

## < 시스템 프로그래밍 -우리가 자주 사용하는 ls 명령 구현하기 >

### <코드 설명>

#### <함수1 - DirName : LS>

dirent 구조체를 사용하여 디렉토리 파일을 읽어온다.

하지만 readdir 함수를 사용해서 디렉토리 파일을 읽어오면 숨김파일도 같이 넘어온다.

숨김파일들은 기본적으로 앞에 .을 붙이고 있기에 dirent가 가리키는 구조의 d\_name의 첫번째 인덱스 값이 .일 경우 즉, 파일명의 맨 앞이 .으로 시작할 경우를 제외시켜주고 나머지를 출력하면 현재 디렉토리의 파일들을 가져올 수 있다.

#### <함수2 - DirIntro : LS -l>

위와 같은 방법으로 현재 디렉토리에 있는 파일들을 가져오며 가져온 파일명을 이용하여 stat 함수를 사용하여 상세 정보를 가져온다.

디렉토리 권한 정보에 대해서는 상수값으로 나오기 때문에 st\_mode를 사용하여 비교하여 정보를 가져와야한다. S\_IFMT 상수의 값이 0xF000 이므로 st\_mode의 값과 AND 연산하면 파일 종류의 상수값만 남기 때문에 그 상수값을 이용하여 해당 파일의 종류를 가져온다.

그 뒤를 이어서 st\_mode와 소유자/그룹/기타 사용자의 읽기/쓰기/실행 권한의 상수값과 비교하여 해당 권한에 따라 가질 경우 r/w/x를 출력하며 아닐 경우 -를 사용하여 나타낸다.

link 갯수는 st\_nlink, 사용자와 그룹의 id는 st\_uid와 st\_gid를 사용하고 사이즈는 st\_size를 사용하면 된다.

수정 시간은 st\_mtime을 사용해야하지만 그대로 사용할 경우 숫자와 문자로 나온다. 때문에 tm이라는 구조체를 사용하여 tm 구조체에 저장돼있는 정보들을 활용하여 LS -l에 필요한 데이터만 가져올 것이다. localtime 함수를 사용하여 tm 구조체를 가진 구조체 t에 수정 시간을 가리키는 st\_mtime의 값을 넣고 %d를 사용하여 필요한 값인 month, day, hour, minutes를 가져오면 된다. 이때 중요한건 tm 구조체에 mon는 기본적으로 0부터 11까지를 나타내므로 우리가 원하는 달이 나오려면 출력값에 +1을 해줘야한다. 그렇게 할 경우 우리가 원하는 수정한 날짜에 대한 월, 일, 시, 분에 대한 정보를 가져올 수 있다.

마지막으로 파일의 이름은 dent->d\_name을 사용하여 가져온다.

<함수 3 - sizeRt : LS -s 500>

크기가 500 이상인 파일의 개수를 저장할 num,

파일의 크기를 저장할 int형 배열 size, 파일의 inode 번호를 저장할 int형 배열 nodeNum,

위 배열에 값을 넣을시 다음 인덱스로 넘어가기 위한 변수 count.

위와 같은 변수들을 선언해준 뒤 먼저 디렉토리를 읽어오면서 stat함수를 사용하여 파일의 크기 st\_size가 500 이상일시 num++, 파일의 크기를 저장할 size 배열에 현재 파일의 st\_size의 값을 저장, 파일의 inode값을 저장할 nodeNum 배열에 현재 파일의 d\_ino를 저장 후 배열의 다음 인덱스로 넘어가기 위한 count++을 해준다.

위와 같은 방식으로 진행되어 while문이 끝날 시 크기가 500이 넘는 파일의 개수와 해당 파일의 inode번호가 size, nodeNum 배열에 순서대로 알맞게 저장된다.

temp를 사용하여 크기순으로 정렬을 해주는데 이때 size와 nodeNum의 순서 또한 같아야하기 때문에 정렬할때마다 묶어서 정렬시켜준다.

그렇게할 경우 size배열에는 파일의 크기가 숫자만으로 정렬돼있으며 nodeNum 배열에는 크기 순으로 정렬된 파일의 inode 번호가 정렬돼 있게된다.

