

---

# File I/O

---

## System Programming

# I. System Call & Standard Library Functions

## System Call & Standard Library Functions

### □ 시스템 호출/ 표준 라이브러리 함수

Functions	Description
open	이미 존재하는 파일을 읽기 또는 쓰기, 새로운 파일을 생성하여 오픈
creat	새로운 파일을 생성
close	open 또는 creat로 열린 파일을 닫음
read	열려진 파일로부터 데이터를 읽음
write	열려진 파일에 데이터를 씀
lseek	파일 안에서 읽기/쓰기 포인터를 지정한 바이트 위치로 이동
unlink/remove	파일을 삭제

# I. System Call & Standard Library Functions

## Open

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

```
int open(const char *pathname, int flags, [mode_t mode]);
```

pathname	개방할 파일의 경로 이름을 가지고 있는 문자열의 포인터
----------	--------------------------------

flags	파일의 개방 방식을 지정
-------	---------------

mode	대부분의 경우 생략할 수 있는 값으로 새롭게 생성하는 파일의 초기 접근 권한을 지정
------	--

반환 값	정상적으로 파일을 개방하게 되면 파일 기술자를 반환 파일 개방이 실패할 경우 -1을 반환
------	--

O\_RDONLY

O\_WRONLY

O\_RDWR

읽기만 가능한 상태로 접근

쓰기만 가능한 상태로 접근

읽기, 쓰기 모두 가능한 상태로 접근

# I. System Call & Standard Library Functions

## Open

파일 개방에 사용되는 플래그

O_RDONLY	파일을 읽기 전용으로 개방, 읽기 이외의 다른 작업을 수행할 수 없음
O_WRONLY	파일을 쓰기 전용으로 개방, 쓰기 이외의 다른 작업을 수행할 수 없음
O_RDWR	파일을 읽기와 쓰기가 동시에 가능한 상태로 개방
O_CREAT	지정한 경로의 파일이 존재하지 않으면 새롭게 생성한 후 개방 지정한 경로의 파일이 존재하면 지정한 상태로 개방
O_EXCL	지정한 경로의 파일이 존재하지 않으면 새롭게 생성 지정한 경로의 파일이 존재하면 open 호출을 실패 (O_CREAT 와 함께 사용)
O_APPEND	파일을 개방한 직후에 읽기/쓰기 포인터의 위치를 파일 내용의 마지막 바로 뒤로 이동
O_TRUNC	파일을 개방한 직후에 읽기/쓰기 포인터의 위치를 파일 내용의 첫 부분으로 이동

# I. System Call & Standard Library Functions

## Open

```
filedes = open(filename, O_RDWR);
```

filename으로 지정한 파일을 읽기와 쓰기가 가능한 상태로 개방

```
filedes = open(filename, O_RDONLY | O_CREAT);
```

- filename으로 지정한 파일이 존재할 경우 읽기 전용으로 개방
- 만약 파일이 존재하지 않으면 새롭게 생성한 후 읽기 전용으로 개방  
(파일의 권한을 생략했기 때문에 기본 값으로 초기권한이 설정)

```
filedes = open(filename, O_WRONLY | O_CREAT, 0644);
```

- filename으로 지정한 파일이 존재할 경우 쓰기 전용으로 개방
- 만약 파일이 존재하지 않으면 새롭게 생성한 후 쓰기 전용으로 개방
- 초기 권한이 0644로 설정

```
filedes = open(filename, O_WRONLY | O_CREAT | O_EXCL, 0644);
```

- filename으로 지정한 파일이 존재하면 open의 수행이 실패
- 만약에 파일이 존재하지 않으면 초기권한 0644로 새롭게 파일을 생성 (쓰기 전용으로 개방)

# I. System Call & Standard Library Functions

## Close

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

```
int close(int filedes);
```

filedes            이전에 open이나 creat에 의해 개방된 파일의 File Descriptor

반환값            작업이 성공할 경우 0이 반환되며, 실패할 경우 -1이 반환된다.

