2017 UNIX 프로그래밍 과제 1

이름 : 임현

분반 : 1분반

학번 : 201511054

1. 개요

파일과 디렉토리에 관련된 시스템 콜과 라이브러리 함수를 이용하여 유닉스 명 령어 ls와 같은 역할을 하는 명령어 myls를 만든다.

2. 수행 예

```
[unix201511054@apple project1]$ vi myls.c
[unix201511054@apple project1]$ gcc -o myls myls.c
[unix201511054@apple project1]$ ./myls
myls    myls.c
[unix201511054@apple project1]$ ./myls -l
-rwxr-xr-x.     1       unix201511054   unixclass1      10079   2017-05-18 02:57        myls
-rw-r--r--.     1       unix201511054   unixclass1      6538    2017-05-18 02:57        myls.c

[unix201511054@apple project1]$ ./myls -a
myls    ..      .       myls.c
[unix201511054@apple project1]$ ./myls /
net     lib64   root    dev     var     selinux sbin    mnt     lost+found      sys     boot  s
rv      bin     home    etc     media   tmp     opt     app     misc    proc    lib     usr
[unix201511054@apple project1]$ ./myls -l -a
-rwxr-xr-x.     1       unix201511054   unixclass1      10079   2017-05-18 02:57        myls
drwxr-x---.     6       unix201511054   unixclass1      4096    2017-05-18 02:57        ..
drwxr-xr-x.     2       unix201511054   unixclass1      4096    2017-05-18 02:57        .
-rw-r--r--.     1       unix201511054   unixclass1      6538    2017-05-18 02:57        myls.c
```

