|  |
| --- |
| 파일 Copy Program 작성 및 상세 설명  과제물 3 |

학과 : 컴퓨터공학과

학번 : 2015313754

이름 : 길태형

목 차

[1. Visual Studio Code에서 파일 Copy 프로그램 작성 및 실행 결과 4](#_Toc39180811)

[1.1. Visual Studio Code에서 파일 Copy 프로그램 작성 및 컴파일 4](#_Toc39180812)

[1.2. 파일 Copy 프로그램 실행 5](#_Toc39180813)

[2. 파일 Copy 프로그램에 대한 상세 설명 6](#_Toc39180814)

[2.1. 파일 Copy 프로그램의 실행 절차 6](#_Toc39180815)

그림 목차

[[그림 1] Visual Studio Code에서 Copy program 작성 4](#_Toc39180872)

[[그림 2] Copy program 컴파일 및 컴파일 결과 5](#_Toc39180873)

[[그림 3] Copy program 실행 결과 6](#_Toc39180874)

[[그림 4] 파일 Copy 프로그램의 코드 7](#_Toc39180875)

[[그림 5] Open() 함수 사용 8](#_Toc39180876)

[[그림 6] read() 함수 사용 9](#_Toc39180877)

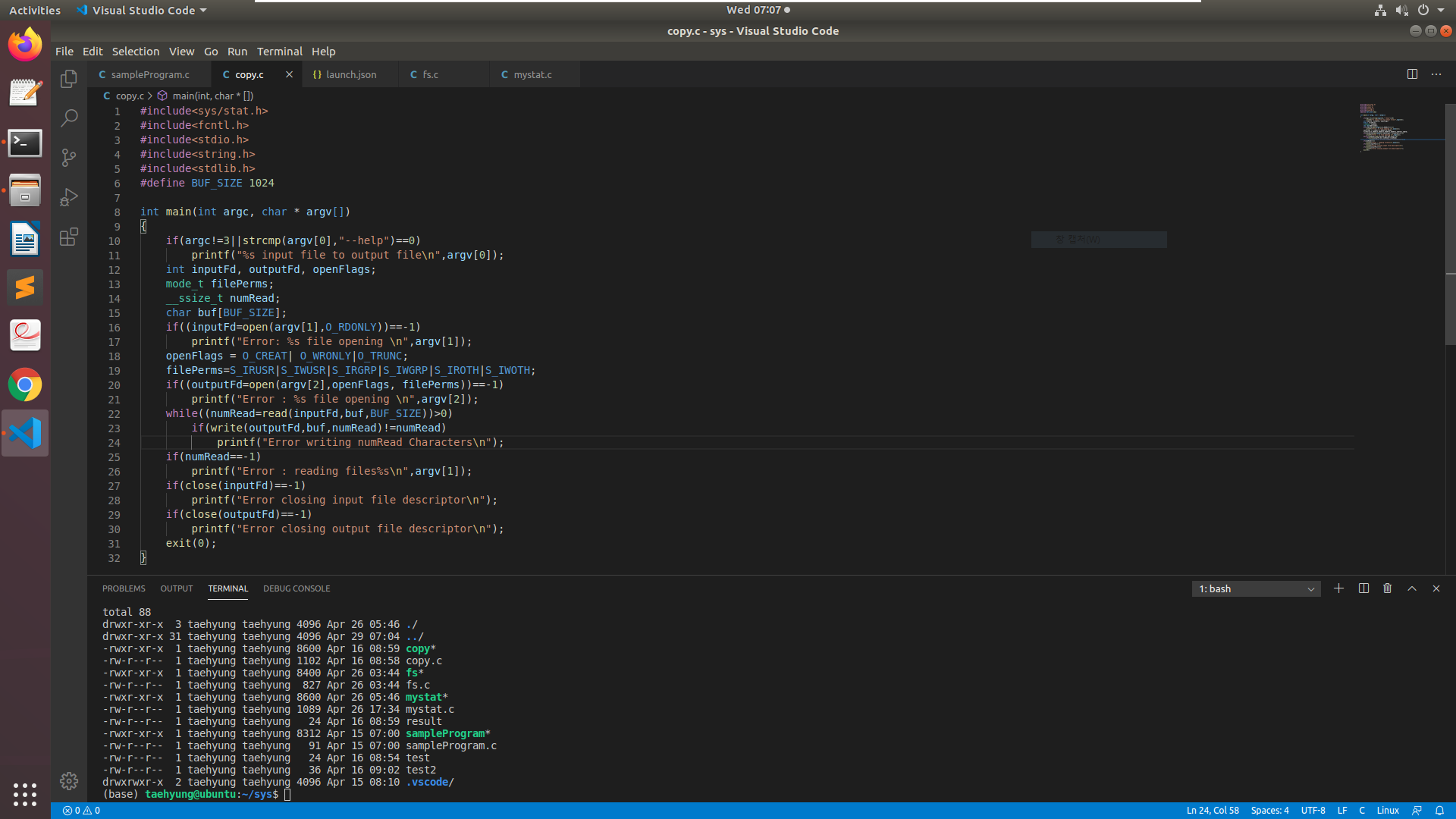
[[그림 7] write() 함수 사용 10](#_Toc39180878)

