



[HW4]_20172129_박예솔

'cp src dst' 명령어 구현

과목	운영체제
제출 일자	2020년 11월 21일
담당 교수	박호현 교수님
학 과	컴퓨터공학과
학 번	20172129
이 름	박예솔

* 리눅스 셸 명령어인 'cp src dst' 명령어를 직접 구현하고 테스트 하시오.

1. 코드 설명

1) src와 dst가 모두 regular file : src file → dst file로 copy

```
void CopyFile(char* input, char* output){
    int n, in, out;
    char buf[1024];

    // copy regular file to regular file.
    if ((in = open(input, O_RDONLY)) < 0) {
        perror(input);
        exit(0);
    }

    if ((out = open(output, O_WRONLY|O_CREAT|O_TRUNC, S_IRUSR|S_IWUSR)) < 0) {
        perror(output);
        exit(0);
    }
    printf(">> Start copying regular file...\n");
    while ((n = read(in, buf, sizeof(buf))) > 0){
        write(out, buf, n);
    }
    printf(">> Finish copying regular file :D\n");
    close(out);
    close(in);
}

int main(int argc, char **argv)
{
    int n, in, out;
    char buf[1024];
    struct stat in_buf;
    struct stat out_buf;
    DIR* input;
    DIR* output;

    if (argc < 3) {
        write(2, "Usage : copy src dst\n", 25);
        return -1;
    }

    stat(argv[1], &in_buf);
    stat(argv[2], &out_buf);

    // 1. argv[1] && argv[2] == regular file
    if(S_ISREG(in_buf.st_mode) == 1 && S_ISREG(out_buf.st_mode) == 1){
        printf(">> start copying: regular file -> regular file\n");
        CopyFile(argv[1], argv[2]);
    }
}
```

우선 struct stat를 사용해서 src와 dst가 Regular File인지 Directory인지를 확인한다.

만약 둘 다 Regular File이라면 CopyFile(src, dst) 함수를 실행하여 파일을 복사해준다. 이때 파일의 Low Level API인 open, read, write, close를 사용하여 파일 복사를 구현하였다. 이는 강의 슬라이드에 올라온 코드와 거의 같으므로 생략하겠다.

2) src와 dst가 모두 directory :

src directory (file & sub-directory) → dst directory로 copy

```
// 2. argv[1] && argv[2] == directory file
else if(S_ISDIR(in_buf.st_mode) == 1 && S_ISDIR(out_buf.st_mode) == 1){
    printf(">> start copying: directory file -> directory file\n");

    input = opendir(argv[1]);
    output = opendir(argv[2]);

    inputFile[0]='\0';
    strcat(inputFile, ".");
    strcat(inputFile, argv[1]);

    outputFile[0]='\0';
    strcat(outputFile, ".");
    strcat(outputFile, argv[2]);
    int res = CopyDir(input, output, argv[1]);
    if(res == 0){
        printf(">> Fail copying directory file :(\n");
    }
}
```

src와 dst가 모두 폴더인 경우에는 우선 복사해 올 디렉토리의 경로와 복사해서 붙여넣을 디렉토리의 경로를 잘 저장해 두어야 한다. 그 이유는 디렉토리 속의 서브 디렉토리를 들어가면 경로를 추가해서 덧붙이고, 서브 디렉토리에서 나오면 경로를 다시 지워주어야 하기 때문이다.

```

int CopyDir(DIR *read_dir, DIR *write_dir, char *dirname){
    while(1) {
        struct dirent *r_dirt = readdir(read_dir);
        struct dirent *w_dirt = readdir(write_dir);
        struct stat statBuf;
        stat(dirname, &statBuf);

        if(r_dirt == NULL){ // EOF
            printf(">> Finish copying directory file [%s] :D\n", dirname);
            return 1;
        }
        lstat(r_dirt->d_name, &statBuf);
        if(S_ISDIR(statBuf.st_mode) == 1){ // directory
            if(strcmp(r_dirt->d_name, ".") == 0 || strcmp(r_dirt->d_name, "..") == 0) {
                printf(">> except file: %s\n", r_dirt->d_name); // Pass
            }
            else{
                printf(">> copying directory file: %s\n", r_dirt->d_name);
                sprintf(in_buf, "%s/%s", inputFile, r_dirt->d_name);
                sprintf(out_buf, "%s/%s", outputFile, r_dirt->d_name);
                printf(">> input directory: %s/ output directory: %s\n", in_buf, out_buf);

                struct stat tempbuf;
                stat(in_buf, &tempbuf);
                int mkdirFlag = mkdir(out_buf, tempbuf.st_mode);
                if(mkdirFlag == 0){
                    printf(">> copy sub-directory %s.\n", in_buf);
                    strcpy(intemp, inputFile);
                    strcpy(outtemp, outputFile);
                    strcpy(inputFile, in_buf);
                    strcpy(outputFile, out_buf);
                }
                else{
                    perror(">> Error: Can't make sub-directory..T.T\n");
                    printf("%d: %s\n", errno, strerror(errno));
                    exit(0);
                }
                DIR* indir = opendir(in_buf);
                DIR* outdir = opendir(out_buf);
                CopyDir(indir, outdir, in_buf);
                // RECURSIVE END
                strcpy(inputFile, intemp);
                strcpy(outputFile, outtemp);
            }
        }
    }
}