이제 while 문으로 다시 디렉토리를 읽으면서 inode 번호와 일치할 경우 해당 파일의 이름을 출력해주면 크기 순으로 정렬이 된다. 이때 중요한 것은 위에 첫번째 while문에서 readdir을 할 경우 디렉토리 내의 파일을 다 읽을 경우 NULL을 반환하기 때문에 두번째 while문을 통해 readdir을 할 경우 파일 정보를 제대로 반환하지 못한다. 때문에 rewinddir()을 사용하여 디렉토리 포인터를 다시 원래 처음으로 돌려준다.

마지막으로 i가 0부터 num(크기가 500이 넘는 파일의 개수)까지 디렉토리를 읽으며 nodeNum의 값이 읽어오는 d\_ino의 값과 같을시 해당 파일의 이름 d\_name을 출력해주면 된다.

물론 이때 또한 i가 증가할때마다 while문이 계속해서 반복하면서 i값이 증가할때마다 readdir 함수가 NULL을 반환하기 때문에 i가 증가하기전 while문이 끝난 후 꼭 rewinddir()을 해준다.

<함수 4 - sizeRtAll : LS -ls 500>

함수 3의 로직을 이용하여 출력 형태만 함수 2와 같은 방식으로 출력해주면 된다.

<함수 5 - printDir : LS -F>

```
if st_mode의 파일 종류가 디렉토리가 아닐 시 {  
    if 사용자, 그룹, 기타 사용자의 실행 권한이 있을시 네임 출력 후 '*' 출력  
    else 없을시 네임 그대로 출력  
}  
  
else 파일 종류가 디렉토리일 시 네임 출력 후 '/' 출력
```

<함수 6 - prinSub : LS -R>

먼저 현재 내가 열고있는 파일의 디렉토리 경로를 알기 위해 getcwd함수를 사용한다. getcwd를 통해 디렉토리의 경로를 path라는 배열에 넣어주기 위해 path배열을 선언하는데 이때 크기는 버퍼의 최대 크기를 넣어주기 위해 PATH\_MAX 값을 사용하며 limit.h 헤더파일을 포함해준다.

path에 현재 열린 파일의 디렉토리 경로가 저장됐으면 열린 디렉토리 속 파일 중 서브 파일을 포함한 디렉토리를 열어주기 위해 openSubDir 함수를 만든다. 이때 넘겨주는 매개변수는 현재 열린 파일 경로인 path를 넘겨준다.

openSubDir 함수 속에서는 넘어온 경로를 통해 현재 경로 속에 포함된 디렉토리 파일을 찾고 찾은 디렉토리의 경로를 추가해 속에 있는 서브 파일들을 출력해줄 것이다. 넘어온 경로를 읽으면서 stat 함수를 통해 st\_mode가 디렉토리인 경우에 찾은 디렉토리의 경로를 저장해줄 char 배열 pathDir[PATH\_MAX] 배열을 선언해준다. 찾은 디렉토리의 파일을 열기 위해 현재 열린파일 경로 + / + 찾은 디렉토리 경로를 해줘야한다. sprintf함수는 문자열 두개를 합쳐서 배열에 넣어주는 함수이다. sprintf(pathDir, "%s/%s", path, dent->d\_name)을 사용하여 두 문자열을 합쳐 디렉토리 속 서브 디렉토리의 경로를 pathDir에 넣어준다. 그리고 서브 디렉토리의 파일 속 내용을 출력해줄 subDir 함수를 만든다. 이때 넘겨주는 매개변수는 만든 서브 디렉토리의 경로 pathDir을 넘겨준다.

subDir 함수 속에서는 디렉토리의 경로를 가져와 내용들을 출력하며 또다른 디렉토리가 있을시 재귀호출을 통해 또다시 출력해주는 기능을 구현했다. 넘어온 서브 디렉토리의 파일경로를 통해 파일을 열어 내용이 없을시 에러를 내며 내용이 있을시 서브 디렉토리 속 디렉토리의 정보를 담을 pathDir[PATH\_MAX] 배열을 선언해준 뒤 sprintf 함수를 통해 현재 열린 파일의 경로와 열린 파일 속 서브 디렉토리의 경로들을 합쳐 pathDir 배열에 넣어주고 서브 디렉토리의 경로를 출력해준다. 그리고나서 이 subDir 함수를 재귀호출을 사용하여 서브 디렉토리 속 서브 디렉토리가 있을시 계속해서 그 내용들을 출력해주게 한다.

