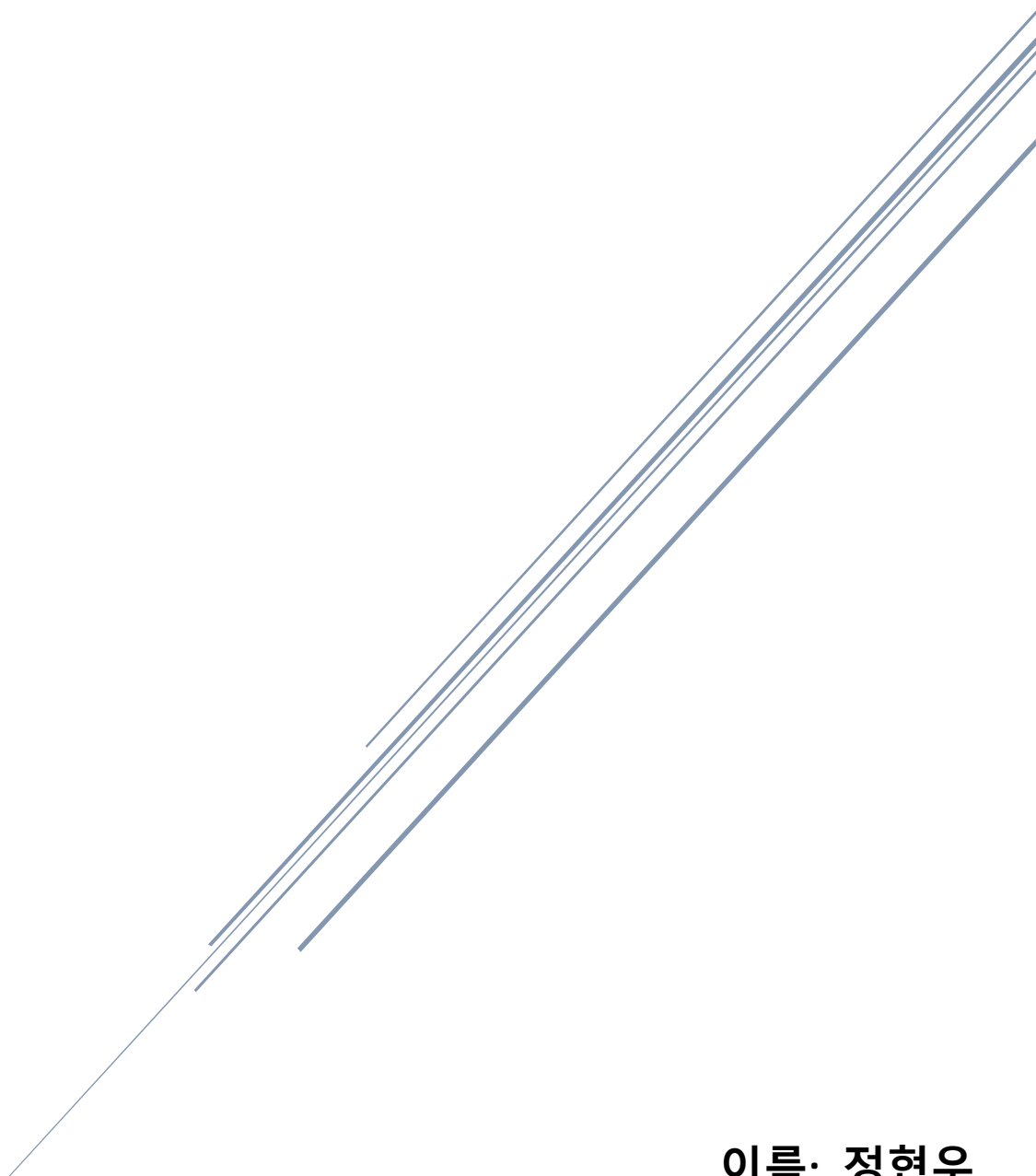


시스템프로그래밍(SW)

