

파일 시스템 11

- 파일 처리 함수 2



컴퓨터소프트웨어학과 김병국 교수

학습목표



- □현재 작업 경로에 대한 추출 및 변환 기능을 프로그래밍 할 수 있다.
- □디렉토리 접근 및 엔트리에 대한 추출 기능을 프로그래밍 할 수 있다.
- □디렉토리 생성/변경/삭제 기능을 프로그래밍 할 수 있다.

목차



- □작업 경로 추출
- □작업 경로 전환
- □디렉토리 접근
- □디렉토리 관리
- □프로그래밍 실습

1. 작업 경로 추출 (1/2)



□ 현재 작업 경로 추출

- 함수 : getcwd(), getwd()
 - 현재 작업중인 경로의 위치를 추출
 - getwd의 경우 내부 정적 공간을 사용(re-entrance주의)
 - 인자:
 - *buf : 현재 경로 저장
 - size : buf의 크기
 - 결과 값:
 - 성공 : 저장된 데이터의 위치(buf의 포인터)
 - 실패 : NULL
- 함수 : get_current_dir_name()
 - 현재 작업중인 경로 추출(getcwd, getwd와 동일)
 - 이 함수의 호출은 내부에서 동적 메모리 할당이 이루어짐(malloc() 함수 호출)
 - 사용 후 반드시 메모리 환원을 해줘야 함(free() 함수 호출)

```
#include <unistd.h>
char *getcwd(char *buf, size t size);
char *getwd(char *buf);
char *get current dir name(void);
```

[함수의 프로토타입]



1. 작업 경로 추출 (2/2)

20 21 printf("getcwd: %s\n", strPwd);

printf("getwd: %s\n", strPwd);

if (getwd(strPwd) != NULL) {



```
□실습
                                                    for (int i = 0; i < 10; i++)
                                          23
                                          24
         #define GNU SOURCE
                                                        pPwd = get_current_dir_name();
                                          25
       □#include <stdio.h>
                                                        printf("%02d : %p : Get_XXX: %s\n", i, pPwd, pPwd);
                                          26
         #include <sys/stat.h>
                                                        free(pPwd); // Check point!!
         #include <sys/types.h>
                                         27
         #include <string.h>
                                          28
         #include <stdlib.h>
                                         29
         #include <unistd.h>
                                         30
                                                    return 0;
                                                                                         [파일명: getcwd.c]
                                          31
         int
        ∃main(void)
   10
   11
             char strPwd[BUFSIZ] = { 0, };
   12
   13
             char* pPwd = NULL;
   14
             if (getcwd(strPwd, BUFSIZ) != NULL) {
   15
```

2. 작업 경로 전환 (1/3)



□ 현재 작업 경로 전환

- 함수 : chdir()
 - 현재 작업중인 경로를 이동
 - 인자:
 - *path : 이동하고자 할 경로
 - 결과 값:
 - 성공 : 0
 - 실패 : -1
- 함수 : fchdir()
 - chdir() 함수와 동일한 기능을 수행
 - 인자:
 - fd : 파일 기술자
 - 결과 : chdir() 함수와 동일

```
#include <unistd.h>
int chdir(const char *path);
int fchdir(int fildes);
```

[함수의 프로토타입]



2. 작업 경로 전환 (2/3)

printf("Changing Path...\n");

20

21



```
□실습 1
                                                      chdir(argv[1]);
                                            23
       F#include <stdio.h>
                                            24
         #include <sys/stat.h>
                                                      if (getcwd(strPwd, BUFSIZ) != NULL) {
                                            25
        #include <sys/types.h>
                                                          printf("getcwd: %s\n", strPwd);
                                            26
        #include <string.h>
                                            27
        #include <unistd.h>
                                            28
                                            29
                                                      return 0;
         int
                                            30
       char strPwd[BUFSIZ] = { 0, };
   10
   11
   12
            if (argc != 2) {
   13
                fprintf(stderr, "Usage: %s <path>\n", argv[0]);
   14
                return -1;
   15
   16
            if (getcwd(strPwd, BUFSIZ) != NULL) {
   17
   18
                printf("getcwd: %s\n", strPwd);
   19
```