[[그림 8] close() 함수 사용 11](#_Toc39180879)

1. Visual Studio Code에서 파일 Copy 프로그램 작성 및 실행 결과
   1. Visual Studio Code에서 파일 Copy 프로그램 작성 및 컴파일

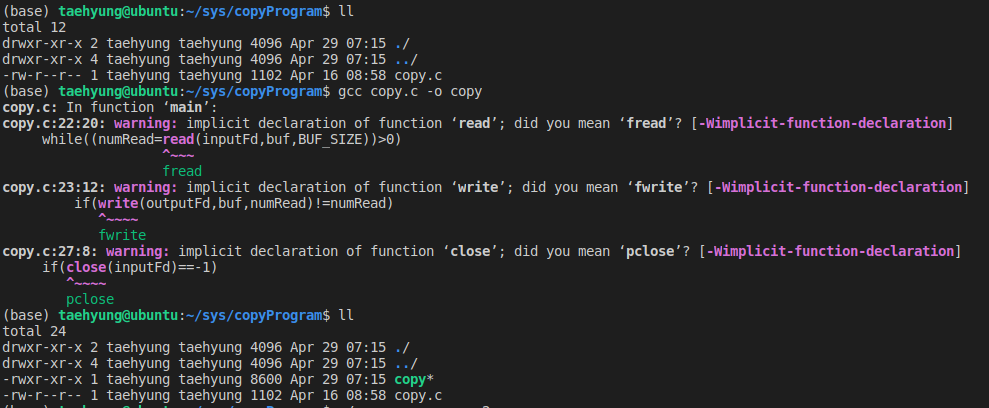
제 AWS의 반응속도가 너무 느려서, 개인 노트북의 VM의 linux환경에서 진행했습니다.

Copy 프로그램의 코드는 강의 내용과 동일합니다.



