

Computer Programming

Lecture 1

이윤진
서울대학교
2007.12.20.

Slide Credits

- 엄현상 교수님
 - 서울대학교 컴퓨터공학부
 - Computer Programming, 2007 봄학기

순서

- 강의 소개
 - 강의 목표
 - 강의 개요
 - 수업 진행 방법 및 평가
- UNIX/LINUX
 - 기초
 - 주요 기능
 - 파일 시스템

강의 목표

- Unix/Linux 중심의 프로그래밍
- 컴퓨터 프로그램과 시스템의 상호작용 이해
- 복잡한 문제의 C 프로그래밍
- 기초적인 C++, Java, 및 윈도우 프로그래밍

강의 개요

- Unix/Linux
- Utilities and Editors
- Batch and Shell Programming
- C Compiler and Linker
- Modularity and Abstraction in C
- C Pointers
- Memory Management in C
- Libraries
- C++, Java, and Windows Programming

수업 진행 방법 및 평가

- 수업 진행 방법
 - 일주일: 강의 1회, 실습 1회
 - 강의: 수업 1시간 15분, 휴식 15분으로 두 번 진행
 - 실습: 조교 강의 + 실습
 - 강의자료: <http://mrl.snu.ac.kr/~yunjin/winter2007/winter2007.html>
- 평가
 - 출석 10% (5번 이상 결석 시 0점, Quiz 포함)
 - 과제 20% (2일 이내 20% 감점, 이후 0점)
 - 중간고사 30%
 - 기말고사 40%
- 그 외
 - 강의 홈페이지 참고

Unix/Linux (1)

순서

- 기초
 - OS
 - UNIX OS
- 주요 기능
 - UNIX Kernel
 - Program execution/Memory layout of C program/System calls
- UNIX 시작하기
- Q&A

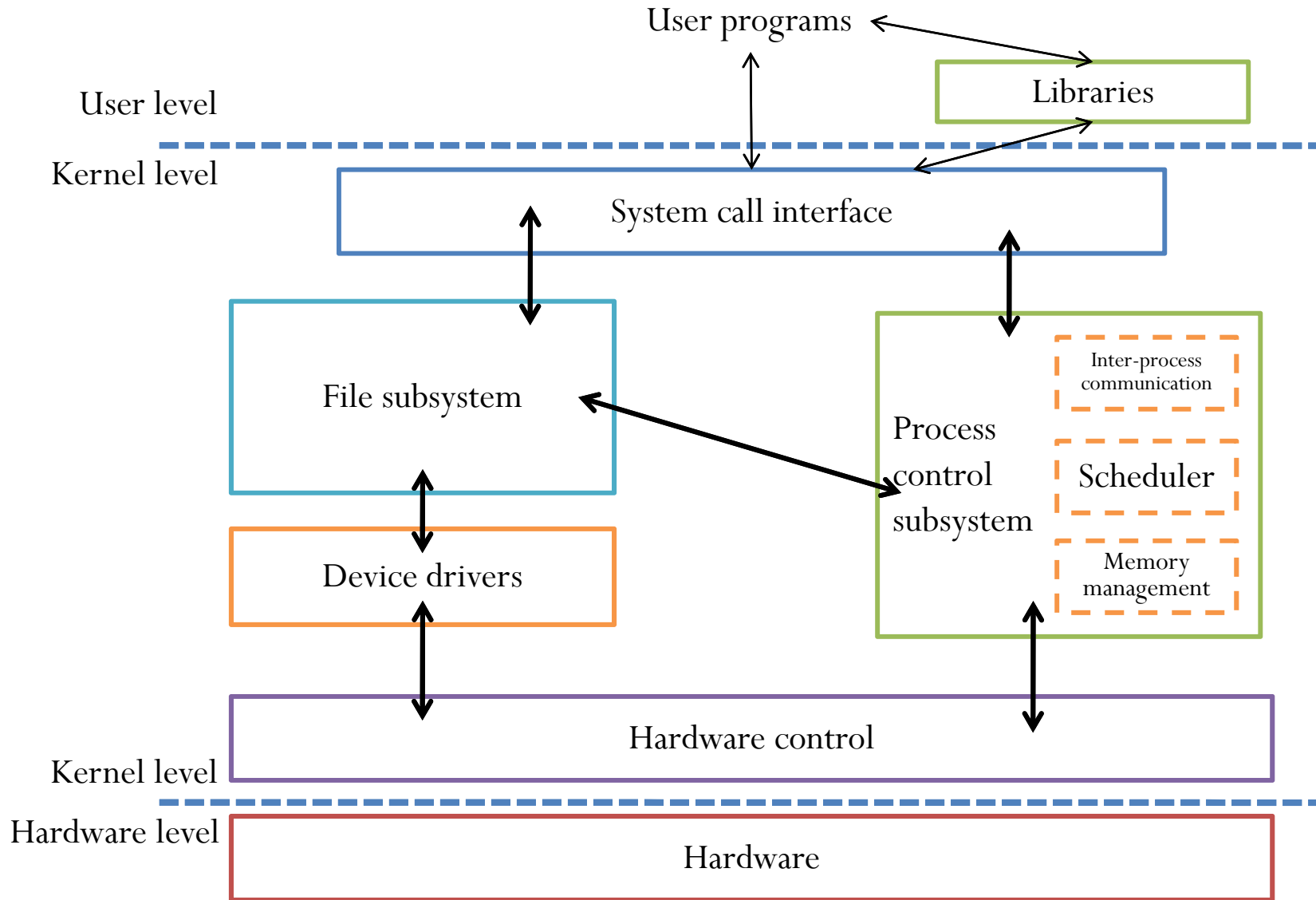
OS (Operating System)