과제 3



이름: 정현우
학번: 32204236

목차

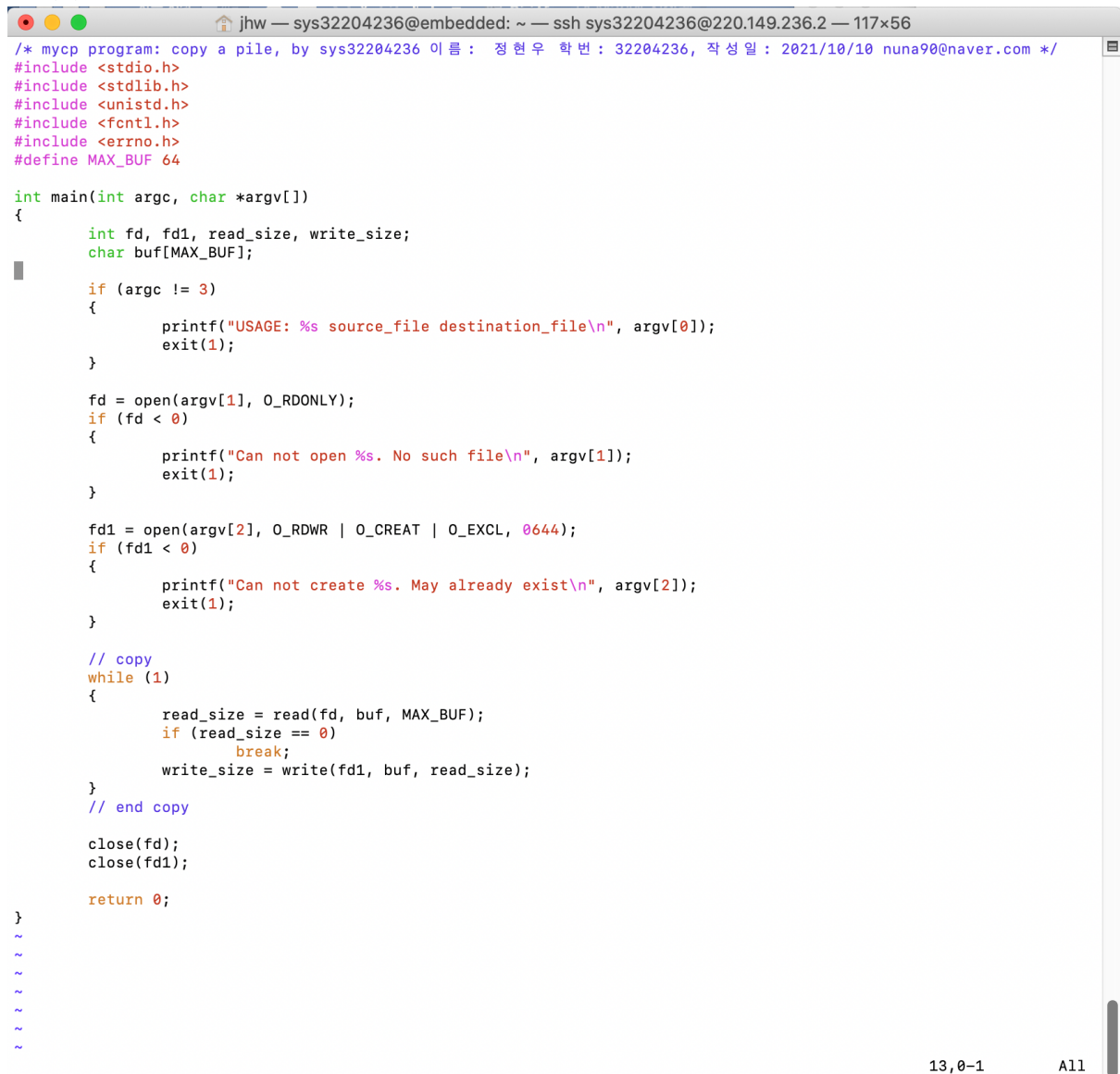
I. 서론.....	2
II. 서론.....	2
1. mycp 프로그램 작성.....	2
2. 속성까지 복사하는 mycp 프로그램 작성.....	4
III. 결론.....	5

I. 서론

이번 과제는 mycp 프로그램을 작성하는 것으로, cp 명령어를 직접 구현하는 것이다. 요구 사항은 총 세가지로, 첫번째는 argc, argv를 사용하는 것이고 두번째는 이미 존재하면 생성 실패 및 에러 메시지 출력을 하는 것이고, 세번째는 whoami와 date를 사용하여 학번과 날짜를 보이게 하는 것이다. 여기에 추가적으로 속성, 즉 접근 권한까지 복사하는 것이 이번 과제의 목표이다.