[그림 1] Visual Studio Code에서 Copy program 작성

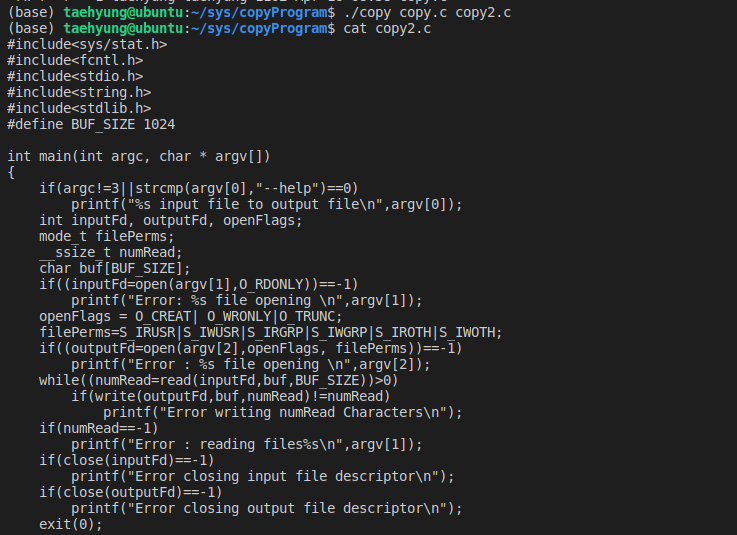
Copy program을 작성한 후, Visual Studio Code의 Terminal에서 gcc 명령어를 활용하여 작성한 C 프로그램을 컴파일 했습니다. 입력한 명령어는 ‘gcc copy.c -o copy’이고, 결과로써, copy 바이너리 파일을 산출했습니다. 아래 사진은 ‘ll’ 명령어를 사용하여 copy 바이너리 파일이 동일한 디렉토리에 있는 것까지 확인한 사진입니다.



[그림 2] Copy program 컴파일 및 컴파일 결과

* 1. 파일 Copy 프로그램 실행

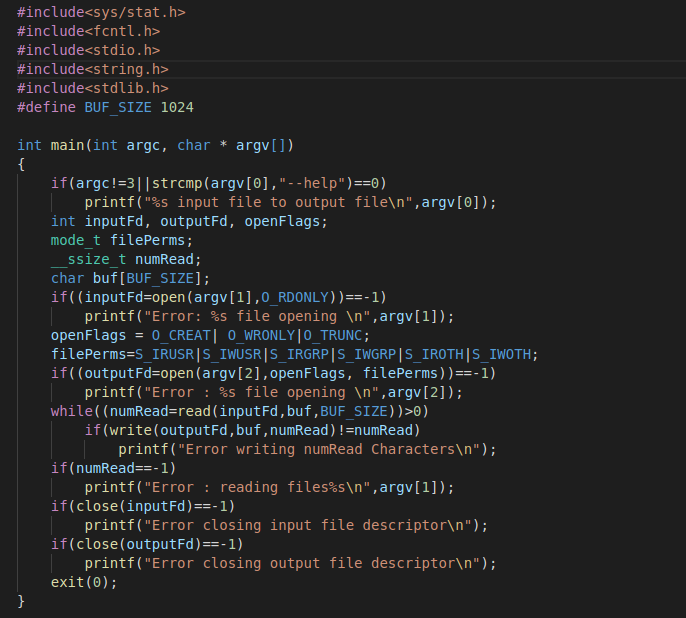
파일 Copy 프로그램을 실행하기 위해 입력한 명령어는 ‘./copy copy.c copy2.c’ 입니다. 즉, copy.c 파일을 복사하여copy2.c 파일을 생성했고, copy2.c 파일 내용을 확인하여, Copy program이 정상적으로 실행됨을 확인했습니다.