#### <함수 7 - prinRever : LS -r>

역순으로 할 시 알파벳으로 시작하는 파일들을 출력하기 전에 한글로 시작하는 파일을 출력해주기 위해 아스키 코드를 이용해 dent->d\_name[0]의 값이 아스키 코드 값 알파벳 A(65)~Z(90)와 a(97)~z(122)이 아닐 시 파일을 출력해준다.

한글로 시작하는 파일의 출력이 끝나면 알파벳의 역순으로 출력하기 위해서 아스키 코드를 사용했다. 소문자 z(122)부터 소문자 a(97)까지를 for문을 통하여 dent->d\_name의 첫번째로 오는 문자가 일치할시 출력하도록 했다. 이때 또한 for문안에 while문이 들어가 readdir을 하므로 while문이 끝날시 rewinddir을 통해 디렉토리의 시작 포인터값을 처음으로 다시 돌려준다. 역순으로 할 시 소문자가 끝나면 대문자가 나오기 때문에 소문자 알파벳으로 시작하는 파일들의 출력이 끝날시 대문자 Z(90)부터 대문자 A(65)까지를 다시 for문과 while문을 통해 역순으로 출력해준다. 마찬가지로 for문 안에 있는 while문이 끝날때마다 readdir의 시작 포인터 값을 처음으로 돌려주기 위해 rewinddir을 사용해준다.

#### <main 함수>

argc의 값이 1개일 경우, 2개일 경우, 3개일 경우, 4개일 경우를 나누고 각 조건 속에서 argv[i]의 값과 원하는 명령의 문자열을 strcmp 함수를 통해 비교하여 참일시 조건에 해당하는 함수들을 호출해준다.

#### <코드>

```
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <dirent.h>
#include <string.h>
#include <time.h>
#include <limits.h>
#define sizeNum 30
void subDir(char *path);

void DirName() {
    DIR *dp;
    struct dirent *dent;
```

```

dp = opendir(".");

int j = 0;
while((dent = readdir(dp))) {
    if (dent->d_name[0] == '.') continue;
    else {
        printf("%s ", dent ->d_name);
        j++;
        if(j % 4 == 0) printf("\n");
    }
}
printf("\n");
closedir(dp);
}

void DirIntro() {
    DIR *dp;
    struct dirent *dent;

    dp = opendir(".");
    while((dent = readdir(dp))) {
        if (dent->d_name[0] == '.') continue;
        else {
            struct stat statbuf;
            stat(dent->d_name, &statbuf);

            int kind = statbuf.st_mode & S_IFMT;
            switch(kind) {
                case S_IFREG:
                    printf("-");
                    break;
                case S_IFDIR:
                    printf("d");
                    break;
                case S_IFLNK:
                    printf("l");

```

```

                                break;
        case S_IFCHR:
            printf("c");
            break;
        case S_IFBLK:
            printf("b");
            break;
        case S_IFSOCK:
            printf("s");
            break;
        case S_IFIFO:
            printf("p");
            break;
    }
    if((statbuf.st_mode & S_IRUSR) != 0) printf("r");
    else printf("-");
    if((statbuf.st_mode & S_IWUSR) != 0) printf("w");
else printf("-");
    if((statbuf.st_mode & S_IXUSR) != 0) printf("x");
    else printf("-");
    if((statbuf.st_mode & S_IRGRP) != 0) printf("r");
    else printf("-");
    if((statbuf.st_mode & S_IWGRP) != 0) printf("w");
    else printf("-");
    if((statbuf.st_mode & S_IXGRP) != 0) printf("x");
    else printf("-");
    if((statbuf.st_mode & S_IROTH) != 0) printf("r");
    else printf("-");
    if((statbuf.st_mode & S_IWOTH) != 0) printf("w");
    else printf("-");
    if((statbuf.st_mode & S_IXOTH) != 0) printf("x");
    else printf("-");
    printf(" ");

    printf("%o ", (unsigned int)statbuf.st_nlink);
    printf("%d ", (unsigned int)statbuf.st_uid);
    printf("%d ", (int)statbuf.st_gid);