II. 본문

1. mycp 프로그램 작성



```
/* mycp program: copy a file, by sys32204236 이름 : 정현우 학번 : 32204236, 작성일 : 2021/10/10 nuna90@naver.com */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#define MAX_BUF 64

int main(int argc, char *argv[])
{
    int fd, fd1, read_size, write_size;
    char buf[MAX_BUF];

    if (argc != 3)
    {
        printf("USAGE: %s source_file destination_file\n", argv[0]);
        exit(1);
    }

    fd = open(argv[1], O_RDONLY);
    if (fd < 0)
    {
        printf("Can not open %s. No such file\n", argv[1]);
        exit(1);
    }

    fd1 = open(argv[2], O_RDWR | O_CREAT | O_EXCL, 0644);
    if (fd1 < 0)
    {
        printf("Can not create %s. May already exist\n", argv[2]);
        exit(1);
    }

    // copy
    while (1)
    {
        read_size = read(fd, buf, MAX_BUF);
        if (read_size == 0)
            break;
        write_size = write(fd1, buf, read_size);
    }
    // end copy

    close(fd);
    close(fd1);

    return 0;
}
```

```
jhw — sys32204236@embedded: ~ — ssh sys32204236@220.149.236.2 — 7...
sys32204236@embedded:~$ vi mycp.c
sys32204236@embedded:~$ gcc -o mycp mycp.c
sys32204236@embedded:~$ ./mycp
USAGE: ./mycp source_file destination_file
sys32204236@embedded:~$ ./mycp hi.txt hello_new.txt
Can not open hi.txt. No such file
sys32204236@embedded:~$ ./mycp hello.txt hello_new.txt
sys32204236@embedded:~$ ls -l
total 80
-rw-r--r-- 1 sys32204236 sys32204236 8980  4월 20  2016 examples.desktop
-rwxrwxr-x 1 sys32204236 sys32204236 5784 10월  3 00:25 gdb_test.out
-rw-r--r-- 1 sys32204236 sys32204236  28 10월 10 23:44 hello_new.txt
-rw-rw-r-- 1 sys32204236 sys32204236  28 10월 10 23:28 hello.txt
[-rwxrwxr-x 1 sys32204236 sys32204236 5492 10월 10 23:43 mycp
-rw-rw-r-- 1 sys32204236 sys32204236  892 10월 10 23:43 mycp.c
-rw-rw-r-- 1 sys32204236 sys32204236  502 10월  2 23:23 my_favorite_poem.txt
[-rw-rw-r-- 1 sys32204236 sys32204236 1011 10월 10 22:56 temp.c
-rw-rw-r-- 1 sys32204236 sys32204236  140 10월  3 00:10 test1.c
-rw-rw-r-- 1 sys32204236 sys32204236  140 10월 10 22:50 test2.c
[-rw-rw-r-- 1 sys32204236 sys32204236  140 10월  3 00:10 test.c
-rw-rw-r-- 1 sys32204236 sys32204236  912 10월  3 00:13 test.o
-rwxrwxr-x 1 sys32204236 sys32204236 4728 10월  3 00:13 test.out
-rw-rw-r-- 1 sys32204236 sys32204236  631 10월  3 00:13 test.s
-rw-rw-r-- 1 sys32204236 sys32204236 2029 10월  3 00:10 wc_man.txt
sys32204236@embedded:~$ cat hello_new.txt
Hello!
My name is 정 현 우
sys32204236@embedded:~$ cat hello.txt
Hello!
My name is 정 현 우
sys32204236@embedded:~$ ./mycp hello.txt hello_new.txt
Can not create hello_new.txt. May already exist
sys32204236@embedded:~$
```

argc와 argv를 main() 함수의 인자로 전달하고, argc가 3이 아니면 에러 메시지를 출력하고 종료하도록 하였다. open() 시스템 콜을 사용해 argv[1]로 받은 파일 이름으로 파일 디스크립터를 읽기 전용으로 열고, 파일이 제대로 열렸는지 if 문으로 확인한다. 제대로 열리지 않았을 경우 에러 메시지를 출력하고 종료한다. 그 다음 argv[2]로 받은 파일 이름을 통해 읽기 쓰기가 가능하도록 하면서 파일을 생성하는데, 이때 이미 파일이 존재할 경우 생성하지 못하게 하고 에러 메시지를 출력하도록 하였다. 접근 권한은 8진수 0644로 해주었다. 그 다음, while 문 내에서 read() 시스템 콜과 write() 시스템 콜을 사용하여 복사를 진행하였다. 그 다음 두 개의 파일 디스크립터를 close() 시스템 콜로 닫아주고 종료한다. 또한, 상단에 파일 이름, 목적, 만든 사람, 이메일, 날짜와 같은 정보를 주석으로 추가해두었다.