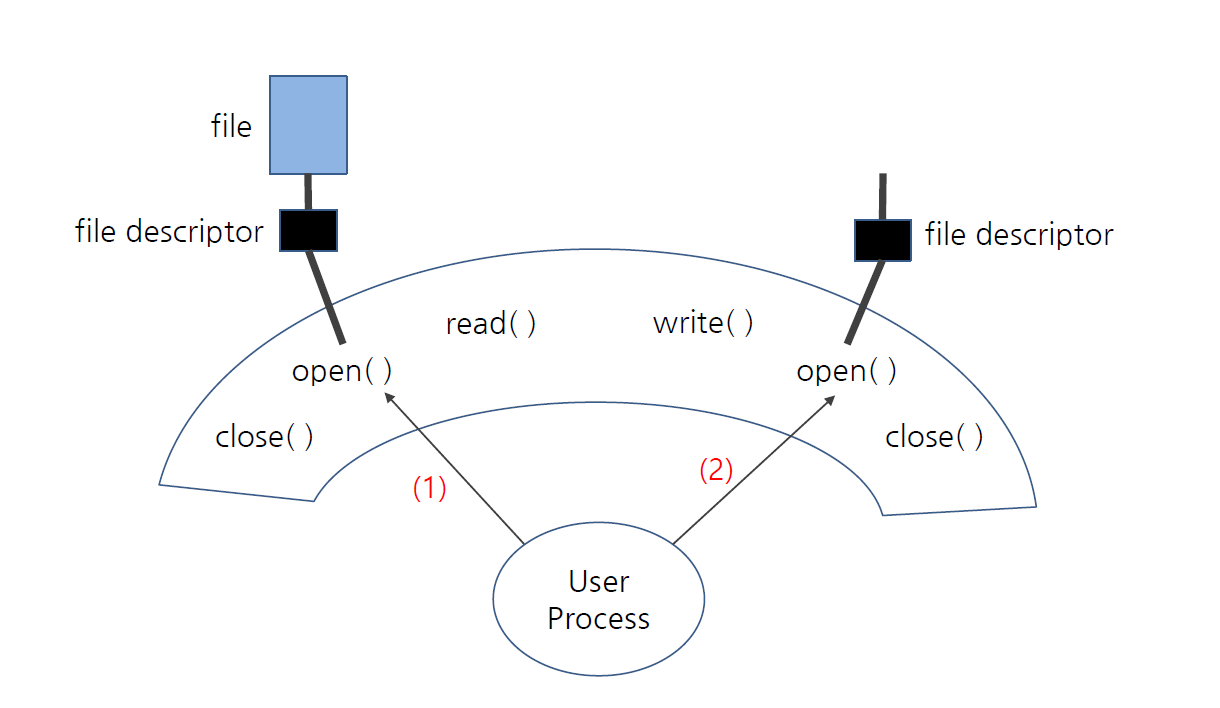
[그림 3] Copy program 실행 결과

1. 파일 Copy 프로그램에 대한 상세 설명
   1. 파일 Copy 프로그램의 실행 절차

아래 그림은 파일 Copy 프로그램의 코드입니다.



[그림 4] 파일 Copy 프로그램의 코드



[그림 5] Open() 함수 사용

먼저, Open()함수를 사용하여 file descriptor를 return 받습니다.

Open()함수의 원형은

Int open(const char \* pathname, int flag, …./\* mode\_t mode \*/); 입니다.

첫번째 매개변수 pathname에 존재하는 파일명, 혹은 생성할 파일명을 입력합니다.

두번째 매개변수 Flag에는 open의 목적을 명시합니다.

예를 들면,

Flag로 O\_RDONLY을 입력하면, open한 파일을 읽기 전용으로만 사용하고,

O\_WRONLY을 입력하면, open한 파일을 쓰기 전용으로만 사용하고,

O\_RDWR을 입력하면, open한 파일을 읽기와 쓰기 두 가지 목적으로 사용합니다.

O\_RDONLY와 같은 값들은 라이브러리에 이미 정의된 int value일 것입니다.

세번째 매개변수, Mode\_t 매개변수는 이미 존재하는 파일을 open 하는 경우가 아닌, 새로 파일을 생성하여 open 할때, 생성하게 되는 파일의 권한을 명시하기 위해 사용됩니다.

예를 들어, 새로 생성하게 되는 파일의 권한을 rwxr-x--x로 설정한다면, mode\_t의 값으로

S\_IRWXU | S\_IRGRP | S\_IXGRP | S\_IXOTH 을 입력하면 됩니다.

권한들 간의 OR bit operation을 통해, permission을 표현합니다.