```

```

        printf("%d ", (int)statbuf.st_size);
        struct tm *t;
        t = localtime(&statbuf.st_mtime);
        printf("%d 월 %d 일 %d:%d ", t->tm_mon+1, t->tm_mday, t->tm_hour, t-
>tm_min);

```

```

        printf("%s ", dent ->d_name);
    }
    printf("\n");
}
closedir(dp);

}
int num = 0;
void sizeRt() {
    DIR *dp;
    struct dirent *dent;

    dp = opendir(".");

    int j = 0;
    int count = 0;
    int size[sizeNum] = {0};
    int nodeNum[sizeNum] = {0};

    while((dent = readdir(dp))) {
        if (dent->d_name[0] == '.') continue;
        else {
            struct stat statbuf;
            stat(dent->d_name, &statbuf);

            if (((int)statbuf.st_size) >= 500) {
                num++;
                size[count] = statbuf.st_size;
                nodeNum[count] = dent->d_ino;
                count++;
            }
        }
    }
}

```

```

        }
    }
}
int x, y, temp, temp2;
for(x=1; x<num; x++) {
    for(y=0; y<=x; y++) {
        if (size[y] <= size[x]) {
            temp = size[x];
            temp2 = nodeNum[x];
            size[x] = size[y];
            nodeNum[x] = nodeNum[y];
            size[y] = temp;
            nodeNum[y] = temp2;
        }
    }
}
rewinddir(dp);
int i=0;

for(i=0; i<num; i++) {
    while((dent = readdir(dp))) {
        struct stat statbuf;
        stat(dent->d_name, &statbuf);

        if (dent->d_name[0] == '.') continue;
        else {
            if(nodeNum[i] == (dent->d_ino)) {
                printf("%s", dent->d_name);
                printf("(%d) ", (int)statbuf.st_size);
                j++;
                if (j % 4 == 0) printf("\n");
            }
        }
    }
    rewinddir(dp);
}
printf("\n");

```



```

        closedir(dp);
    }

void sizeRtAll() {
    DIR *dp;
    struct dirent *dent;

    dp = opendir(".");

    int j = 0;
    int count = 0;
    int size[sizeNum] = {0};
    int nodeNum[sizeNum] = {0};

    while((dent = readdir(dp))) {
        if (dent->d_name[0] == '.') continue;
        else {
            struct stat statbuf;
            stat(dent->d_name, &statbuf);

            if (((int)statbuf.st_size) >= 500) {
                num++;
                size[count] = statbuf.st_size;
                nodeNum[count] = dent->d_ino;
                count++;
            }
        }
    }

    int x, y, temp, temp2;
    for(x=1; x<num; x++) {
        for(y=0; y<=x; y++) {
            if (size[y] <= size[x]) {
                temp = size[x];
                temp2 = nodeNum[x];
                size[x] = size[y];
                nodeNum[x] = nodeNum[y];
                size[y] = temp;
            }
        }
    }
}

```

```

        nodeNum[y] = temp2;
    }
}
rewinddir(dp);
int i=0;
for(i=0; i<num; i++) {
    while((dent = readdir(dp))) {
        struct stat statbuf;
        stat(dent->d_name, &statbuf);

        if (dent->d_name[0] == '.') continue;
        else {
            if(nodeNum[i] == (dent->d_ino)) {
                int kind = statbuf.st_mode & S_IFMT;
                switch(kind) {
                    case S_IFREG:
                        printf("-");
                        break;
                    case S_IFDIR:
                        printf("d");
                        break;
                    case S_IFLNK:
                        printf("l");
                        break;
                    case S_IFCHR:
                        printf("c");
                        break;
                    case S_IFBLK:
                        printf("b");
                        break;
                    case S_IFSOCK:
                        printf("s");
                        break;
                    case S_FIFO:
                        printf("p");
                        break;
                }
            }
        }
    }
}