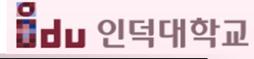
2. 작업 경로 전환 (3/3)

19

if (getcwd(strPwd, BUFSIZ) != NULL) {

nFd = open(argv[1], O_RDONLY);

printf("getcwd: %s\n", strPwd);



```
nFd = open(argv[1], O_RDONLY);
                                                               23
□실습 2
                                                                        printf("Changing Path...\n");
         ∃#include <stdio.h>
                                                               25
                                                               26
          #include <sys/stat.h>
                                                                        fchdir(nFd);
                                                               27
          #include <sys/types.h>
                                                              28
          #include <fcntl.h>
                                                                        if (getcwd(strPwd, BUFSIZ) != NULL) {
                                                              29
          #include <string.h>
                                                                            printf("getcwd: %s\n", strPwd);
                                                              30
          #include <unistd.h>
                                                              31
                                                              32
          int
                                                                        close(nFd);
                                                              33
        34
    10
                                                                        return 0;
                                                              35
              int nFd = -1;
    11
                                                              36
              char strPwd[BUFSIZ] = { 0, };
    12
    13
              if (argc != 2) {
    14
                  fprintf(stderr, "Usage: %s <path>\n", argv[0]);
    15
                 return -1;
    16
    17
    18
```

3. 디렉토리 접근 (1/4)



□ 디렉토리 열기

- 함수 : opendir() & fopendir()
 - 디렉토리 스트림 열기(접근)
 - 인자:
 - *name : 접근할 디렉토리 이름(또는 경로)
 - fd : open()에 의해 접근된 디렉토리의 파일 기술자
 - 결과 값:
 - 성공 : 디렉토리 포인터
 - 실패 : NULL
- 함수 : closedir()
 - 디렉토리 닫기
 - 인자 :
 - *dirp : 디렉토리 포인터
 - 결과 값:
 - 성공 : 0
 - 실패 : -1

```
#include <sys/types.h>
#include <dirent.h>

DIR *opendir(const char *name);

DIR *fdopendir(int fd);

int closedir(DIR *dirp);
```

[프로토타입]



3. 디렉토리 접근 (3/4)

□ 디렉토리내 엔트리 추출

- 함수 : readdir()
 - 파일 리스트를 출력
 - 현재 접근 오프셋(offset)에 해당하는 디렉토리의 엔트 리(entry) 정보를 반환

#include <dirent.h>

struct dirent *readdir(DIR *dirp);

[readdir() 함수의 프로토타입]

- 현재 오프셋의 값의 추출은 telldir() 함수를 이용
- 오프셋 값의 변경은 seekdir() 함수를 이용
- 인자:
 - *dirp : 읽을 디렉토리 포인터
- 결과 값:
 - 성공 : 디렉토리 엔트리 정보
 - 실패 : NULL

[dirent 구조체의 내용]

```
enum
{
    DT_UNKNOWN = 0,
    DT_FIFO = 1,
    DT_CHR = 2,
    DT_DIR = 4,
    DT_BLK = 6,
    DT_REG = 8,
    DT_LNK = 10,
    DT_SOCK = 12,
    DT_WHT = 14
};
```

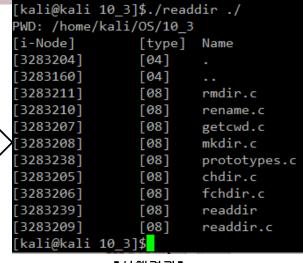
[파일의 타입]

3. 디렉토리 접근 (4/4)