# I. System Call & Standard Library Functions

## Creat

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

```
int creat(const char *pathname, mode_t mode);
```

pathname	개방할 파일의 경로 이름을 가지고 있는 문자열의 포인터이다.
mode	새롭게 생성하는 파일의 초기 접근 권한을 지정 creat는 존재하지 않는 파일을 새롭게 생성하는 것이 사용 목적이기 때문에 생략 불가
반환값	정상적으로 파일을 개방하게 되면 파일 기술자를 반환 파일 개방이 실패할 경우 -1을 반환

```
filesdes = creat(pathname, 0644);
```

```
...
```

```
filesdes = open(pathname, O_WRONLY | O_CREAT | O_TRUNC, 0644);
```

# I. System Call & Standard Library Functions

## Read

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

```
ssize_t read(int filedes, void *buf, size_t count);
```

filedes            읽기 작업을 수행할 File Descriptor

buf                파일로부터 읽어 들인 내용을 저장하기 위한 공간  
                    일반적으로 배열을 사용하게 되는데 배열의 데이터 형식은 어느 것이라도 상관 없음

count             읽어 들일 파일 내용의 크기를 지정  
                    바이트 단위로 기술

반환값            파일로부터 읽기 작업이 성공할 경우  
                    1) 읽어 들인 파일 내용의 바이트 크기가 반환 (1 이상의 값)  
                    2) 읽어 들인 내용이 없을 경우 (EOF일 경우) 0을 반환  
                    읽기 작업이 실패할 경우  
                    1) -1을 반환



# I. System Call & Standard Library Functions

## Write

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

```
ssize_t write(int filedes, const void *buf, size_t count);
```

filedes            쓰기 작업을 수행할 File Descriptor

buf                파일로 쓰려고 하는 내용이 저장되어 있는 공간  
                    일반적으로 배열을 사용하게 되면 배열의 데이터 형식은 어느 것이라도 상관없음

count             buf에 있는 데이터 중에 실제로 파일로 저장할 데이터의 크기

반환값            파일로 쓰기가 성공한 데이터의 크기  
                    대부분의 경우 count에서 지정한 값과 동일한 값이 반환  
                    만약 count의 값과 반환값이 다르다면 쓰기 작업이 실패

# I. System Call & Standard Library Functions

## lseek

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

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

```
off_t lseek(int 1filedes, off_t 3offset, int 2whence);
```

filedes      읽기/쓰기 포인터를 변경할 파일을 지정


offset      새롭게 지정할 읽기/쓰기 포인터의 위치를 의미  
오프셋이기 때문에 기준에 따라 음수가 될 수도 있음

whence      offset의 기준이 된다. 파일의 맨 처음(SEEK\_SET)  
현재 포인터의 위치(SEEK\_CUR)  
파일의 맨 마지막(SEEK\_END)

반환값      작업이 성공하면 파일의 첫 부분을 기준으로 한 포인터의 오프셋을 반환  
작업이 실패할 경우 (off\_t)-1이 반환

## II. 예제 코드

### Example Code #1




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

int main ()
{
    int filedес;
    char tmpstr[] = "Hello my friend!!\n";
    filedес = open ("ex3_text.txt", O_RDWR|O_CREAT, 0644);
    write (filedes, tmpstr, strlen(tmpstr));

    close (filedes);
    return 0;
}
```

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

int main ()
{
    int filedес;
    ssize_t nread;
    char tmpstr[1024];

    filedес = open ("ex3_text.txt", O_RDWR);
    memset (tmpstr, 0, sizeof(tmpstr));
    nread = read (filedes, tmpstr, 1024);
    printf ("%s", tmpstr);

    close (filedes);
    return 0;
}
```

### III. 실습 코드

#### Practice #1

##### □ Temp1.txt 파일을 읽어서 temp2.txt에 복사하는 프로그램

- 이름, 전공, 학번, 생년월일, 전화번호를 입력 받아서 Temp1.txt에 저장 (순서대로 입력)
  - 정보 입력은 scanf를 이용
  - 입력 정보 출력
  - Temp1.txt는 프로그램 내에서 생성 후 입력 받은 정보 저장
- Temp1.txt에서 데이터를 읽어서 출력한 후, temp2.txt에 복사함
  - Temp1.txt의 정보를 읽어서 출력
  - Temp2.txt에 데이터 복사
- Temp2.txt의 데이터를 읽어서 출력

---

*Thank you for your attention !!*

---

Q and A