3. 소스 코드

```
/**
* ------------------------------------------------------------
* @title : myls.c
* @author : 임현 (201511054@sangmyung.kr)
* @since : 2017 - 05 - 16
* @brief : 유닉스 명령어 ls와 같은 역할을 하는 명령어 myls
* ------------------------------------------------------------
*/

#include <sys/types.h>
#include <sys/stat.h> // stat 구조체
#include <dirent.h> // dirent 구조체
#include <unistd.h>
#include <stdlib.h>
#include <time.h> // tm 구조체
#include <pwd.h> // passwd 구조체
#include <grp.h> // group 구조체
#include <stdio.h>

/**
* 함수 명 : void Get_Option(int argc, char *argv[], int *opt_check_l, int *opt_check_a);
* 설명 : Opt의 유무를 검사하는 함수
*/
void Get_Option(int argc, char *argv[], int *opt_check_l, int *opt_check_a);

/**
* 함수 명 : void Myls(DIR *dp, int opt_check_l, int opt_check_a, int dir_check, char *argv[],
int dir_name);
* 설명 : Myls의 기능을 수행하는 함수
*/
void Myls(DIR *dp, int opt_check_l, int opt_check_a, int dir_check, char *argv[], int
dir_name);

/**
* 함수 명 : void Argv_Check(int argc, char *argv[], int *dir_check, int *dir_name);
* 설명 : 특정 디렉토리를 명시했는지 검사하는 함수
*/
void Argv_Check(int argc, char *argv[], int *dir_check, int *dir_name);

/**
* 함수 명 : void Mode_itoa(struct stat sbuf);
* 설명 : 접근 권한 값을 integer에서 ASCII로 바꿔주는 함수
*/
void Mode_itoa(struct stat sbuf);

int main(int argc, char *argv[]) { // main 함수의 명령 인자
        int dir_check = 0; // 특정 디렉토리를 명시했는지 확인해주는 변수
        int opt_check_a = 0; // -a 옵션이 있는지 확인해주는 변수
        int opt_check_l = 0; // -l 옵션이 있는지 확인해주는 변수
```

```
        int dir_name = 0; // 특정 디렉토리가 몇 번째 argv에 있는지 알려주는 변수
        DIR *dp; // 디렉토리를 open하고 저장할 변수

                        // 특정 디렉토리를 명시했는지 검사
        Argv_Check(argc, argv, &dir_check, &dir_name);

        // 디렉토리를 여는 부분
        if (dir_check == 0) // 특정 디렉토리가 없는 경우
                dp = opendir("."); // 현재 디렉토리를 염
        else if (dir_check == 1) { // 특정 디렉토기 없는 경우
                if ((dp = opendir(argv[dir_name])) == NULL) { // 특정 디렉토리를 염
                        perror("opendir"); // 없을 경우 오류 메시지 출력
                        exit(1);
                }
        }

        // Opt의 유무를 검사
        Get_Option(argc, argv, &opt_check_l, &opt_check_a);

        // Myls 기능 수행
        Myls(dp, opt_check_l, opt_check_a, dir_check, argv, dir_name);

        // 디렉토리를 닫아줌
        closedir(dp);

        return 0;
}

// Opt의 유무를 검사하는 함수
void Get_Option(int argc, char *argv[], int *opt_check_l, int *opt_check_a) {
        int n;
        while ((n = getopt(argc, argv, "la")) != -1) { // getopt 함수로 인자가 있는지 확인 후
읽음
                switch (n) { // switch 문으로 옵션별 기능 수행
                case 'l':
                        *opt_check_l = 1; // -l 옵션이 있음
                        break;
                case 'a':
                        *opt_check_a = 1; // -a 옵션이 있음
                        break;
                }
        }
}

// Myls 기능 수행
void Myls(DIR *dp, int opt_check_l, int opt_check_a, int dir_check, char *argv[], int dir_name)
{
        struct dirent *dent; // dirent(디렉토리 항목) 구조체 (헤더파일 : <sys/dirent.h>)
        struct stat sbuf; // stat(파일 정보 검색) 구조체 (헤더파일 : <sys/stat.h>)
        char path[BUFSIZ]; // 경로
