리눅스 프로그래밍 프로젝트 보고서

**\* 프로젝트 설명**

리눅스 프로그래밍 수업 시간에 배운 서버 – 클라이언트 모델로 동작하는 사물함 관리 시스템 구현

**\* 역할 분담**

202246117 임찬우 – 보고서 작성, 코드 구현

202012216 엄주영 – 프로그램 데모 녹화, readme.txt 작성, 코드 구현

**\* 코드 전체 흐름**

- 실행 과정

1. 서버가 실행 된 후 락커 구조체, 서버 소켓이 생성됨
2. 사용자에게 사물함 개수를 입력받고, 사물함을 초기화
3. 클라이언트 실행 후 클라이언트 소켓이 생성되고 도메인 소켓에 연결하여 서버와 통신
4. 서버가 연결을 수락하고, 클라이언트는 메뉴에서 원하는 작업을 골라서 수행

- 클라이언트

1. **소켓 생성 및 서버 연결**:

* 클라이언트 소켓을 생성하고 manager 유닉스 도메인 소켓에 연결하여 서버와 통신합니다.

2. **사물함 개수 읽기**:

* index라는 FIFO 파일을 통해 서버로부터 사물함 개수를 읽어옵니다.

3. **사물함 상태 표시**:

* 서버로부터 사물함 상태를 읽어와 화면에 표시합니다.

4. **메뉴 표시 및 사용자 선택**:

* 사용자에게 사물함 사용, 내용물 삭제, 관리자와의 대화 중 하나를 선택하도록 메뉴를 표시합니다.

5. **사물함 사용**:

* 사용자가 사물함을 사용하도록 도와줍니다. 사물함 비밀번호를 설정하고, 사물함에 내용을 넣을 수 있습니다.

6. **사물함 내용물 삭제**:

* 사용자가 사물함에서 특정 내용물을 삭제할 수 있도록 합니다.

7. **관리자와의 대화**:

* 사용자가 관리자와 실시간 채팅을 할 수 있도록 합니다. 채팅을 종료하려면 'Q'를 입력합니다.

8. **프로그램 종료**:

* 사용자가 계속 작업을 진행할지 여부를 물어보고, 'N'을 입력하면 프로그램이 종료됩니다.

- 서버

1. **소켓 생성 및 서버 설정**:

* 서버 소켓을 생성하고 manager 유닉스 도메인 소켓에 바인드하여 클라이언트의 연결을 대기합니다.

2. **사물함 개수 입력 및 초기화**:

* 사용자에게 사물함 개수를 입력받고, 사물함을 초기화합니다. 사물함 상태는 비어 있으며 비밀번호는 '0000'으로 설정됩니다.

3. **클라이언트 연결 수락 및 처리**:

* 클라이언트의 연결을 수락하고, 클라이언트를 처리하는 자식 프로세스를 생성합니다.

4. **사물함 상태 표시**:

* 현재 사물함의 상태를 클라이언트에 전송합니다.

5. **클라이언트 요청 처리**:

* 클라이언트의 요청을 읽어와 채팅 모드, 사물함 사용 모드, 내용물 삭제 모드를 처리합니다.

6. **사물함 사용 모드**:

* 사물함이 비어 있으면 비밀번호를 설정하고 내용물을 추가할 수 있습니다. 비어 있지 않으면 비밀번호를 확인하여 사물함을 비우고 내용물을 삭제합니다.

7. **내용물 삭제 모드**:

* 사물함의 비밀번호를 확인한 후 특정 내용물을 삭제할 수 있습니다.

8. **업데이트된 사물함 상태 저장**:

* 클라이언트 요청 처리 후 업데이트된 사물함 상태를 파이프를 통해 부모 프로세스에 전달합니다.

**\* 추가기능**

1. 서버와 클라이언트가 서로 채팅으로 소통할 수 있는 기능
2. 사물함에 원하는 아이템을 추가하고 삭제할 수 있는 기능
3. 비밀번호를 설정 시 두 번을 입력하여 제대로 입력되었는지 확인해 줄 수 있는 기능
4. 동적할당(malloc)을 통해서 사물함의 크기를 원하는 만큼 할당해주는 기능

**\* 우수성을 나타내는 기능**

1. 서버와 클라이언트가 채팅을 소통하는 기능

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**- 클라이언트**

1. 변수 선언

 send\_message: 클라이언트가 보낼 메시지를 저장하는 배열.

 receive\_message: 서버(관리자)가 보낸 메시지를 저장하는 배열.

 mode: 채팅 모드를 나타내며 'c'로 설정.

2. 채팅 모드 시작 신호

 write(client\_socket, &mode, 1);: 클라이언트 소켓을 통해 'c' 모드 신호를 서버에 전송.

 printf: 채팅 시작 메시지를 출력.

3. 채팅 루프 시작

 printf("Client : ");: 클라이언트가 입력할 메시지 프롬프트를 출력.

 scanf(" %[^\n]s", send\_message);: 클라이언트의 입력을 받아 send\_message에 저장. 이 형식 지정자 %[^\n]는 엔터 키를 누르기 전까지의 모든 문자열을 입력받음.

 write(client\_socket, send\_message, sizeof(send\_message));: 클라이언트가 입력한 메시지를 서버에 전송.

