File & Directories

System Programming

실습

파일 & 디렉토리 system call

❑ 책자(이론) 예제 코드 컴파일 및 실행 해보기

- Prog. 4.3
  - lstat함수를 stat로 바꿔서 컴파일 및 실행해보기
- Prog. 4.8
- Prog. 4.9
- Prog. 4.12

파일 & 디렉토리 system call

❑ Prog. 4. 3 실행

Sample output from Figure 4.3 is

```
$ ./a.out /etc/passwd /etc /dev/log /dev/tty \
> /var/lib/oprofile/opd_pipe /dev/sr0 /dev/cdrom
/etc/passwd: regular
/etc: directory
/dev/log: socket
/dev/tty: character special
/var/lib/oprofile/opd_pipe: fifo
/dev/sr0: block special
/dev/cdrom: symbolic link
```

파일 & 디렉토리 system call

❑ Prog. 4. 8 실행

```
$ ls -l a.out
-rwxrwxr-x  1 sar          15945 Nov 30 12:10 a.out
$ ./a.out a.out
read access OK
open for reading OK
$ ls -l /etc/shadow
-r--------  1 root          1315 Jul 17  2002 /etc/shadow
$ ./a.out /etc/shadow
access error for /etc/shadow: Permission denied
open error for /etc/shadow: Permission denied
$ su                                    become superuser
Password:                               enter superuser password
# chown root a.out                      change file's user ID to root
# chmod u+s a.out                       and turn on set-user-ID bit
# ls -l a.out                           check owner and SUID bit
-rwsrwxr-x  1 root         15945 Nov 30 12:10 a.out
# exit                                  go back to normal user
$ ./a.out /etc/shadow
access error for /etc/shadow: Permission denied
open for reading OK
```

파일 & 디렉토리 system call

❑ Prog. 4. 9 실행

```
$ umask                                  first print the current file mode creation mask
002
$ ./a.out
$ ls -l foo bar
-rw-------  1 sar                         0 Dec  7 21:20 bar
-rw-rw-rw-  1 sar                         0 Dec  7 21:20 foo
$ umask                                  see if the file mode creation mask changed
002
```

실습

파일 & 디렉토리 system call

❑ Prog. 4. 12 실행

```
$ ls -l foo bar
-rw-r--r--  1 sar              0 Dec  7 21:20 bar
-rw-rwSrw-  1 sar              0 Dec  7 21:20 foo
```

System Call & Standard Library Functions

❑ **파일의 정보를 관리하는 시스템 호출/ 표준 라이브러리 함수**

Functions	Description
umask	파일 생성 마스크를 설정한다.
access	파일에 대한 사용자의 접근 권한을 확인한다.
chmod/fchmod	파일에 대한 접근 권한을 변경한다.
chown/fchown	파일의 소유주와 그룹을 변경한다.
link	파일의 새로운 이름을 생성한다. (hard-link)
rename	파일의 이름이나 위치를 변경한다.
symlink	파일의 새로운 이름을 생성한다. (soft-link, symbolic link)
readlink	심볼형 링크의 값(실제 내용)을 읽어온다.
stat/fstat	파일의 상태 정보를 가져온다.

uamsk

```
#include <sys/types.h>
#include <sys/stat.h>

mode_t umask(mode_t mask);
```

mask	파일에 대한 접근 권한으로 mask로 지정한 항목은 새로운 파일 생성 시 적용되지 않는다.
반환값	umask의 실행은 항상 성공하며 기존의 mask 값을 반환한다.

```
mode_t oldmask;
oldmask = umask(037);
filedes = open("data.txt", O_CREAT, 0777);    /* 0777은 8진수 777을 의미 */
```

파일 생성 시 적용하는 초기 접근 권한	(0777) 111 111 111
umask로 제한한 접근 권한	& (~0037) 111 100 000
실제로 적용되는 초기 접근 권한	(0740) 111 100 000

예제 코드

Example Code #1

```
#include<fcntl.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/stat.h>

int main ()
{
        int filedes;
        mode_t oldmask;

        filedes = open ("test1.txt", O_CREAT, 0777);
        close (filedes);

        oldmask = umask (037);
        filedes = open ("test2.txt", O_CREAT, 0777);
        close (filedes);
        return 0;
}
```

access

```
#include <unistd.h>

int access(const char *pathname, int mode);
```

pathname	파일에 대한 경로이름이다.
mode	검사하려는 접근 권한으로 R_OK, W_OK, X_OK, F_OK를 사용할 수 있다.
반환값	access 호출이 성공하면 0을 반환하고, 실패하면 -1을 반환한다.

이름	의미
R_OK	읽기 권한을 검사한다.
W_OK	쓰기 권한을 검사한다.
X_OF	실행 권한을 검사한다. (디렉터리일 경우 탐색 권한을 검사한다.)
F_OK	대상 파일이 존재하는지 검사한다.

예제 코드

Example Code #2

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>

int main ()
{
        char *fname = "test.txt";
        //char *fname = "test1.txt";

        if (access (fname, R_OK) == -1)
        {
                fprintf (stderr, "User cannot read file %s₩n",fname);
                exit (1);
        }
        printf ("%s readable, proceeding₩n", fname);
        return 0;
}
```

chmod, fchmod

```
#include <sys/types.h>
#include <sys/stat.h>

int chmod(const char *path, mode_t mode);
int fchmod(int filedes, mode_t mode);
```

path	파일에 대한 경로 이름이다.
filedes	개방된 파일의 파일 기술자이다.
mode	파일에 새롭게 적용하려는 접근 권한이다.
반환값	호출이 성공하면 0을 반환하고, 실패하면 -1을 반환한다.

```
mode_t mode;
mode = S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH;
chmod("test.txt", mode);
```

```
chmod("test.txt", 0644);
```

예제 코드

Example Code #3

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<sys/stat.h>

int main ()
{
        mode_t mode1, mode2;
        mode1 = S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH;
        mode2 = 0644;

        if (chmod ("test1.txt", mode1) == -1)
                exit(1);
        if (chmod ("test2.txt", mode2) == -1)
                exit(1);
        printf ("Rest of program ... ₩n");
        return 0;
}
```

Q and A