이렇게 작성된 프로그램을 터미널 상에서 확인해보았다. gcc로 번역하고 테스트를 해보았다. 복사할 파일이 없을 때, 복사 내용이 저장될 파일이 이미 존재할 때에 대해 에러 처리가 잘 되어 있으며, 인자가 부족할 때는 사용법을 보여준다. 실행해서 hello.txt를 hello_new.txt로 복사하고 ls -l 명령어를 사용하면, 속성은 다르지만 파일이 만들어져 있는 것을 확인할 수 있다. cat을 통해 내용을 보면 잘 복사되어 있는 것을 알 수 있다.

redirection을 구현하는 것과 무엇이 다른지 살펴보았다. redirection을 구현할 때는 dup2() 를 사용

하여 파일 디스크립터를 중복시킨다. 즉, 파일 디스크립터가 같은 inode를 가리키도록 하여, 결국 같은 내용을 가지게 하는 방식으로 알 수 있다. 반면 cp를 구현할 때는 파일의 내용 수준에서 복사를 하는 것이 다르다. 과제를 하기 시작했을 때 dup2()를 사용하려 했으나, 내용을 복사하는 것과 파일 디스크립터를 중복시키는 것에는 차이가 있을 것이라 생각하여 생각을 바꾸고 위와 같이 while 문을 사용한 내용 복사를 구현하였다.

2. 속성까지 복사하는 mycp 프로그램 작성

```
jhw — sys32204236@embedded: ~ — ssh sys32204236@220.149.236.2 — 120x59
/* mycp program: copy a file, by sys32204236 이름 : 정현우 학번 : 32204236, 작성일 : 2021/10/10 nuna90@naver.com */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#define MAX_BUF 64

int main(int argc, char *argv[])
{
    int fd, fd1, read_size, write_size;
    char buf[MAX_BUF];
    struct stat st;

    if (argc != 3)
    {
        printf("USAGE: %s source_file destination_file\n", argv[0]);
        exit(1);
    }

    fd = open(argv[1], O_RDONLY);
    if (fd < 0)
    {
        printf("Can not open %s. No such file\n", argv[1]);
        exit(1);
    }

    stat(argv[1], &st);

    fd1 = open(argv[2], O_RDWR | O_CREAT | O_EXCL, 0644);
    if (fd1 < 0)
    {
        printf("Can not create %s. May already exist\n", argv[2]);
        exit(1);
    }

    // copy
    chmod(argv[2], st.st_mode);
    while (1)
    {
        read_size = read(fd, buf, MAX_BUF);
        if (read_size == 0)
            break;
        write_size = write(fd1, buf, read_size);
    }
    // end copy

    close(fd);
    close(fd1);

    return 0;
}
~
~
~
~
```

40,28-35

All

```
jhw — sys32204236@embedded: ~ — ssh sys32204236@220.149.236.2 — 84x29
sys32204236@embedded:~$ vi mycp.c
sys32204236@embedded:~$ gcc -o mycp mycp.c
sys32204236@embedded:~$ ./mycp hello.txt hello_attr.txt
sys32204236@embedded:~$ ls -l
total 84
-rw-r--r-- 1 sys32204236 sys32204236 8980  4월  20  2016 examples.desktop
-rwxrwxr-x 1 sys32204236 sys32204236 5784 10월  3 00:25 gdb_test.out
-rw-rw-r-- 1 sys32204236 sys32204236  28 10월 10 23:48 hello_attr.txt
-rw-rw-r-- 1 sys32204236 sys32204236  28 10월 10 23:44 hello_new.txt
-rw-rw-r-- 1 sys32204236 sys32204236  28 10월 10 23:28 hello.txt
-rwxrwxr-x 1 sys32204236 sys32204236 5864 10월 10 23:48 mycp
-rw-rw-r-- 1 sys32204236 sys32204236 1006 10월 10 23:48 mycp.c
-rw-rw-r-- 1 sys32204236 sys32204236  502 10월  2 23:23 my_favorite_poem.txt
-rw-rw-r-- 1 sys32204236 sys32204236 1011 10월 10 22:56 temp.c
-rw-rw-r-- 1 sys32204236 sys32204236  140 10월  3 00:10 test1.c
-rw-rw-r-- 1 sys32204236 sys32204236  140 10월 10 22:50 test2.c
-rw-rw-r-- 1 sys32204236 sys32204236  140 10월  3 00:10 test.c
-rw-rw-r-- 1 sys32204236 sys32204236  912 10월  3 00:13 test.o
-rwxrwxr-x 1 sys32204236 sys32204236 4728 10월  3 00:13 test.out
-rw-rw-r-- 1 sys32204236 sys32204236  631 10월  3 00:13 test.s
-rw-rw-r-- 1 sys32204236 sys32204236 2029 10월  3 00:10 wc_man.txt
sys32204236@embedded:~$ cat hello_attr.txt
Hello!
My name is 정 현 우
sys32204236@embedded:~$ whoami
sys32204236
sys32204236@embedded:~$ date
2021. 10. 10. (일) 23:51:07 KST
sys32204236@embedded:~$
```