```

```

    }

    if((statbuf.st_mode & S_IRUSR) != 0) printf("r");
else printf("-");
if((statbuf.st_mode & S_IWUSR) != 0) printf("w");
else printf("-");
if((statbuf.st_mode & S_IXUSR) != 0) printf("x");
else printf("-");
if((statbuf.st_mode & S_IRGRP) != 0) printf("r");
else printf("-");
if((statbuf.st_mode & S_IWGRP) != 0) printf("w");
else printf("-");
if((statbuf.st_mode & S_IXGRP) != 0) printf("x");
else printf("-");
if((statbuf.st_mode & S_IROTH) != 0) printf("r");
else printf("-");
    if((statbuf.st_mode & S_IWOTH) != 0) printf("w");
else printf("-");
if((statbuf.st_mode & S_IXOTH) != 0) printf("x");
else printf("-");
printf(" ");

    printf("%o ", (unsigned int)statbuf.st_nlink);
printf("%d ", (unsigned int)statbuf.st_uid);
printf("%d ", (int)statbuf.st_gid);
printf("%d ", (int)statbuf.st_size);
struct tm *t;
t = localtime(&statbuf.st_mtime);
printf("%d 월 %d 일 %d:%d ", t->tm_mon+1, t->tm_mday, t->tm_hour, t->tm_min);

```

```

printf("%s ", dent ->d_name);
printf("\n");
}
}

```

```

}
rewinddir(dp);

```

```

    }
    printf("\n");
    closedir(dp);
}

void prinDir() {
    DIR *dp;
    struct dirent *dent;

    dp = opendir(".");

    int j = 0;
    while((dent = readdir(dp))) {
        if (dent->d_name[0] == '.') continue;
        else {
            struct stat statbuf;
            stat(dent->d_name, &statbuf);

            int kind = statbuf.st_mode & S_IFMT;

            if(kind != S_IFDIR) {
                if((statbuf.st_mode & S_IXUSR) != 0 && (statbuf.st_mode &
S_IXGRP) != 0 && (statbuf.st_mode & S_IXOTH) != 0 ) {
                    printf("%s", dent ->d_name);
                    printf("* ");
                }

                else {
                    printf("%s ", dent ->d_name);
                }
            }
            else {
                printf("%s", dent ->d_name);
                printf("/ ");
            }
            j++;
            if(j % 4 == 0) printf("\n");
        }
    }
}

```

```

    }
    printf("\n");
    closedir(dp);
}

void openSubDir(char *path) {
    DIR *dp;
    struct dirent *dent;

    dp = opendir(path);

    if(dp == NULL) {
        perror("None Dir");
    }
    else {
        while(dent = readdir(dp)) {
            struct stat statbuf;
            stat(dent->d_name, &statbuf);

            if (dent->d_name[0] == '.') continue;
            else if ((statbuf.st_mode & S_IFMT) != S_IFDIR) continue;
            else {
                char pathDir[PATH_MAX];
                sprintf(pathDir, "%s/%s", path, dent->d_name);
                printf("Current Dir : %s\n", pathDir);
                subDir(pathDir);
            }
        }
        rewinddir(dp);
    }

    closedir(dp);
}

void subDir(char *path) {
    DIR *dp;

```

```

struct dirent *dent;

dp = opendir(path);

if(dp == NULL) {
    perror("None Dir");
}
else {
while(dent = readdir(dp)) {
    struct stat statbuf;
    stat(dent->d_name, &statbuf);

    if (dent->d_name[0] == '.') continue;
    else {
        char pathDir[PATH_MAX];
        sprintf(pathDir, "%s/%s",path,dent->d_name);
        printf("Current Dir : %s\n",pathDir);
        subDir(pathDir);
    }
}
rewinddir(dp);
}
printf("\n");
closedir(dp);
}

```

```

void prinSub() {
    DIR *dp;
    struct dirent *dent;

    dp = opendir(".");

    char path[PATH_MAX];
    if (getcwd(path, PATH_MAX)==NULL) {
        perror("getcwd error");
        exit(EXIT_FAILURE);
    }
}