        struct passwd *pw; // passwd 구조체 (헤더파일 : <pwd.h>)
        struct group *grp; // group 구조체 (헤더파일 : <grp.h>)
```

```
        struct tm *tm; // tm(시간 정보 분해) 구조체 (헤더파일 : <iso/time_iso.h> (<time.h>))
        char tbuf[257]; // 문자열 버퍼

        pw = getpwuid(getuid()); // getuid 함수로 UID를 얻은 후 해당 UID에 관한 자세한 정보를
getwuid 함수로 검색
        grp = getgrgid(getgid()); // getgid 함수로 GID를 얻은 후 해당 GID에 관한 자세한 정보를
getgrgid 함수로 검색

        while ((dent = readdir(dp))) { // 디렉토리를 열고 항목의 정보를 읽음
                if (opt_check_a == 0 && dent->d_name[0] == '.') continue; // a 옵션이 없을
경우 .과 ..을 제외하고 읽음
                else { // a 옵션이 있을 경우
                        if (opt_check_l == 1) { // l 옵션이 있을 경우 파일을 자세히 읽음
                                if (dir_check == 0) sprintf(path, "./%s", dent->d_name); //
특정 디렉토리를 명시하지 않은 경우의 경로 설정
                                else sprintf(path, "%s/%s", argv[dir_name], dent->d_name); //
특정 디렉토리를 명시한 경우의 경로 설정
                                stat(path, &sbuf); // stat의 경로 설정

                                Mode_itoa(sbuf); // 접근 권한 값을 ASCII로 출력하는 함수
                                printf("%o₩t", (unsigned int)sbuf.st_nlink); // 하드링크 번호
                                printf("%s₩t", pw->pw_name); // owner 이름
                                printf("%s₩t", grp->gr_name); // group 이름
                                printf("%d₩t", (int)sbuf.st_size); // 파일 크기
                                tm = localtime(&sbuf.st_ctime); // ctime을 받아옴
                                strftime(tbuf, sizeof(tbuf), "%G-%m-%d %H:%M", tm); // 출력
형식 지정자에 따라 buf에 문자열을 생성하고 출력
                                printf("%s₩t", tbuf); // 시간
                                printf("%s₩n", dent->d_name); // 파일 명
                        }
                        else // l 옵션이 없을 경우
                                printf("%s₩t", dent->d_name); // 파일 명
                }
        }
        printf("₩n");
}

// 특정 디렉토리를 명시했는지 검사
void Argv_Check(int argc, char *argv[], int *dir_check, int *dir_name) {
        int i;
        for (i = 1; i < argc; i++) { // 2번째 매개 변수부터 검사
                if (argv[i][0] != '-') { // -로 시작하지 않는 매개변수 검사
                        *dir_check = 1; // 특정 디렉토리가 있음
                        *dir_name = i; // 특정 디렉토리의 자리 값
                        break;
                }
        }
}

// 접근 권한 값을 Integer에서 ASCII로 바꿔줌
void Mode_itoa(struct stat sbuf) {
        char mode[11] = "----------"; // 기본 선언
```

```
        if (S_ISDIR(sbuf.st_mode)) mode[0] = 'd'; // 디렉토리일 경우
        if (sbuf.st_mode & S_IRUSR) mode[1] = 'r'; // 소유자 읽기 권한
        if (sbuf.st_mode & S_IWUSR) mode[2] = 'w'; // 소유자 쓰기 권한
        if (sbuf.st_mode & S_IXUSR) mode[3] = 'x'; // 소유자 실행 권한
        if (sbuf.st_mode & S_IRGRP) mode[4] = 'r'; // 그룹 읽기 권한
        if (sbuf.st_mode & S_IWGRP) mode[5] = 'w'; // 그룹 쓰기 권한
        if (sbuf.st_mode & S_IXGRP) mode[6] = 'x'; // 그룹 실행 권한
        if (sbuf.st_mode & S_IROTH) mode[7] = 'r'; // 기타 사용자 읽기 권한
        if (sbuf.st_mode & S_IWOTH) mode[8] = 'w'; // 기타 사용자 쓰기 권한
        if (sbuf.st_mode & S_IXOTH) mode[9] = 'x'; // 기타 사용자 실행 권한

        printf("%s.₩t", mode);
}
```