- S/W that Controls the Computer or Manages Its Resources
 - Control User Programs
 - CPU(s)
 - Memory
 - Devices
- Act as the Interface between the Users and the Computer
- Provide Services for Programs It Runs

Unix: OS

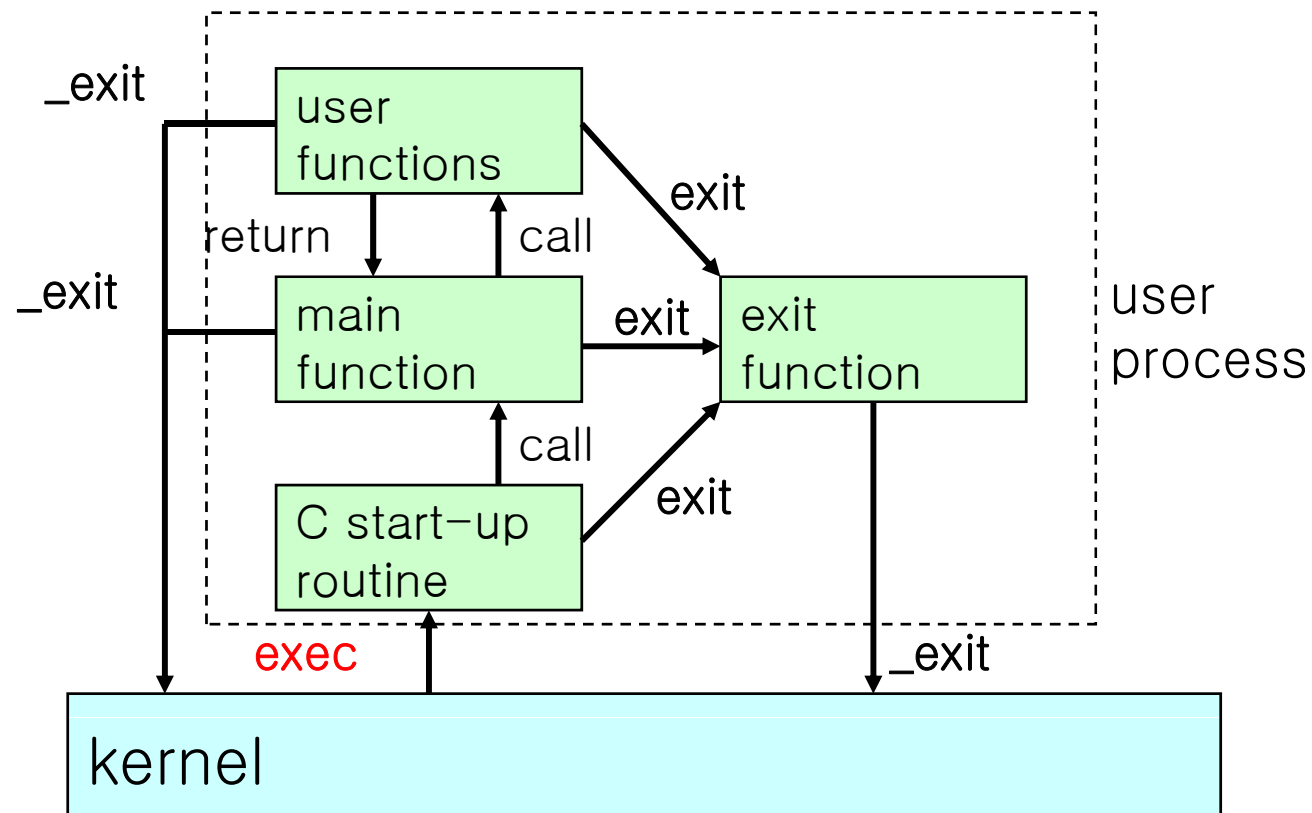
- Made Up of the Kernel (Including the File System) and Shell (Command-Line Interface)
- Written in C
 - Portability
- Having Various Versions
 - Linux
 - Distributed along with Technical Support and Training from Several Vendors