stat() 시스템 콜을 사용해서 속성 정보를 알아낸다. 기존에 만든 코드를 수정하는 방식으로 작성되었다. 필요한 헤더 파일을 만들고, stat()을 통해 얻은 argv[1] 파일 정보의 구조체에서 st_mode를 통해 접근 권한 정보를 얻고, chmod()를 이용하여 argv[2] 파일의 속성을 바꿔준다.

프로그램을 작성한 다음, gcc로 번역하여 테스트해보았다. hello.txt 파일을 hello_attr.txt 파일로 복사하고 ls -l 커맨드로 확인해보았다. hello.txt와 첫 mycp로 복사한 hello_new.txt의 속성은 다른 것은 확인할 수 있지만, 새로 고친 mycp로 복사한 hello_attr.txt는 속성이 hello.txt로부터 잘 복사되었다고 확인할 수 있다. 그 다음 cat 커맨드를 통해 내용까지 확실히 복사된 것을 볼 수 있다. 마지막에는 whoami와 date를 통해 내가 누군지를 보이고, 과제를 한 날짜와 시간에 대한 정보를 출력하였다.

III. 결론

mycp를 작성하며 각종 시스템 콜을 사용하는 과제를 진행하였다. 기존에 존재하는 커맨드인 cp를 open()으로 파일을 열고 파일 디스크립터를 얻어 얻은 파일 디스크립터를 통해 read()와 write() 시스템 콜을 사용하며, 마지막에 close() 시스템 콜을 사용하는 방식으로 직접 구현하는 것이다. argc, argv를 통해 터미널 상에 입력하는 명령어와 인자들의 관계를 파악하고, 그를 활용하여 구현하는 것에 성공하였다. 추가로 주어진 요구사항인 속성까지 복사하는 기능을 추가하는 것은 만들어 두었던 mycp.c 소스에서 stat()과 chmod()와 필요한 헤더 파일 등을 추가하는 방식으로 만들어졌다. 이미 존재하는 파일은 생성 실패 및 에러 메시지 출력하고 whoami와 date 커맨드까지 사용하여 과제에서 요구된 3가지 사항들과 추가 요구 사항까지 모두 지켜 mycp 프로그램을 구현하는 것에 성공하였다.

특히 인상 깊었던 부분이 존재하는데, 바로 버퍼를 사용하는 것이다. 전까지는 c 언어에서 파일 입출력을 할 때, 왜 버퍼를 지정하는지 이해하지 못했다. 그러나 이번 실습에서 `read()`, `write()`에서 버퍼를 사용하며 버퍼의 크기를 크게 할 것이냐, 작게 할 것이냐에 대한 강의 내용과 연관 지어 왜 버퍼를 사용해야 하는지 생각하자, 버퍼의 필요성이 이해가 되었다. 버퍼를 쓰지 않고 한번에 쓰기엔, 메모리 공간을 너무 많이 차지하게 되고, 데이터를 한 바이트씩 적기엔 시간이 너무 많이 드는 것이다. 이렇게 새롭게 배우는 것을 통해 이전에 제대로 알지 못했던 것을 깊게 배우는 경험을 가질 수 있었던 것이 굉장히 인상깊었다.