4. 소스 코드 캡처

```
unix201511054@apple:~/project1
/**
 * ------------------------------------------------------------
 * @title : myls.c
 * @author : 임현 (201511054@sangmyung.kr)
 * @since : 2017 - 05 - 16
 * @brief : 유닉스 명령어 ls와 같은 역할을 하는 명령어 myls
 * ------------------------------------------------------------
 */

#include <sys/types.h>
#include <sys/stat.h> // stat 구조체
#include <dirent.h> // dirent 구조체
#include <unistd.h>
#include <stdlib.h>
#include <time.h> // tm 구조체
#include <pwd.h> // passwd 구조체
#include <grp.h> // group 구조체
#include <stdio.h>

/**
 * 함수 명 : void Get_Option(int argc, char *argv[], int *opt_check_l, int *opt_check_a);
 * 설명 : Opt의 유무를 검사하는 함수
 */
void Get_Option(int argc, char *argv[], int *opt_check_l, int *opt_check_a);

/**
 * 함수 명 : void Myls(DIR *dp, int opt_check_l, int opt_check_a, int dir_check, char *argv[], int dir_name);
 * 설명 : Myls의 기능을 수행하는 함수
 */
void Myls(DIR *dp, int opt_check_l, int opt_check_a, int dir_check, char *argv[], int dir_name);

/**
 * 함수 명 : void Argv_Check(int argc, char *argv[], int *dir_check, int *dir_name);
 * 설명 : 특정 디렉토리를 명시했는지 검사하는 함수
 */
void Argv_Check(int argc, char *argv[], int *dir_check, int *dir_name);

/**
 * 함수 명 : void Mode_itoa(struct stat sbuf);
 * 설명 : 접근 권한 값을 integer에서 ASCII로 바꿔주는 함수
 */
void Mode_itoa(struct stat sbuf);

int main(int argc, char *argv[]) { // main 함수의 명령 인자
        int dir_check = 0; // 특정 디렉토리를 명시했는지 확인해주는 변수
        int opt_check_a = 0; // -a 옵션이 있는지 확인해주는 변수
        int opt_check_l = 0; // -l 옵션이 있는지 확인해주는 변수
        int dir_name = 0; // 특정 디렉토리가 몇 번째 argv에 있는지 알려주는 변수
        DIR *dp; // 디렉토리를 open하고 저장할 변수

        // 특정 디렉토리를 명시했는지 검사
        Argv_Check(argc, argv, &dir_check, &dir_name);
"myls.c" 157L, 6538C                                          1,1           꼭대기
unix201511054@apple:~/project1
        // 특정 디렉토리를 명시했는지 검사
        Argv_Check(argc, argv, &dir_check, &dir_name);

        // 디렉토리를 여는 부분
        if (dir_check == 0) // 특정 디렉토리가 없는 경우
                dp = opendir("."); // 현재 디렉토리를 염
        else if (dir_check == 1) { // 특정 디렉토리 없는경우
                if ((dp = opendir(argv[dir_name])) == NULL) { // 특정 디렉토리를 염
                        perror("opendir"); // 없을 경우 오류 메시지 출력
                        exit(1);
                }
        }

        // Opt의 유무를 검사
        Get_Option(argc, argv, &opt_check_l, &opt_check_a);

        // Myls 기능 수행
        Myls(dp, opt_check_l, opt_check_a, dir_check, argv, dir_name);

        // 디렉토리를 닫아줌
        closedir(dp);

        return 0;
}

// Opt의 유무를 검사하는 함수
void Get_Option(int argc, char *argv[], int *opt_check_l, int *opt_check_a) {
        int n;
        while ((n = getopt(argc, argv, "la")) != -1) { // getopt 함수로 인자가 있는지 확인 후 읽음
                switch(n) { // switch 문으로 옵션별 기능 수행
                        case 'l':
                                *opt_check_l = 1; // -l 옵션이 있음
                                break;
                        case 'a':
                                *opt_check_a = 1; // -a 옵션이 있음
                                break;
                }
        }
}

// Myls 기능 수행
void Myls(DIR *dp, int opt_check_l, int opt_check_a, int dir_check, char *argv[], int dir_name) {
        struct dirent *dent; // dirent(디렉토리 항목) 구조체 (헤더파일 : <sys/dirent.h>)
        struct stat sbuf; // stat(파일 정보 검색) 구조체 (헤더파일 : <sys/stat.h>)
        char path[BUFSIZ]; // 경로
        struct passwd *pw; // passwd 구조체 (헤더파일 : <pwd.h>)
        struct group *grp; // group 구조체 (헤더파일 : <grp.h>)
        struct tm *tm; // tm(시간 정보 분해) 구조체 (헤더파일 : <iso/time_iso.h> (<time.h>))
        char tbuf[257]; // 문자열 버퍼

        pw = getpwuid(getuid()); // getuid 함수로 UID를 얻은 후 해당 UID에 관한 자세한 정보를 getwuid 함수로 검색
        grp = getgrgid(getgid()); // getgid 함수로 GID를 얻은 후 해당 GID에 관한 자세한 정보를 getgrgid 함수로 검색
                                                              51,2-9        47%
```