```

따라서 전역변수 inputFile을 이용해서 경로를 업데이트해준다. 왼쪽 코드를 보면 CopyDir 함수는 읽고 쓸 디렉토리의 포인터와, 읽을 디렉토리의 이름을 매개변수로 받아온다. 해당 폴더로 들어가서 우선 폴더 속에 또 폴더가 있는지 _ISDIR을 이용해서 확인해준다. 만약 서브 폴더가 있으면 mkdir을 사용해서 복사할 서브 폴더를 붙여 넣을 폴더의 경로 안에 생성해준다. 폴더가 생성되고 나면 파일 경로를 strcpy를 이용해서 갱신시킨다. 이렇게 폴더 속의 폴더를 재귀함수로 계속 불러주며 경로를 갱신해준다.

```

else if(S_ISREG(statBuf.st_mode) == 1){ // regular file
    printf(">> copying regular file: %s\n", r_dirt->d_name);
    char ref[1024];
    sprintf(ref, "%s/%s", inputFile, r_dirt->d_name);
    char outref[1024];
    sprintf(outref, "%s/%s", outputFile, r_dirt->d_name);

    CopyFile(ref, outref);
}
else{
    printf(">> error: Not regular file & directory file. (Filename: %s)\n", r_dirt->d_name);
}

```

이렇게 디렉토리나 서브 디렉토리를 찾다가 regular 파일을 찾게 되면 파일 경로에 복사할 파일의 이름을 합쳐서 ref로 저장한다. 그렇게 저장한 경로를 앞서 언급한 CopyFile(src, dst) 함수를 이용해서 파일 복사를 진행해주면 원하는 경로에 복사할 파일과 같은 이름으로 파일이 복사된다.

3) src가 directory이고 dst가 regular file → 오류 메시지 출력

```

// 3. argv[1] == directory file && argv[2] == regular file
else if(S_ISDIR(in_buf.st_mode) == 1 && S_ISREG(out_buf.st_mode) == 1){
    perror(">> Copy Failed. [src] is directory file. But [dst] is regular file...T.T\n");
    exit(1);
}