```

```

    }
else {
    printf("Current Dir : %s\n",path);
    prinDir();
    printf("\n");
    openSubDir(path);
}

closedir(dp);
}

void prinRever() {
    if(
        rewinddir(dp);

    while((dent = readdir(dp))) {
        if (dent->d_name[0] == '.') continue;
        else if (91 >= dent->d_name[0] && dent->d_name[0] >= 65)
            continue;
        else if (122 >= dent->d_name[0] && dent->d_name[0] >= 97)
            continue;
        else {
            printf("%s\n", dent->d_name);
        }
    }
    rewinddir(dp);

    char ch1, ch2;
    int j = 0;
    for(ch2=122; ch2>=97; ch2--) {
        while((dent = readdir(dp))) {
            if (dent->d_name[0] == '.') continue;
            else {
                if (dent->d_name[0] == ch2) {
                    printf("%s ",dent->d_name);
                    j++;
                    if (j % 4 == 0) printf("\n");
                }
            }
        }
    }
}

```

```

        }
    }
}
rewinddir(dp);
}
for(ch1=90; ch1>=65; ch1--) {
    while((dent = readdir(dp))) {
        if (dent->d_name[0] == '.') continue;
        else {
            if (dent->d_name[0] == ch1) {
                printf("%s ",dent->d_name);
                j++;
                if(j % 4 == 0) printf("\n");
            }
        }
    }

}
rewinddir(dp);
}
closedir(dp);
}

```

```

int main(int argc, char *argv[]) {
    if (argc == 1 && argc < 2) {
        DirName();
    }
    else if (argc == 2) {
        if (strcmp(argv[1], "-l") == 0) {
            DirIntro();
        }
        else if (strcmp(argv[1], "-F") == 0) {
            prinDir();
        }
        else if (strcmp(argv[1], "-R") == 0) {
            prinSub();
        }
        else if (strcmp(argv[1], "-r") == 0) {

```



```

        prinRever();
    }

}

else if (argc == 3) {
    if (strcmp(argv[1], "-s") == 0 && strcmp(argv[2], "500") == 0) {
        sizeRt();
    }

    else if (strcmp(argv[1], "-ls") == 0 && strcmp(argv[2], "500") == 0) {
        sizeRtAll();
    }

    else if (strcmp(argv[1], "-d") == 0 && strcmp(argv[2], "-l") == 0) {
        prinDir();
    }

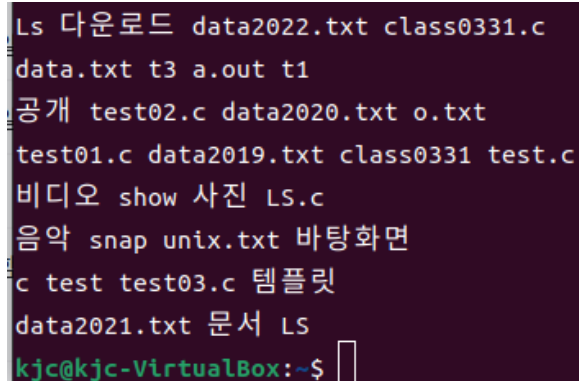
}

else if (argc == 4) {
    sizeRtAll();
}

return 0;
}

```

### <실행화면>



```

Ls 다운로드 data2022.txt class0331.c
data.txt t3 a.out t1
공개 test02.c data2020.txt o.txt
test01.c data2019.txt class0331 test.c
비디오 show 사진 LS.c
음악 snap unix.txt 바탕화면
c test test03.c 템플릿
data2021.txt 문서 LS
kjc@kjc-VirtualBox:~$