UNIX Kernel

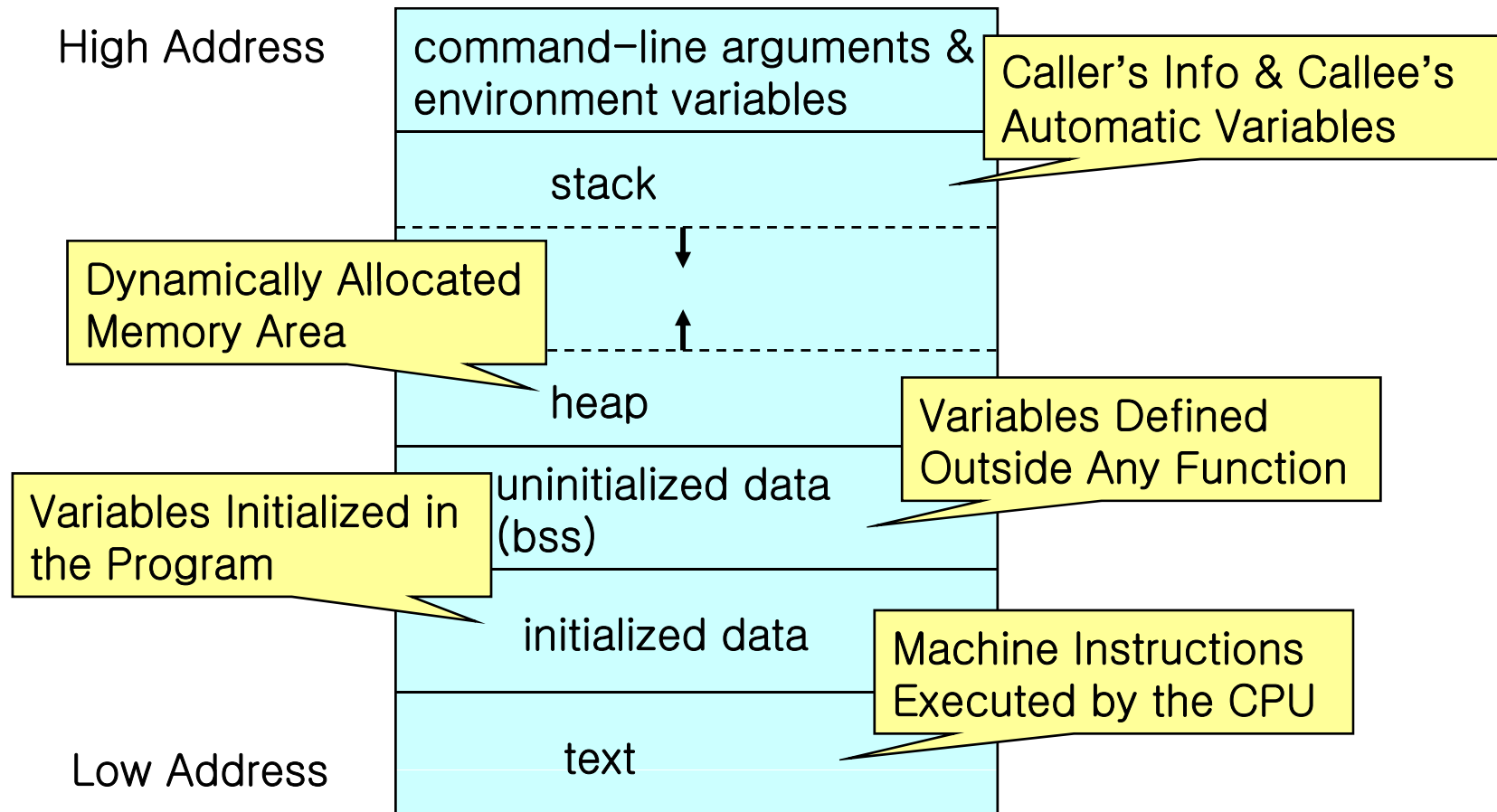


Program Execution

- Program: executable residing in a disk file
- Process: executable instance of a program