Copy할 input file을 읽기 전용으로 열고, output file을 생성, 작성, 덧붙일 수 있는 목적으로 엽니다. Output file의 목적을 설정하는 해당 코드는 openFlags = O\_CREATE|O\_WRONLY|O\_TRUNC입니다.

Output file의 permission은 rw-rw-rw- 입니다. Output file의 permission을 설정하는 해당 코드는

Fileperms = S\_IRUSR|S\_IWUSR|S\_IRGRP|S\_IWGRP|S\_IROTH|S\_IWOTH; 입니다.

Open()함수를 사용하여 input file, output file을 여는 해당 코드는

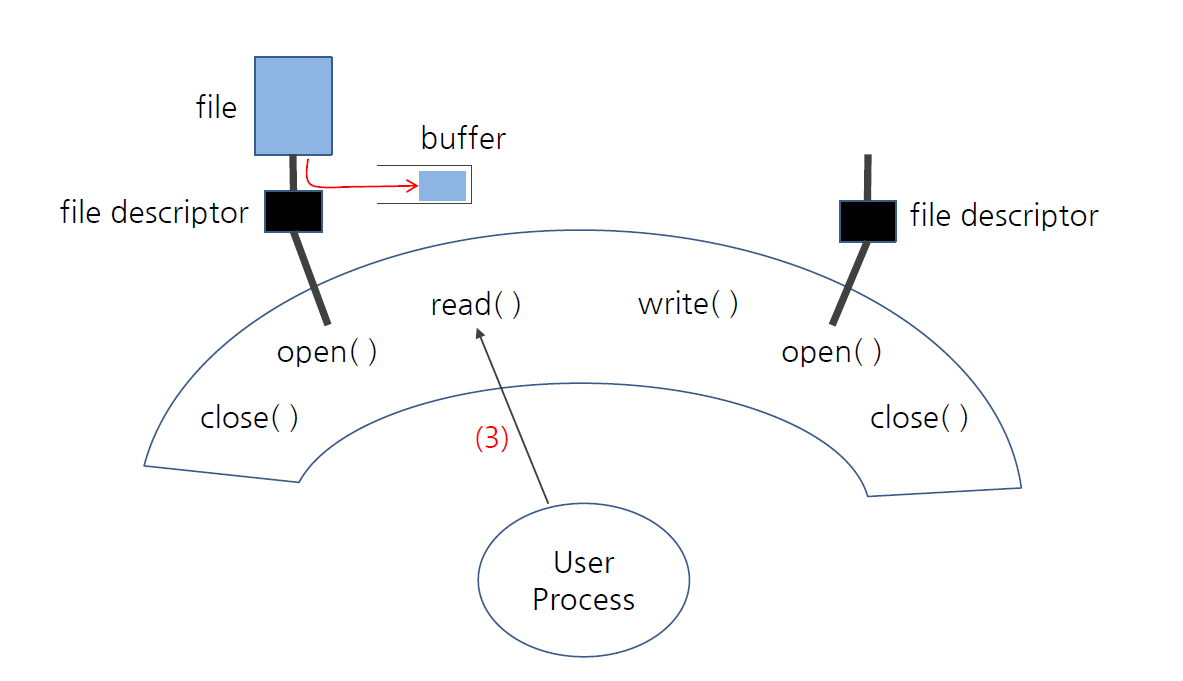
inputFd=open(argv[1],O\_RDONLY); 입니다.

inputFd에는 input file의 file descriptor가 저장됩니다.

outputFd = open(argv[2], openFlags, filePerms); 를 사용하여 output file을 생성하거나 엽니다.

Output file을 생성해야할 수 있으므로, mode\_t인자에 추가적으로 값을 입력합니다.

그리고 error 상황을 다루기 위해, 조건문을 사용하여, open()함수의 반환값이 -1인 경우, error임을 출력하도록 작성했습니다.



[그림 6] read() 함수 사용

Read()함수는 input file로부터 buffer로 data를 읽어오는 역할을 수행합니다.