```

```
kjc@kjc-VirtualBox:~$ ./LS -l
-rwxrwxr-x 1 1000 1000 16169 4월 12일 1:37 Ls
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 다운로드
-rw-rw-r-- 1 1000 1000 67 3월 29일 3:2 data2022.txt
-rw-rw-r-- 1 1000 1000 336 4월 7일 11:1 class0331.c
-rw-rw-r-- 1 1000 1000 13 3월 28일 0:46 data.txt
-rwxrwxr-x 1 1000 1000 16096 4월 7일 11:23 t3
-rwxrwxr-x 1 1000 1000 20976 4월 13일 22:16 a.out
-rwxrwxr-x 1 1000 1000 20272 3월 30일 23:0 t1
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 공개
drwxrwxr-x 2 1000 1000 4096 4월 7일 11:0 test02.c
-rw-rw-r-- 1 1000 1000 67 3월 29일 3:1 data2020.txt
-rw-rw-r-- 1 1000 1000 7 4월 7일 10:27 o.txt
-rw-rw-r-- 1 1000 1000 6993 4월 13일 22:58 test01.c
-rw-rw-r-- 1 1000 1000 67 3월 30일 15:5 data2019.txt
-rwxrwxr-x 1 1000 1000 16056 3월 31일 11:26 class0331
-rw-rw-r-- 1 1000 1000 308 3월 28일 1:28 test.c
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 비디오
-rwxrwxr-x 1 1000 1000 20272 3월 30일 23:41 show
drwxr-xr-x 3 1000 1000 4096 3월 30일 15:3 사진
-rw-rw-r-- 1 1000 1000 14173 4월 13일 22:42 LS.c
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 음악
drwx----- 4 1000 1000 4096 3월 30일 23:38 snap
-rw-rw-r-- 1 1000 1000 12 4월 7일 11:46 unix.txt
```

```
kjc@kjc-VirtualBox:~$ ./LS -s 500
LS(20976) a.out(20976) show(20272) t1(20272)
Ls(16169) test(16136) t3(16096) class0331(16056)
LS.c(14173) test01.c(6993) 문서(4096) 템플릿(4096)
바탕화면(4096) snap(4096) 음악(4096) 사진(4096)
비디오(4096) test02.c(4096) 공개(4096) 다운로드(4096)
```

```
kjc@kjc-VirtualBox:~$
```

```
kjc@kjc-VirtualBox:~$ ./LS -l -s 500
-rwxrwxr-x 1 1000 1000 20976 4월 13일 22:16 LS
-rwxrwxr-x 1 1000 1000 20976 4월 13일 22:16 a.out
-rwxrwxr-x 1 1000 1000 20272 3월 30일 23:41 show
-rwxrwxr-x 1 1000 1000 20272 3월 30일 23:0 t1
-rwxrwxr-x 1 1000 1000 16169 4월 12일 1:37 Ls
-rwxrwxr-x 1 1000 1000 16136 3월 28일 1:3 test
-rwxrwxr-x 1 1000 1000 16096 4월 7일 11:23 t3
-rwxrwxr-x 1 1000 1000 16056 3월 31일 11:26 class0331
-rw-rw-r-- 1 1000 1000 14173 4월 13일 22:42 LS.c
-rw-rw-r-- 1 1000 1000 6993 4월 13일 22:58 test01.c
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 문서
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 템플릿
drwxr-xr-x 3 1000 1000 4096 4월 13일 23:1 바탕화면
drwx----- 4 1000 1000 4096 3월 30일 23:38 snap
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 음악
drwxr-xr-x 3 1000 1000 4096 3월 30일 15:3 사진
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 비디오
drwxrwxr-x 2 1000 1000 4096 4월 7일 11:0 test02.c
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 공개
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 다운로드
```

```
kjc@kjc-VirtualBox:~$ ./LS -ls 500
-rwxrwxr-x 1 1000 1000 20976 4월 13일 22:16 LS
-rwxrwxr-x 1 1000 1000 20976 4월 13일 22:16 a.out
-rwxrwxr-x 1 1000 1000 20272 3월 30일 23:41 show
-rwxrwxr-x 1 1000 1000 20272 3월 30일 23:0 t1
-rwxrwxr-x 1 1000 1000 16169 4월 12일 1:37 Ls
-rwxrwxr-x 1 1000 1000 16136 3월 28일 1:3 test
-rwxrwxr-x 1 1000 1000 16096 4월 7일 11:23 t3
-rwxrwxr-x 1 1000 1000 16056 3월 31일 11:26 class0331
-rw-rw-r-- 1 1000 1000 14173 4월 13일 22:42 LS.c
-rw-rw-r-- 1 1000 1000 6993 4월 13일 22:58 test01.c
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 문서
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 템플릿
drwxr-xr-x 3 1000 1000 4096 4월 13일 23:1 바탕화면
drwx----- 4 1000 1000 4096 3월 30일 23:38 snap
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 음악
drwxr-xr-x 3 1000 1000 4096 3월 30일 15:3 사진