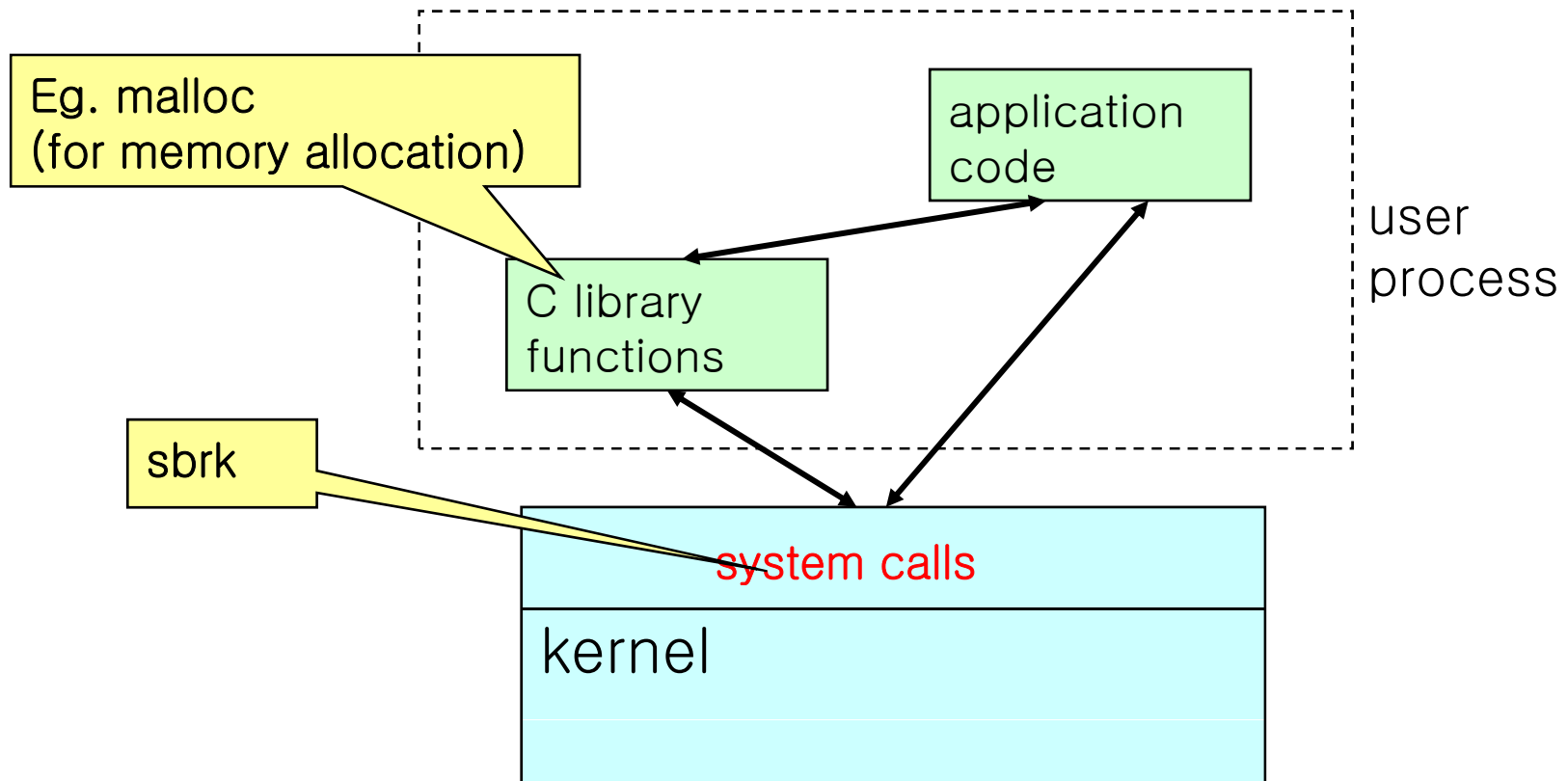


Memory Layout of C Program



System Calls

- Entry Points Directly into the Kernel



UNIX 시작하기

- Logging in

```
martini:~> more /etc/passwd
      : (omitted)
net001:x:3001:3001::/home/net/net001:/bin/bash
      : (omitted)
```

- Online Documentation

```
martini:~> man more
      : (omitted)
NAME
  more - 문자속성을 살린 파일 보기 프로그램
      : (omitted)
```

UNIX 시작하기

- File system

```
martini:~> ls /  
bin boot cdrom ...  
martini:~> pwd  
/csehome/net001
```

- Process control

```
martini:~> ps  
  PID TTY          TIME CMD  
10742 pts/0    00:00:00 tcsh  
10977 pts/0    00:00:00 ps
```

Unix/Linux (2)