Read()함수의 원형은 다음과 같습니다.

ssize\_t read(int fd, void \* buffer, size\_t count);

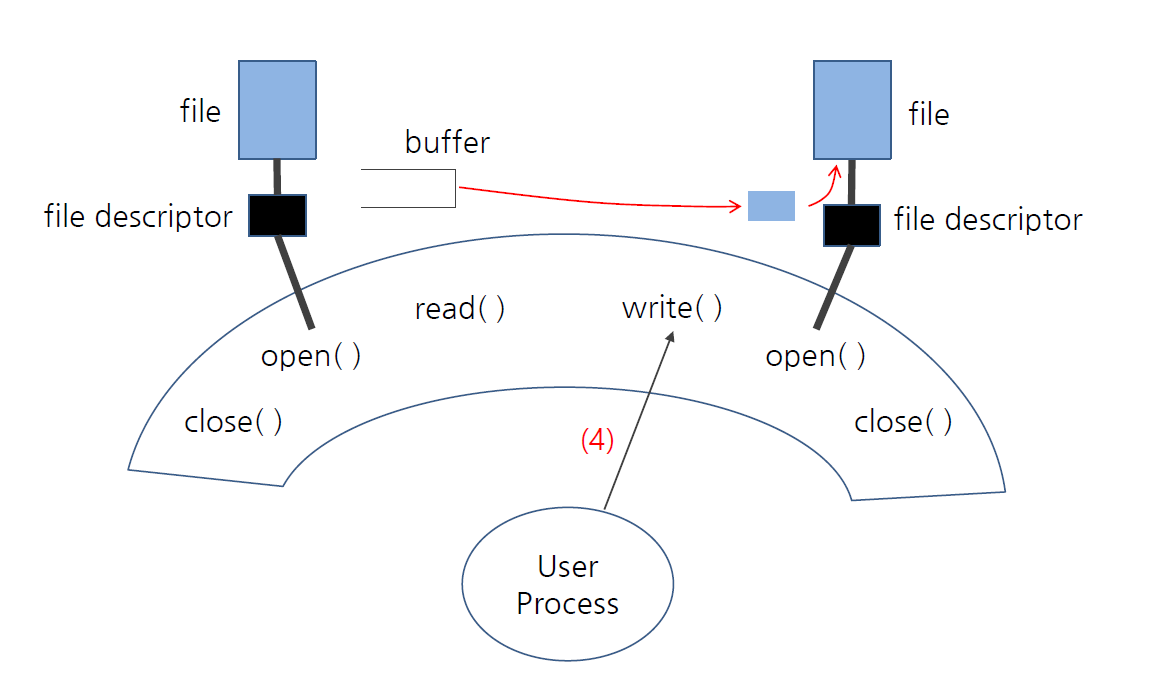
첫번째 인자 fd에는 file descriptor를 입력하고, 두번째 인자 buffer에는 input data가 놓여질 memory buffer의 주소를 입력하고, 세번째 인자 count에는 읽어올 최대 byte 수를 입력합니다.

반환값은, input file로부터 얼마만큼의 byte를 읽어왔는지를 반환합니다.

EOF일 경우, 0을 반환하고, error가 발생할 경우 -1을 반환합니다.

numRead=read(inputFd, buf, BUF\_SIZE) 가 해당 input file로부터 최대 버퍼의 size만큼 읽어오는 코드입니다. while문의 조건으로 사용하여, 읽어온 byte의 수가 0초과 일 때 만 반복 실행합니다.

또한, error 상황을 다루기 위해, numRead가 -1인 경우, 즉 error가 발생한 경우, Error 메시지를 출력하도록 작성했습니다.