```

S_ISDIR 과 S_ISREG를 이용해서 src가 디렉토리이고 dst가 Regular File 이면 오류 메시지를 출력해준다.

```
// 4. argv[1] == regular file && argv[2] == directory file
else if(S_ISREG(in_buf.st_mode) == 1 && S_ISDIR(out_buf.st_mode) == 1){
    printf(">> start copying: regular file -> directory file\n");
    // read directory path
    char input[50];
    input[0]='\0';
    strcat(input, "./");
    strcat(input, argv[1]);
    printf("%s\n", input);
    // write directory path
    char output[50];
    output[0]='\0';
    strcat(output, "./");
    strcat(output, argv[2]);
    strcat(output, "/");
    strcat(output, argv[1]);

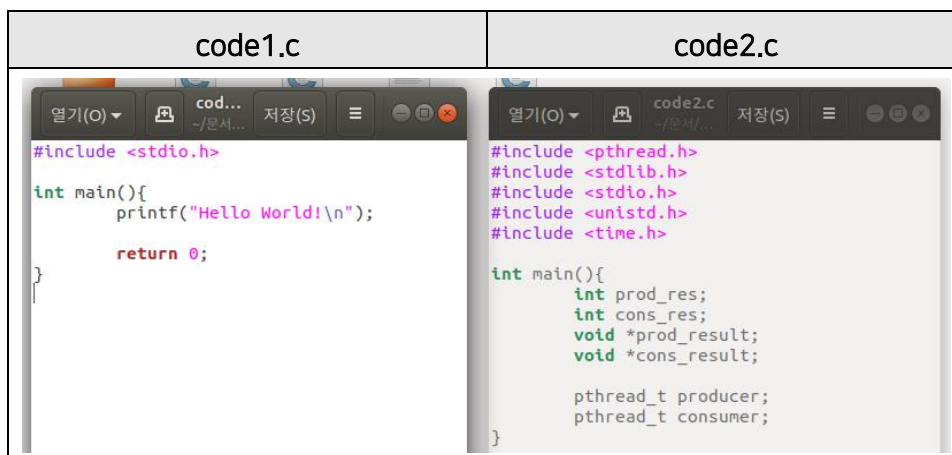
    CopyFile(input, output);
}
```

추가적으로 src가 regular file이고 dst가 폴더인 경우를 구현했는데, input은 그대로 src로 두고, dst를 파일 경로로 만들어 주어서 원하는 디렉토리 안에 파일이 복사될 수 있도록 구현하였다.

2. 실행 결과

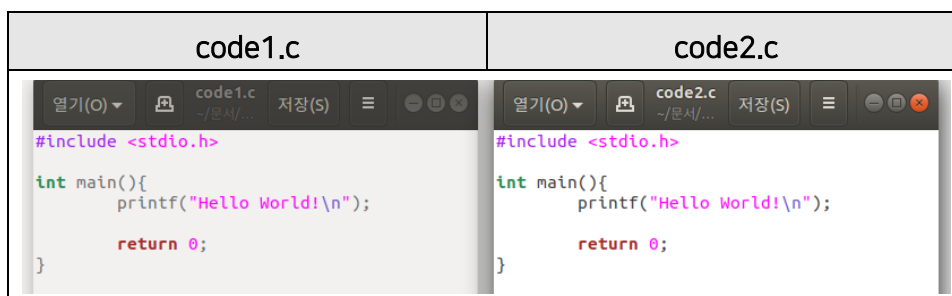
1) src와 dst가 모두 regular file : src file → dst file로 copy

< 실행 전 >



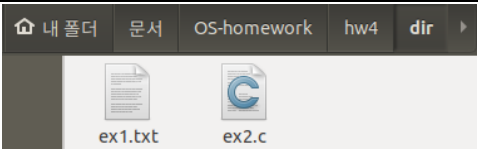
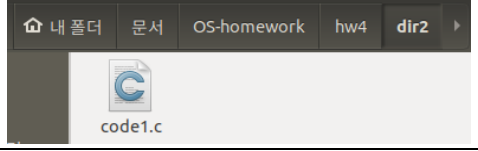
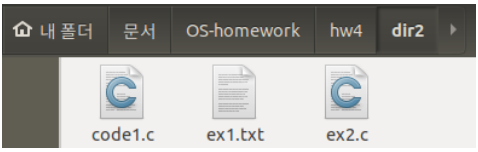
```
yesol@yesol-virtual-machine:~/문서/OS-homework/hw4$ ./copy code1.c code2.c
>> start copying: regular file -> regular file
>> Start copying regular file...
>> Finish copying regular file :0
```