unix201511054@apple:~/project1
101 pw = getpwuid(getuid()); // getuid 함수로 UID를 얻은 후 해당 UID에 관한 자세한 정보를 getwuid 함수로 검색
102 grp = getgrgid(getgid()); // getgid 함수로 GID를 얻은 후 해당 GID에 관한 자세한 정보를 getgrgid 함수로 검색
103
104 while ((dent = readdir(dp))) { // 디렉토리를 열고 항목의 정보를 읽음
105 if (opt_check_a == 0 && dent->d_name[0] == '.') continue; // a 옵션이 없을 경우 .과 ..을 제외하고 읽음
106 else { // a 옵션이 있을 경우
107 if (opt_check_l == 1) { // l 옵션이 있을 경우 파일을 자세히 읽음
108 if (dir_check == 0) sprintf(path, "./%s", dent->d_name); // 특정 디렉토리를 명시하지 않은 경우의 경로 설정
109 else sprintf(path, "%s/%s", argv[dir_name],dent->d_name); // 특정 디렉토리를 명시한 경우의 경로 설정
110 stat(path, &sbuf); // stat의 경로 설정
111
112 Mode_itoa(sbuf); // 접근 권한 값을 ASCII로 출력하는 함수
113 printf("%o\t", (unsigned int)sbuf.st_nlink); // 하드링크 번호
114 printf("%s\t", pw->pw_name); // owner 이름
115 printf("%s\t", grp->gr_name); // group 이름
116 printf("%d\t", (int)sbuf.st_size); // 파일 크기
117 tm = localtime(&sbuf.st_ctime); // ctime을 받아옴
118 strftime(tbuf, sizeof(tbuf), "%G-%m-%d %H:%M", tm); // 출력 형식 지정자에 따라 buf에 문자열을 생성하고 출력
119 printf("%s\t", tbuf); // 시간
120 printf("%s\n", dent->d_name); // 파일 명
121 }
122 else // l 옵션이 없을 경우
123 printf("%s\t", dent->d_name); // 파일 명
124 }
125 }
126 printf("\n");
127 }
128
129 // 특정 디렉토리를 명시했는지 검사
130 void Argv_Check(int argc, char *argv[], int *dir_check, int *dir_name) {
131 int i;
132 for (i = 1; i < argc; i++) { // 2번째 매개 변수부터 검사
133 if (argv[i][0] != '-') { // -로 시작하지 않는 매개변수 검사
134 *dir_check = 1; // 특정 디렉토리가 있음
135 *dir_name = i; // 특정 디렉토리의 자리 값
136 break;
137 }
138 }
139 }
140
141 // 접근 권한 값을 Integer에서 ASCII로 바꿔줌
142 void Mode_itoa(struct stat sbuf) {
143 char mode[11] = "----------"; // 기본 선언
144
145 if (S_ISDIR(sbuf.st_mode)) mode[0] = 'd'; // 디렉토리일 경우
146 if (sbuf.st_mode & S_IRUSR) mode[1] = 'r'; // 소유자 읽기 권한
147 if (sbuf.st_mode & S_IWUSR) mode[2] = 'w'; // 소유자 쓰기 권한
148 if (sbuf.st_mode & S_IXUSR) mode[3] = 'x'; // 소유자 실행 권한
149 if (sbuf.st_mode & S_IRGRP) mode[4] = 'r'; // 그룹 읽기 권한
150 if (sbuf.st_mode & S_IWGRP) mode[5] = 'w'; // 그룹 쓰기 권한
151 if (sbuf.st_mode & S_IXGRP) mode[6] = 'x'; // 그룹 실행 권한
152 if (sbuf.st_mode & S_IROTH) mode[7] = 'r'; // 기타 사용자 읽기 권한
101,2-9 95%
unix201511054@apple:~/project1
151 if (sbuf.st_mode & S_IXGRP) mode[6] = 'x'; // 그룹 실행 권한
152 if (sbuf.st_mode & S_IROTH) mode[7] = 'r'; // 기타 사용자 읽기 권한
153 if (sbuf.st_mode & S_IWOTH) mode[8] = 'w'; // 기타 사용자 쓰기 권한
154 if (sbuf.st_mode & S_IXOTH) mode[9] = 'x'; // 기타 사용자 실행 권한
155
156 printf("%s.\t", mode);
157 }
151,2-9 바닥

5. 파일 이름

경로 : ~/project1

소스 파일 : myls.c

수행 파일 : myls

6. 제출일

2017년 5월 18일 수업시간

7. 제출물

소스파일 프린트, 수행 예 캡처