4. 채팅 종료 조건

strcmp 함수로 send\_message가 "Q"인지 확인하여, "Q"를 입력하면 루프를 탈출하고 채팅을 종료.

5. 서버 메시지 수신 및 출력

 read(client\_socket, receive\_message, sizeof(receive\_message));: 서버로부터 메시지를 읽어 receive\_message에 저장.

 printf("Manager : %s\n", receive\_message);: 수신한 메시지를 출력.

**- 서버**

**1. 채팅 모드 시작 확인**

if (mode == 'c') : mode가 'c'인지 확인하여 채팅 모드를 시작함

**2. 클라이언트로부터 메시지 읽기**

read(client\_socket, receive\_message, sizeof(receive\_message)); : 클라이언트로부터 메시지를 읽어옴

**3. 채팅 종료 조건 확인**

if (!strcmp(receive\_message, "Q")) break; : 클라이언트로부터 받은 메시지가 "Q"인지 확인하여, "Q"일 경우 루프를 종료함

**4. 메시지 전송**

write(client\_socket, send\_message, sizeof(send\_message)); : 사용자가 입력한 메시지를 클라이언트로 전송합니다

**5. 종료**

while (1) : 위의 과정이 무한 루프로 반복되며, 클라이언트와의 채팅이 유지됨 클라이언트가 "Q"를 보낼 때까지 루프는 종료되지 않음

1. 사물함에 원하는 아이템을 추가하고 삭제하는 기능

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**- 클라이언트**

**1. 변수 선언**

char user\_id[100], password[10], item\_to\_delete[300]; char mode = 'd'; char response[100];

**2. 삭제 모드 설정 및 사물함 번호 전송**

write(client\_socket, &mode, 1);: 삭제 모드를 서버에 전송

sleep(1);: 1초 대기(연결 안정성을 위해 추가된 것일 가능성 있음)

write(client\_socket, user\_id, sizeof(user\_id));: user\_id를 서버에 전송

**3. 비밀번호 전송**

write(client\_socket, password, sizeof(password)); : 입력된 내용을 서버에 전송

**4. 서버의 응답 출력**

read(client\_socket, response, sizeof(response)); printf("%s\n", response); : 서버로부터 응답을 읽어 response에 저장하고 이를 출력

**- 서버**

**1. 삭제 모드 시작 확인**

else if (mode == 'd') : mode가 'd'인 경우 내용물 삭제 모드를 시작함

**2. 사용자 ID 읽기**

read(client\_socket, user\_id, 100);

locker\_id = atoi(user\_id);

클라이언트로부터 사용자 ID를 읽어옵니다. 이 ID는 삭제할 내용물이 있는 사물함을 식별합니다.

**3. 사물함이 비어있음을 확인하고 비밀번호 확인**

if (!lockers[locker\_id].empty) : 사물함이 비어있는지 확인

strcpy(send\_password, lockers[locker\_id].pwd); : 사물함의 비밀번호를 send\_password에 복사

read(client\_socket, received\_password, sizeof(received\_password)); : 클라이언트로부터 비밀번호를 읽어옴

if (!strcmp(received\_password, lockers[locker\_id].pwd)) : 클라이언트로부터 받은 비밀번호와 사물함 비밀번호가 일치하는지 확인

**4. 삭제할 내용물 읽기**

read(client\_socket, item\_to\_delete, sizeof(item\_to\_delete)); : 클라이언트로부터 삭제할 내용물을 읽어옴

**5. 내용물 검색 및 삭제**

for (int i = 0; i < lockers[locker\_id].content\_count; i++) {

if (strcmp(lockers[locker\_id].content[i], item\_to\_delete) == 0) {

found = 1;

**사물함의 내용물에서 삭제할 항목을 찾음**

for (int j = i; j < lockers[locker\_id].content\_count - 1; j++) {

strcpy(lockers[locker\_id].content[j], lockers[locker\_id].content[j + 1]);

}

lockers[locker\_id].content[lockers[locker\_id].content\_count - 1][0] = '\0'; // 마지막 항목 초기화

lockers[locker\_id].content\_count--;

write(client\_socket, "내용물 삭제 완료", sizeof("내용물 삭제 완료"));

break;

}

}

**항목을 찾으면, 그 항목을 삭제하고 배열을 재정렬**

**6. 삭제할 내용물이 없는 경우**

if (!found) {

write(client\_socket, "해당 내용물이 없습니다", sizeof("해당 내용물이 없습니다"));

}

**만약 삭제할 항목을 찾지 못한 경우, 클라이언트에 해당 내용물이 없음을 알림**

1. 동적할당을 통해서 사물함의 크기를 원하는 만큼 지정하는 기능

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- 서버**

**1. 사물함 개수 입력**

printf("사물함 개수를 입력하세요 : ");

scanf("%s", str);

num\_lockers = atoi(str);

사용자로부터 사물함의 개수를 입력받아 num\_lockers 변수에 저장함

**2. 동적 메모리 할당**

lockers = (struct locker\*)malloc((num\_lockers + 1) \* sizeof(struct locker));

입력받은 사물함 개수를 기반으로 num\_lockers + 1 크기의 struct locker 배열을 동적으로 할당함

이 배열은 각 사물함의 정보를 저장하는 데 사용됨

**3. 사물함 초기화**

initializeLockers(lockers, num\_lockers);

초기화 함수를 호출하여 할당된 사물함 배열을 초기 상태로 설정함