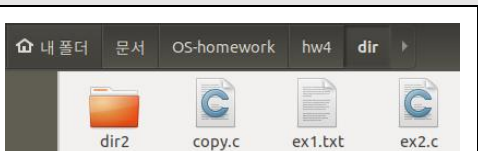
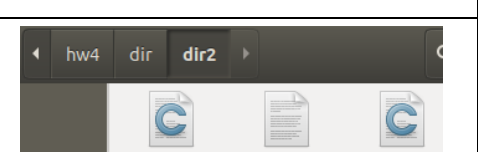
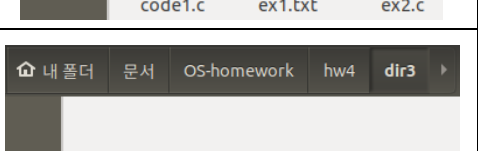
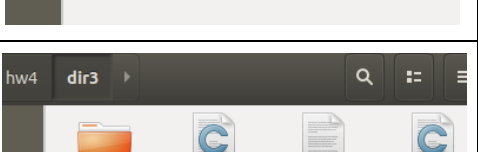
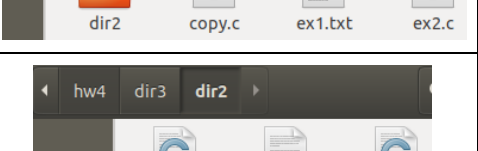
< 실행 후 >



2) src와 dst가 모두 directory :

src directory (file & sub-directory) → dst directory로 copy

CASE 1. sub-directory 가 없는 디렉토리 복사	
dir	
dir2	
copy 후 dir2	
<pre>yesol@yesol-virtual-machine:~/문서/OS-homework/hw4\$./copy dir dir2 >> start copying: directory file -> directory file >> Start copying directory file... >> copying directory file: .. >> copying regular file: ex2.c ./dir/ex2.c >> Start copying regular file... >> Finish copying regular file :D >> copying regular file: ex1.txt ./dir/ex1.txt >> Start copying regular file... >> Finish copying regular file :D >> copying directory file: . >> Finish copying directory file :D</pre> <p>ex2.c 파일 복사 → ex1.txt 파일 복사 (./dir/ex2.c & ./dir/ex1.txt)</p>	

CASE 2. sub-directory 가 있는 디렉토리 복사	
dir	
dir 속 dir2	
dir3	
copy 후 dir3	
dir3 속 dir2	
<pre>yesol@yesol-virtual-machine:~/문서/OS-homework/hw4\$./copy dir dir3 >> start copying: directory file -> directory file >> except file: .. >> copying regular file: copy.c >> Start copying regular file... >> Finish copying regular file :D >> copying regular file: ex2.c >> Start copying regular file... >> Finish copying regular file :D >> copying regular file: ex1.txt >> Start copying regular file... >> Finish copying regular file :D >> except file: . >> copying directory file: dir2 >> input directory: ./dir/dir2/ output directory: ./dir3/dir2 >> copy sub-directory ./dir/dir2. >> copying regular file: code1.c >> Start copying regular file... >> Finish copying regular file :D >> except file: .. >> copying regular file: ex2.c >> Start copying regular file... >> Finish copying regular file :D >> copying regular file: ex1.txt >> Start copying regular file... >> Finish copying regular file :D >> except file: . >> Finish copying directory file [./dir/dir2] :D >> Finish copying directory file [dir] :D</pre>	

dir/copy.c 파일 복사 → dir/ex2.c 파일 복사 → dir/ex1.txt 파일 복사 → ./dir/dir2/ 디렉토리 복사 시작, 복사되는 경로는 ./dir3/dir2 → dir/dir2/code1.c 복사 → dir/dir2/ex2.c 복사 → dir/dir2/ex1.txt 복사 → ./dir/dir2 디렉토리 복사 끝 → dir 디렉토리 복사 끝

3) src가 directory이고 dst가 regular file → 오류 메시지 출력

```
yesol@yesol-virtual-machine:~/문서/OS-homework/hw4$ ./copy dir code1.c
>> Copy Failed. [src] is directory file. But [dst] is regular file...T.T
```

4) src가 regular file dst가 directory

```
yesol@yesol-virtual-machine:~/문서/OS-homework/hw4$ ./copy copy.c dir
>> start copying: regular file -> directory file
./copy.c
>> Start copying regular file...
>> Finish copying regular file :D
```



명령창이 열려 있는 현재 디렉토리인 hw4에 있는 copy.c 파일을 dir에 보냈다.

5) 명령어 틀렸을 때

```
yesol@yesol-virtual-machine:~/문서/OS-homework/hw4$ ./copy dir
Usage : ./copy src dst
```