6장 과제

2018312567 조명하

실습문제 1.

1.1 소스 코드

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>

/* 파일 전체를 잠금하는 프로그램 */
int main(int argc, char *argv[]) {
    int fd, ret, c;
    static struct flock lock;
    if (argc != 2) {
        fprintf(stderr, "사용법 : file_lock file\n"); // 명령어 입력 오류
        exit(1);
    }

    if ((fd = open(argv[1], O_RDWR)) == -1) {  // 파일 open 오류
        perror(argv[1]);
        exit(2);
    }
    /*잠금 설정*/
    lock.l_type = F_WRLCK;    // 쓰기 잠금
    lock.l_start = 0;
    lock.l_whence = SEEK_SET;    // 처음부터 시작
    lock.l_len = 0;    // 전체 잠금
    lock.l_pid = getpid();

    ret = fcntl(fd, F_SETLK, &lock);    // 여기서부터 파일 잠금
    printf("%s is locked.\n", argv[1]);

    if (ret == 0){
        system("gedit test.txt\n");  // 파일 잠금한 상태에서 test.txt 오픈
    }
    else {
        perror(argv[1]);  // 파일 잠금 오류
        exit(3);
    }

    close(fd); // 파일 close
    return 0;
}
```

명령어를 입력받아서 정상적으로 입력하지 않으면 사용법을 출력하고 프로그램을 종료한다.

명령어를 입력받아서 파일을 읽고 쓸 수 있게 열고 열 수 없으면 에러메시지를 출력하고 프로그램을 종료한다.

Lock의 내용을 쓰기 잠금, 파일의 처음부터, 끝까지 잠금한다고 설정하고 fcntl함수로 잠금을 설정하고 파일이 잠금되었다고 출력한다. 성공적으로 잠금되어 ret값이 0이면 system()으로 잠금한 파일을 연다. ret값이 0이 아니면 파일 잠금에 오류가 있는 것이므로 에러 메시지를 출력하고 프로그램을 종료한다.

파일을 닫고 프로그램을 종료한다.

1.2 프로그램 실행 결과

```
[chomyungha@localhost 06]$ gcc -o file_lock file_lock.c
[chomyungha@localhost 06]$ ./file_lock test.txt &  gedit test.txt
[1] 3388
test.txt is locked.
[1]+  Done                    ./file_lock test.txt
[chomyungha@localhost 06]$
```

소스 코드를 컴파일하고 file_lock 으로 test.txt를 잠금하고 실행시키자 그 다음의 gedit test.txt는 실행되지 않는다.

실습문제 2.

1.1 소스 코드

```c
#include <stdio.h>
#include <fcntl.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <string.h>


/* fcntl()로 lockf() 구현하는 프로그램 */
int main(int argc, char *argv[]) {
    int fd, len, ret;
    struct flock lock;    // flock 구조체

    if (argc != 4) {
        fprintf(stderr, "사용법 : lockF file cmd size\n");   // 명령어 입력 오류
        exit(1);
    }

    if ((fd = open(argv[1], O_RDWR)) == -1) {  // 파일 open 오류
        perror(argv[1]);
        exit(2);
    }

    len = atoi(argv[3]); // 길이 int로 캐스팅
```

명령어가 제대로 입력되지 않은 경우 명령어 사용 방법을 출력하고 프로그램을 종료한다.

파일을 읽기, 쓰기 권한으로 열고 제대로 열리지 않으면 에러 메시지를 출력하고 프로그램을 종료한다.

세 번째로 입력받은 것을 정수형으로 변환해서 len 에 저장한다.

```
    /*잠금 설정 : 쓰기 잠금만 가능*/
    if (strcmp(argv[2], "F_LOCK") == 0 ){
        lock.l_type = F_WRLCK;   // 쓰기 잠금
        lock.l_whence = SEEK_CUR;
        lock.l_len = len;
        ret = fcntl(fd, F_SETLKW,&lock);    // 잠금 성공할 때까지 기다리기
    }
    else if (strcmp(argv[2],"F_TLOCK" ) == 0 ) {
        lock.l_type = F_WRLCK;   //쓰기 잠금
        lock.l_whence = SEEK_CUR;
        lock.l_len = len;
        ret = fcntl(fd, F_SETLK,&lock); // 결과 바로 반환
    }
    else if (strcmp(argv[2],"F_TEST" ) == 0 ) {
        fcntl(fd, F_GETLK, &lock);   // 파일 상태 확인
        if (lock.l_type == F_UNLCK)  // 잠금 안 됐으면 0
            ret = 0;
        else                         // 잠금됐으면 -1
            ret = -1;
    }
    else if (strcmp(argv[2],"F_ULOCK" ) == 0 ) {
        lock.l_type = F_UNLCK;   // 잠금 해제하기
        lock.l_whence = SEEK_CUR;
        lock.l_len = len;
        ret = fcntl(fd, F_SETLK,&lock);
    }
    else {
        printf("NOT SUPPORT THIS CMD\n");   // 지원하지 않는 cmd
        exit(4);
    }

    printf("%d\n", ret); // 반환값 출력하기
    close(fd);

    return 0;
}
```

입력받은 명령어에 따라 lock 구조체의 변수 설정을 다르게 해 준다.

1) cmd 가 F_LOCK 인 경우. 쓰기 잠금으로 현재 위치부터 len 만큼 잠금 설정을 하고 성공할 때까지 기다린다

2) cmd 가 F_TLOCK 인 경우, 쓰기 잠금으로 현재 위치부터 len 만큼 잠금 설정을 하고 성공하든 실패하든 바로 반환값을 return 한다.

3) cmd 가 F_TEST 인 경우, 파일의 상태를 확인하고 잠금이 안 됐으면 0, 잠금되어있으면 -1 을 return 한다.

4) cmd 가 F_ULOCK 인 경우, 현재 위치부터 len 만큼 파일의 잠금을 해제한다.

5) cmd 가 지원하지 않는 형식인 경우, 메시지를 출력하고 종료한다.

반환값을 출력하고 파일을 닫고 종료한다.

1.2 실행 결과

```
[chomyungha@localhost 06]$ gcc -o lockF lockf.c
[chomyungha@localhost 06]$ ./lockF test2.txt F_LOCK 0
0
[chomyungha@localhost 06]$ ./lockF test2.txt F_TLOCK 0
0
[chomyungha@localhost 06]$ ./lockF test2.txt F_TEST 0
0
[chomyungha@localhost 06]$ ./lockF test2.txt F_ULOCK 0
0
[chomyungha@localhost 06]$ ./lockF test2.txt F_UnlocK 0
NOT SUPPORT THIS CMD
[chomyungha@localhost 06]$
```

파일 잠금과 실행, 잠금 해재, 명령어가 이상할 경우 모두 정상적으로 실행된다. 한 파일 내에서 파일을 열고닫는 것이 아니기 때문에 모두 성공한 것이다. 이 lockF 프로그램의 메인함수를 다른 프로그램에서 호출해서 사용한다면 정상적인 결과가 나올 것이다.