순서

- File System 기초
- File System 특징
 - Code Example
- Q&A

File System 기초

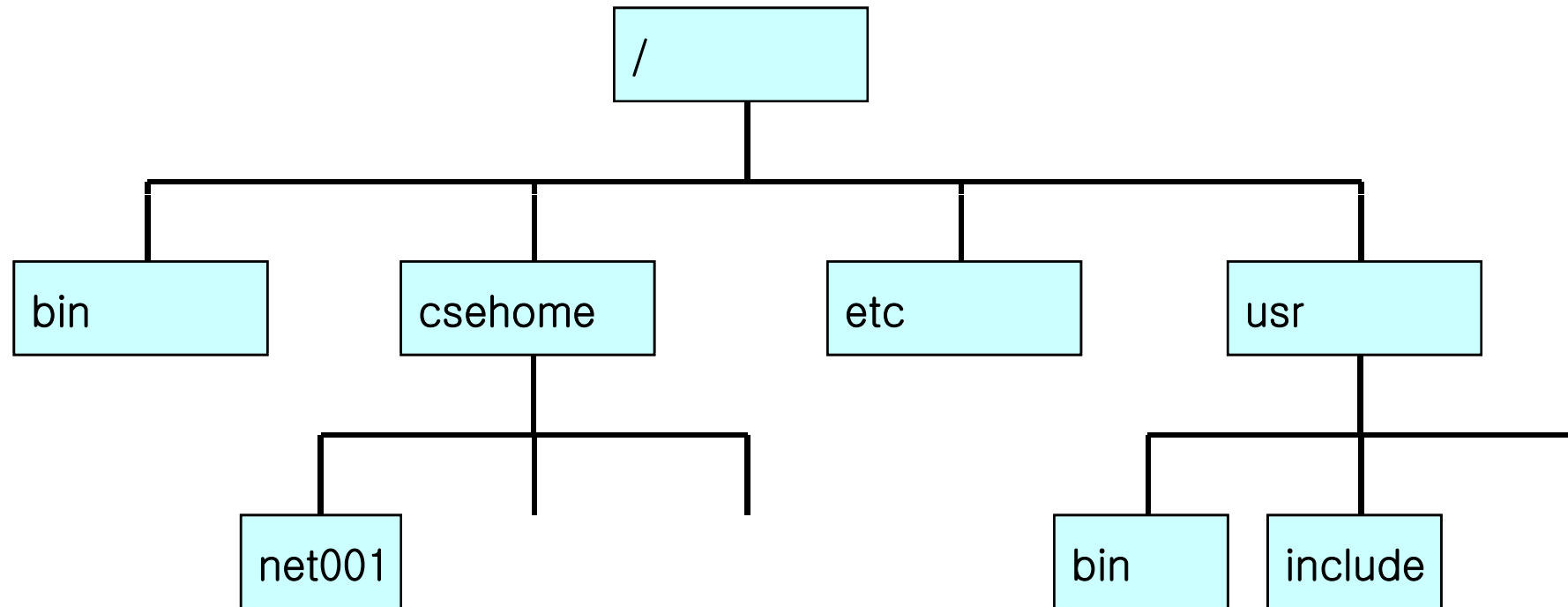
- File
 - Set of Data Stored on the System's Disk
 - Identified by a Filename
 - Uppercase/Lowercase Letters, Numbers, Periods, Underscores, and Hyphens
 - Filename Extensions
 - Not Handled Separately
- File System
 - Abstraction Used by the Kernel to Represent and Organize the System's Storage Resources

File System 특징

- Hierarchical Structure
- Consistent Treatment of File Data
- Ability to Create and Delete Files
- Dynamic Growth of Files
- Protection of File Data
- Treatment of Peripheral Devices as Files

Hierarchical Structure

- Directory
 - Simulated File Folder on Disk



- Pathname: e.g., /usr/bin, /usr/include

Consistent Treatment of File Data

- Treatment of File Data as an Unformatted Stream of Bytes
 - Syntax of Accessing the Data in a File
 - Identical for All Programs
 - Semantics of the Data
 - Imposed by the Program

Ability to Create and Delete Files

- open, creat, & close

```
int open(char *name, int flags);  
/* flags: O_RDONLY, O_WRONLY, or O_RDWR defined <fcntl.h> */  
int creat(char *name, int perms);  
/* perms: usually, three digit octal number, e.g., 0666 */  
int close(int fd);  
/* fd: file descriptor */
```

- File Descriptor

- Integer Used to Refer to an Open File

- read & write

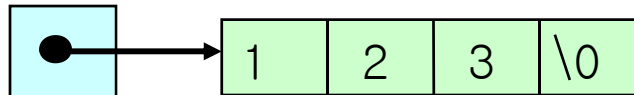
```
int read(int fd, char *buf, int n);  
int write(int fd, char *buf, int n);  
/* fd: file descriptor */
```

C Programming의 기초

- Character String or String Constant
 - Sequence of characters in double quotes

```
char *charstr = "123";
```

charstr:



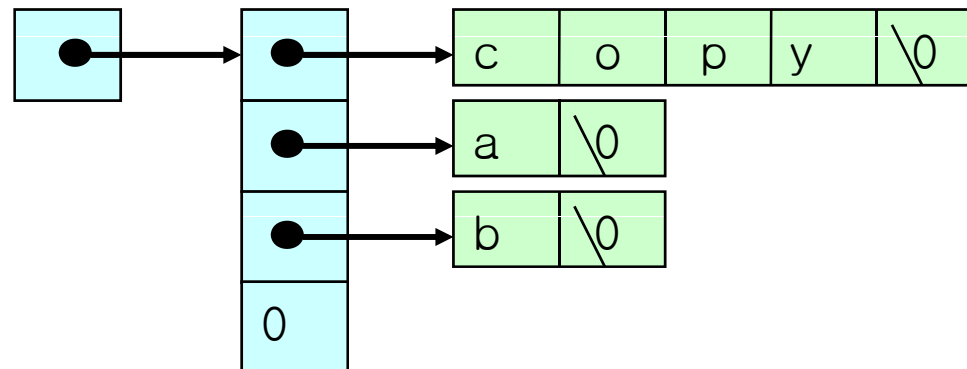
- Command-Line Arguments

```
main(int argc, char *argv[]) {
```

```
...
```

```
martini:~ > copy a b
```

argv:



Example: Program to Copy a File

```
#include <fcntl.h>
char buffer[2048];
main(int argc, char *argv[])
{
    int fdold, fdnew, count;
    if(argc != 3) {
        printf("need 2 arguments for copy program\n");
        exit(1);
    }
    if((fdold = open(argv[1], O_RDONLY)) == -1) {
        printf("cannot open file %s\n", argv[1]);
        exit(1);
    }
    if((fdnew = creat(argv[2], 0666)) == -1) {
        printf("cannot create file %s\n", argv[2]);
        exit(1);
    }
    while((count = read(fdold, buffer, sizeof(buffer))) > 0)
        write(fdnew, buffer, count);
    exit(0);
}
```

Example (계속)

- Compilation & Execution

```
martini:~> gcc -o copy copy.c
martini:~> ls -al
drwxr-xr-x  2 net001 cseusers 216 2005-03-07 15:51 .
drwxr-xr-x  3 net001 cseusers  72 2005-03-07 13:10 ..
-rwxr-xr-x  1 net001 cseusers 15K 2005-03-07 14:44 copy
-rw-r--r--  1 net001 cseusers 588 2005-03-07 14:44 copy.c
martini:~> ./copy copy.c copy1.c
```

- Relevant Command

```
martini:~> chmod g+w copy.c
martini:~> ls -al copy.c
-rw-rw-r--  1 net001 cseusers 588 2005-03-07 14:44 copy.c
```

Example (계속)

- `chmod who(=,+,-)permissions filename`
- Who
 - u: The user who owns the file
 - g: The group the file belongs to
 - o: The other users
 - a: all of the above
- Permissions
 - r: Permission to **r**ead the file (4)
 - w: Permission to **w**rite (or delete) the file (2)
 - x: Permission to **e**xecute the file (1)
- `chmod 754 copy`

Dynamic Growth of Files

- Flexible File System
- Internal Tree Structure
 - inode
 - Information about each file in a structure that the kernel maintains
 - Owner of the file
 - Size of the file
 - Device that the file is located on
 - Pointers to where the actual data blocks for the file are located on disk

Dynamic Growth of Files (계속)

```
shell [ jin{51} ~] stat copy.c
```

```
File: `copy.c'
```

```
Size: 366      Blocks: 2      IO Block: 4096  regular file
```

```
Device: 14h/20d Inode: 115414824  Links: 1
```

```
Access: (0666/-rw-rw-rw-) Uid: ( 1318/   jin)  Gid: ( 203/   cse)
```

```
Access: 2007-12-18 21:12:54.725645004 +0900
```

```
Modify: 2007-12-18 21:12:54.092035374 +0900
```

```
Change: 2007-12-18 21:12:54.092035374 +0900
```

Treatment of Peripheral Devices as Files

- Current Terminal as a File

```
martini:~ > ./copy /dev/tty term  
Input[Control-D]  
martini:~ > more term
```

- Result

```
Input
```

```
martini:~ > ./copy /dev/tty /dev/tty  
Input[Enter]
```

- Result

```
Input
```