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 비디오
drwxrwxr-x 2 1000 1000 4096 4월 7일 11:0 test02.c
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 공개
drwxr-xr-x 2 1000 1000 4096 3월 28일 0:29 다운로드

kjc@kjc-VirtualBox:~$
```

```
kjc@kjc-VirtualBox:~$ ./LS -F
Ls* 다운로드/ data2022.txt class0331.c
data.txt t3* a.out* t1*
공개/ test02.c/ data2020.txt o.txt
test01.c data2019.txt class0331* test.c
비디오/ show* 사진/ LS.c
음악/ snap/ unix.txt 바탕화면/
c test* test03.c 템플릿/
data2021.txt 문서/ Ls*
```

```
kjc@kjc-VirtualBox:~$ ./LS -d -l
Ls* 다운로드/ data2022.txt class0331.c
data.txt t3* a.out* t1*
공개/ test02.c/ data2020.txt o.txt
test01.c data2019.txt class0331* test.c
비디오/ show* 사진/ LS.c
음악/ snap/ unix.txt 바탕화면/
c test* test03.c 템플릿/
data2021.txt 문서/ Ls*
kjc@kjc-VirtualBox:~$
```

```
kjc@kjc-VirtualBox:~$ ./LS -R
Current Dir : /home/kjc
Ls* 다운로드/ data2022.txt class0331.c
data.txt t3* a.out* t1*
공개/ test02.c/ data2020.txt o.txt
test01.c data2019.txt class0331* test.c
비디오/ show* 사진/ LS.c
음악/ snap/ unix.txt 바탕화면/
c test* test03.c 템플릿/
data2021.txt 문서/ Ls*
```

```
Current Dir : /home/kjc/다운로드
```

```
Current Dir : /home/kjc/공개
```

```
Current Dir : /home/kjc/test02.c
```

```
Current Dir : /home/kjc/비디오
```

```
Current Dir : /home/kjc/사진/스크린샷
Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-04-13 23-09-19.png
None Dir: Not a directory

Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-04-13 23-10-54.png
None Dir: Not a directory

Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-03-30 15-03-21.png
None Dir: Not a directory

Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-03-30 23-44-37.png
None Dir: Not a directory

Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-04-13 23-10-15.png
None Dir: Not a directory

Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-04-13 23-11-22.png
None Dir: Not a directory

Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-03-30 23-42-20.png
None Dir: Not a directory

Current Dir : /home/kjc/사진/스크린샷/스크린샷 2023-04-13 23-10-36.png
None Dir: Not a directory
```

```
Current Dir : /home/kjc/음악
```

```
Current Dir : /home/kjc/snap
```

```
Current Dir : /home/kjc/snap/firefox
```

```
Current Dir : /home/kjc/snap/firefox/2487
```

```
Current Dir : /home/kjc/snap/firefox/current
```

```
Current Dir : /home/kjc/snap/firefox/common
```

```
Current Dir : /home/kjc/snap/firefox/2559
```

```
Current Dir : /home/kjc/snap/snapd-desktop-integration
```

```
Current Dir : /home/kjc/snap/snapd-desktop-integration/49
```

```
Current Dir : /home/kjc/snap/snapd-desktop-integration/current
```

```
Current Dir : /home/kjc/snap/snapd-desktop-integration/common
```

```
Current Dir : /home/kjc/snap/snapd-desktop-integration/57
```

```
Current Dir : /home/kjc/바탕화면
```

```
Current Dir : /home/kjc/바탕화면/새 폴더
```

```
Current Dir : /home/kjc/바탕화면/LS.txt
```

```
None Dir: Not a directory
```

```
Current Dir : /home/kjc/바탕화면/test01.txt
```

```
None Dir: Not a directory
```

```
Current Dir : /home/kjc/템플릿
```

```
Current Dir : /home/kjc/문서
```

```
kjc@kjc-VirtualBox:~$
```

```
kjc@kjc-VirtualBox:~$ ./LS -r
다운로드
공개
비디오
사진
음악
바탕화면
템플릿
문서
unix.txt t3 t1 test02.c
test01.c test.c test test03.c
show snap o.txt data2022.txt
data.txt data2020.txt data2019.txt data2021.txt
class0331.c class0331 c a.out
Ls LS.c LS kjc@kjc-VirtualBox:~$
```