Bdu 인덕대학교

```
□실습
```

```
F#include <stdio.h>
     #include <dirent.h>
     #include <sys/types.h>
     #include <sys/stat.h>
     #include <fcntl.h>
     #include <unistd.h>
    ⊟main(int argc, char* argv[])
         DIR* pDir;
11
         char pPwd[BUFSIZ] = { 0 };
12
         struct dirent* pEntry;
13
         pDir = opendir(".");
17
         getcwd(pPwd, BUFSIZ);
         printf("PWD: %s\n", pPwd);
         printf("[i-Node]\t[type]\tName\n");
         while (pEntry = readdir(pDir)) {
21
             printf("[%d]\t[%02d]\t%s\n",
22
23
                  pEntry->d_ino, pEntry->d_type, pEntry->d_name);
          closedir(pDir);
         return 0;
                                                      [작성 코드]
```



[실행결과]



11

5. 디렉토리 관리 (1/6)



□디렉토리 생성

- 함수: mkdir()
 - 지정한 이름의 디렉토리를 생성
 - 인자:
 - *pathname: 생성한 디렉토리명
 - mode : 접근 권한
 - 결과 값:
 - 성공 : 0
 - 실패 : -1

```
#include <sys/stat.h>
int mkdir(const char *pathname, mode_t mode);
```

[mkdir() 함수의 프로토타입]



5. 디렉토리 관리 (2/6)



□실습

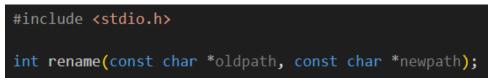
```
F#include <stdio.h>
     #include <sys/stat.h>
     #include <sys/types.h>
     #include <string.h>
     #include <unistd.h>
     int
    ⊟main(int argc, char* argv[])
         if (argc != 2) {
10
11
              fprintf(stderr, "Usage: %s <pathname>\n", argv[0]);
12
              return -1;
13
14
          if (mkdir(argv[1], 0755) == 0) {
15
16
              printf("Creating %s was completed.\n", argv[1]);
17
          } else {
              printf("Creating %s was failed.\n", argv[1]);
18
19
21
          return 0;
22
```

5. 디렉토리 관리 (3/6)



□디렉토리 변경

- 함수: rename()
 - 지정한 이름의 디렉토리를 새로운 이름으로 변경
 - 인자:
 - *oldpath : 디렉토리 원본
 - *newpath : 새로운 이름
 - 결과 값:
 - 성공 : 0
 - 실패 : -1



[rename() 함수의 프로토타입]

5. 디렉토리 관리 (4/6)



□실습

```
F#include <stdio.h>
      #include <sys/stat.h>
      #include <sys/types.h>
      #include <string.h>
      #include <unistd.h>
    □main(int argc, char* argv[])
         if (argc != 3) {
10
              fprintf(stderr, "Usage: %s <old path> <new path>\n", argv[0]);
11
12
              return -1;
13
14
          if (rename(argv[1], argv[2]) == 0) {
15
              printf("Renameing from %s to %s was completed.\n", argv[1], argv[2]);
17
          else {
18
19
              printf("Renameing from %s to %s was failed.\n", argv[1], argv[2]);
21
          return 0;
23
```

5. 디렉토리 관리 (5/6)



□디렉토리 삭제

- 함수: rmdir()
 - 지정한 이름의 디렉토리를 삭제
- 인자:
 - *pathname: 삭제할 디렉터리명
- 결과 값:
 - 성공 : 0
 - 실패 : -1



[rmdir() 함수의 프로토타입]

5. 디렉토리 관리 (6/6)



□실습

```
F#include <stdio.h>
     #include <sys/stat.h>
     #include <sys/types.h>
     #include <string.h>
     #include <unistd.h>
      int
    □main(int argc, char* argv[])
         if (argc != 2) {
10
11
              fprintf(stderr, "Usage: %s <pathname>\n", argv[0]);
12
              return -1;
13
14
         if (rmdir(argv[1]) == 0) {
15
16
              printf("Removing %s was completed.\n", argv[1]);
17
          else {
18
    自
19
              printf("Removing %s was failed.\n", argv[1]);
20
21
22
          return 0;
23
```

수고하셨습니다.

