<겨울방학 멘토링 2조 - 디렉토리 탐색 프로그램>

1. 개요

프로그램이 위치한 디렉토리로부터 하위 디렉토리로 탐색을 하여 원하는 이름의 파일을 찾아 해당 파일의 경로를 출력하거나, 입력받은 디렉토리의 총 크기를 구하고 내부 구조를 트리 형태로 출력하는 기능을 갖는 프로그램 구현하기.

2. 명세

<프로그램 입력>

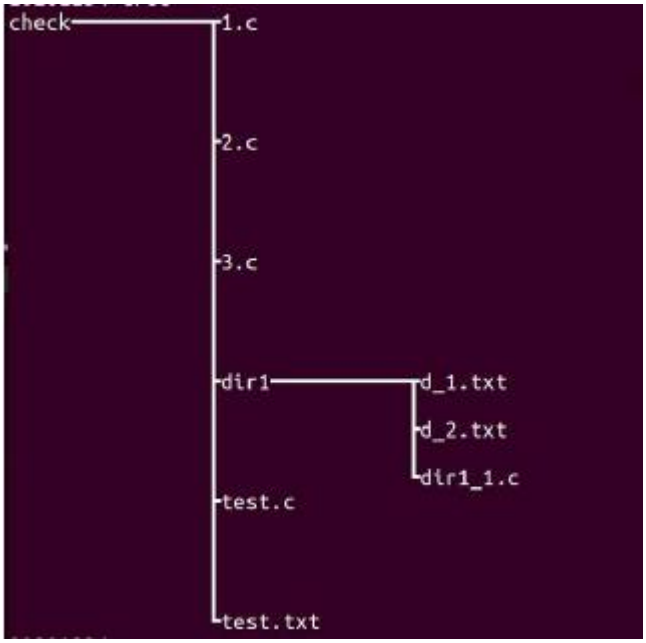
argv[1]: 총 크기를 구하고 싶은 디렉토리의 경로(또는 이름)

argv[2]: 찾고싶은 파일의 이름

<구현해야 할 기능>

1. opendir(), readdir()을 이용하여 디렉토리 내의 파일 탐색, 재귀적으로 함수를 호출하여 DFS or 큐를 이용하여 BFS방식으로 디렉토리 탐색. (트리에서의 깊이 우선 탐색 또는 너비 우선 탐색)
2. 디렉토리 탐색 과정에서 입력받은 이름의 파일을 찾았을 시 해당 파일의 경로를 출력.
3. 입력받은 이름의 디렉토리의 크기를 구하여 출력.
4. 입력받은 이름의 디렉토리의 내부 구조를 트리 형태로 출력.

ex)



※ Linux는 디렉토리의 기본 크기가 4096bytes로 지정되어있기 때문에 직접 디렉토리 안의 파일의 크기를 합해서 크기를 구하는 식으로 디렉토리의 크기 구하기를 구현, ".", ".."은 제외.

3. 참고 함수 및 구조체

<함수>

#include <sys/stat.h>

#include <sys/types.h>

#include <unistd.h>

int stat(const char *pathname, struct stat *statbuf);

⇨ pathname에 해당하는 파일의 정보를 statbuf에 저장해주는 함수이다. 해당 파일에 대한 거의 모든 정보를 담고 있으므로 아주 유용한 함수.

#include <sys/types.h>

#include <dirent.h>

DIR *opendir(const char *name);

⇨ 파일 여는 것처럼 디렉토리 여는 역할을 하는 놈이다.

#include <dirent.h>

struct dirent *readdir(DIR *dp);

⇨ 주로 while문 안에 사용하며, DIR 포인터가 가리키는 디렉토리 안의 파일들을 순차적으로 읽어오는 놈이다.

#include <unistd.h>

char *getcwd(char *buf, size_t size);

⇨ 현재 작업 디렉토리의 이름을 size만큼 buf에 복사한다. 리눅스 명령어 pwd와 같은 기능을 한다.

#include <sys/types.h>

#include <dirent.h>

void rewinddir(DIR *dp);

⇨ 디렉토리 읽기 위치를 처음으로 이동시킨다. while문에서 readdir로 디렉토리 내부를 모두 읽을 때 이걸 쓰면 첫 놈부터 다시 읽을 수 있다.

<구조체>

```
struct stat {
        dev_t           st_dev;         /*ID of device containing file */
        ino_t           st_ino;          /*inode number*/
        mode_t          st_mode;      /*파일 종류 및 권한에 관한 정보*/
        nlink_t         st_nlink;       /*number of hard links*/
        uid_t           st_uid;          /*user ID of owner*/
        gid_t           st_gid;          /*group ID of owner*/
        dev_t           st_rdev;        /*device ID (if special file)*/
        off_t           st_size;         /*total size, in byte*/
        blksize_t       st_blksize;     /*blocksize for file system I/O*/
        blkcnt_t        st_blocks;      /*number of 512B blocks allocated*/
        time_t;         st_atime;       /*time of last access*/
        time_t;         st_mtime;       /*time of last modification*/
        time_t          st_xtime;       /*time of last status change*/
};
```

```
struct dirent
{
    long d_ino;                         /* 아이노드 */
    off_t d_off;                        /* dirent 의 offset */
    unsigned short d_reclen;            /* d_name 의 길이 */
    char d_name [NAME_MAX+1];   /* 파일 이름(없다면 NULL로 종료) */
}
```