#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <dirent.h>

#include <sys/stat.h>

#include <sys/types.h>

int main(int argc, char\* argv[]) {

int argc : main()함수에 전달되는 데이터의 개수

char\* argv[] :main()함수에 전달되는 실제적인 데이터로 char형 포인터 배열로 구성

struct dirent\* dent;

디렉토리 파일에 속하는 파일들을 관리하는 dirent 구조체로 정의

struct stat buf;

디렉토리 파일 정보를 저장하는 buf 구조체로 정의

int kind;

파일의 정보를 상수로 계산하기 위한

char path[BUFSIZ];

파일명과 파일에 대한 경로를 나타내주기 위한 path

if (argc != 2) {

printf("Error, Can not open directory.");

exit(1);

}

argc != 2

입력받은 데이터의 개수가 2개가 아닐 경우 디렉토리를 입력받지 못한 것과 같음

exit(1)

1를 반환하여 현재 프로그램 완전히 종료

if (argc == 2) {

입력받은 데이터의 개수가 2개일 경우 디렉토리를 입력받은 것과 같음

DIR\* dp = opendir(argv[1]);

디렉터리 포인터 DIR\* dp 선언

argv[1]에 입력받은 디렉토리를 opendir()함수를 이용하여 열어준다.

if (dp == NULL) {

perror("Error Can not approach directory.");

exit(1);

}

디렉토리가 비어있을 경우 디렉토리를 여는 것에 실패했다는 오류문 출력

exit(1) : 1를 반환하여 현재 프로그램을 완전히 종료

while (dent = readdir(dp))

{

if (dent->d\_name[0] == '.') continue;

while문을 이용하여 현재 디렉토리의 파일들을 읽는다.

if : .과 ..를 제외한 항목의 정보를 읽어준다.

else { //.과..파일이 아닐 경우

sprintf(path, "%s/%s", argv[1], dent->d\_name);

디렉토리명과 읽어온 항목의 이름을 연결하여 경로를 만들어준다.

stat(path, &buf);

stat()함수로 path 정보를 검색하면 상세한 내용을 알 수 있다.

kind = buf.st\_mode & S\_IFMT;

st\_mode와 S\_IFMT를 AND 연산하여 파일의 종류 탐색

switch (kind){

case S\_IFDIR:

printf("%s is Directory.\n", dent->d\_name); break;

case S\_IFREG:

printf("%s is File.\n", dent->d\_name);

break;

} // switch 종료

switch ()를 이용하여 case에 맞을 경우 해당 실행문들을 실행한다.

S\_IFDIR : 디렉토리 파일일 경우

S\_IFREG : 일반 파일일 경우

} //else문 종료

} //while문 종료

closedir(dp);

closedir()함수를 이용하여 열었던 dp 디렉토리를 닫아준다.

}

return 0;

main()함수를 종료하고 빠져나가기 위해 0을 return한다.

}