부모-자식 관계
  - 프로세스는 부모-자식 관계를 가지고 있음
  - 필요에 따라 부모 프로세스(parent process)는 자식 프로세스(child process)를 생성하고, 자식 프로세스는 또 다른 자식 프로세스 생성 가능
  - 부팅할 때 스케줄러가 실행한 프로세스인 **systemd**와 **kthreadd** 프로세스를 제외하면 모든 프로세스는 부모 프로세스를 가지고 있음
  - 자식 프로세스는 할 일이 끝나면 부모 프로세스에 결과를 돌려주고 종료
- 프로세스의 번호
  - 각 프로세스는 고유한 번호를 가지고 있는데 이것이 PID
- 프로세스의 종류
  - 데몬 프로세스
    - 특정 서비스를 제공하기 위해 존재하며 리눅스 커널에 의해 실행
  - 고아 프로세스
    - 자식 프로세스가 아직 실행 중인데 부모 프로세스가 먼저 종료된 자식 프로세스는 고아(orphan) 프로세스
    - 1번 프로세스가 고아 프로세스의 새로운 부모 프로세스가 되어 고아 프로세스의 작업 종료 지원
  - 좀비 프로세스
    - 자식 프로세스가 실행을 종료했는데도 프로세스 테이블 목록에 남아 있는 경우
    - 좀비 프로세스는 프로세스 목록에 **defunct** 프로세스라고 나오기도함
    - 좀비 프로세스가 증가하면 프로세스 테이블의 용량이 부족해서 일반 프로세스가 실행되지 않을 수도 있음



부모 프로세스와  
자식 프로세스의 관계

# 프로세스 관리 명령

## ■ 프로세스 목록 보기

◦ 현재 실행 중인 프로세스의 목록을 보는 명령: **ps**

■ 유닉스(SVR4) 옵션 : 묶어서 사용할 수 있고, 붙임표로 시작한다(예 : **-ef**).

■ BSD 옵션 : 묶어서 사용할 수 있고, 붙임표로 시작하지 않는다(예 : **aux**).

### ps

**기능** 현재 실행 중인 프로세스에 대한 정보를 출력한다.

**형식** ps 옵션

**옵션**

〈유닉스 옵션〉	-e : 시스템에서 실행 중인 모든 프로세스의 정보를 출력한다. -f : 프로세스에 대한 자세한 정보를 출력한다. -u uid : 특정 사용자에게 대한 모든 프로세스의 정보를 출력한다. -p pid : pid로 지정한 특정 프로세스의 정보를 출력한다.
〈BSD 옵션〉	a : 터미널에서 실행한 프로세스의 정보를 출력한다. u : 프로세스 소유자의 이름, CPU 사용량, 메모리 사용량 등 상세 정보를 출력한다. x : 시스템에서 실행 중인 모든 프로세스의 정보를 출력한다.
〈GNU 옵션〉	--pid PID 목록 : 목록으로 지정한 특정 PID 정보를 출력한다.

**사용 예** ps            ps -ef            ps aux

# 프로세스 관리 명령

## ■ 현재 단말기의 프로세스 목록 출력하기 : ps

- ps 명령을 옵션 없이 사용하면 현재 셸이나 터미널에서 실행한 사용자 프로세스에 대한 정보를 출력

```
user1@myubuntu:~$ ps
  PID TTY          TIME CMD
 5501 pts/1        00:00:00 bash
 6162 pts/1        00:00:00 ps
user1@myubuntu:~$
```

```
user1@myubuntu:~$ ps -f
UID          PID  PPID  C STIME TTY          TIME CMD
user1        5501  5500  0 00:32 pts/1        00:00:00 -bash
user1        6163  5501  0 04:33 pts/1        00:00:00 ps -f
user1@myubuntu:~$
```

## ■ 프로세스의 상세 정보 출력하기 : -f 옵션

- 프로세스의 상세한 정보를 출력: PPID와 터미널 번호, 시작 시간 등

ps -f의 출력 정보

항목	의미	항목	의미
UID	프로세스를 실행한 사용자 ID	STIME	프로세스의 시작 날짜나 시간
PID	프로세스 번호	TTY	프로세스가 실행된 터미널의 종류와 번호
PPID	부모 프로세스 번호	TIME	프로세스 실행 시간
C	CPU 사용량(% 값)	CMD	실행되고 있는 프로그램 이름(명령)



# 여러가지 도구

- ls | grep
- awk
- find
- redirection
  - stdin - standard input (0) : 입력
  - stdout - standard output (1) : 정상 결과 출력
  - stderr - standard error (2) : 오류 내용 출력 (출력은 동일하나, 정상 출력과 오류출력을 구분)
  - > : 결과를 파일로 저장 (덮어쓰기)
  - >> : 결과를 파일로 저장 (덮어쓰지 않고 기존 파일에 추가)
  - < : 입력(파일, 텍스트 등)을 명령어에 실행 (덮어쓰기)
  - << : 입력을 명령어에 실행 (덮어쓰지 않고 기존 파일에 추가)
  - ls -al > dummy.t
  - ls -al 1> dummy2.t

# 기타명령어

- **Data 복사**
  - cp, tar, cpio, dd
- **연결**
  - link : hard, soft
- **문서편집**
  - vi, sed (ed), tr, (touch, cat)
- **정보**
  - stat, man, w, df, uname, history
- **찾기**
  - find, grep
- **관리**
  - service, systemctl