[그림 7] write() 함수 사용

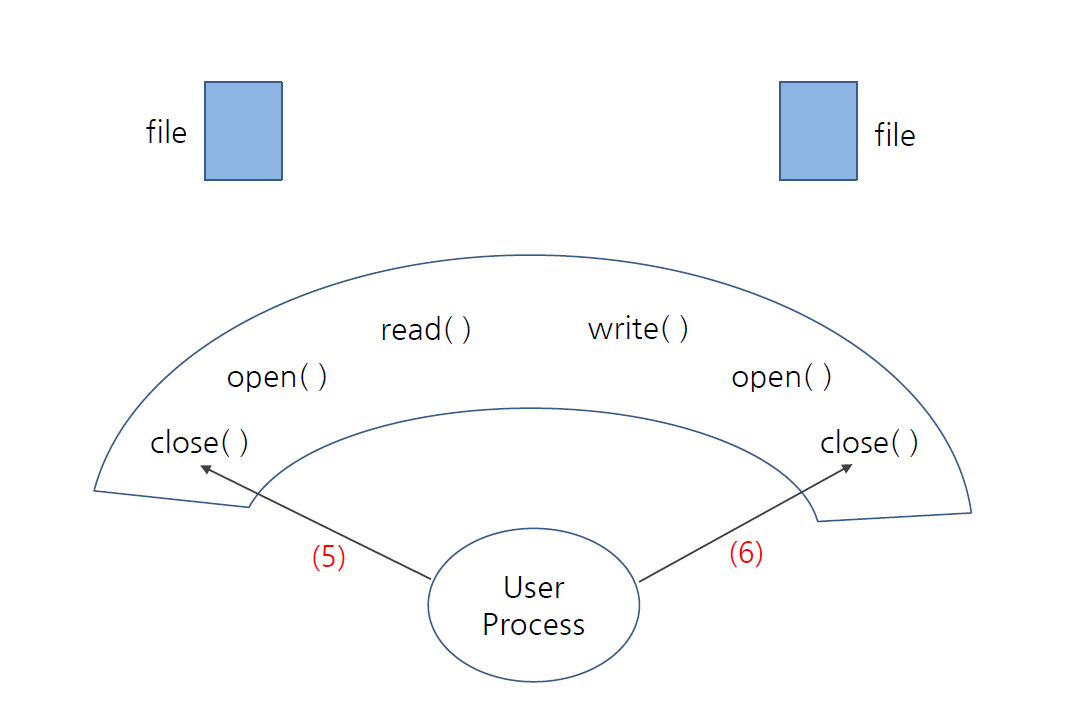
Write()함수는 buffer에 있는 data를 output file로 옮겨 쓰는 역할을 수행합니다.

Write()함수의 원형은 다음과 같습니다.

ssize\_t write(int fd, void \*buffer, size\_t count)  
첫번째 인자 fd에는 output file의 file descriptor를 입력하고, 두번째 인자buffer에는 data를 읽어올 buffer의 memory address를 입력합니다. 세번째 인자 count에는 buffer로부터 얼마만큼의 data를 가져올 것 인지 지정합니다.

Write()함수는 정상적으로 쓰여진 byte의 수를 반환하고, erro가 있을 시에는 -1을 반환합니다.

위의 코드 중, write(outputFd,buf,numRead)에 해당됩니다. 반환 값이 numRead, 즉 읽어온 data의 byte 수와 같지 않을 경우 Error를 반환합니다. Buffer의 Data가 부분적으로 손실되거나, error가 발생할 경우 모두, numRead와 값이 같지 않을 것입니다.



[그림 8] close() 함수 사용

Close()함수는 file descriptor를 닫는 역할을 수행합니다.

Close()함수의 원형은

Int close(int fd); 입니다.

첫번째 인자 fd에는 닫을 file의 descriptor를 입력합니다.

성공적으로 닫혔을 경우, 0을 반환하고, error가 발생했을 경우 -1를 반환합니다.

위의 코드에서는 close(inputFd) close(outputFd)에 해당하고,

반환값이 -1인지 비교함으로써 error가 발생했는지 확인합니다.