**Unix System Programing 2020**

**Quiz Show 설계 보고서**

**IT정보공학과**

**201614929 곽성준**

**201819144 권민정**

1. **역할**

서버(Server): 곽성준

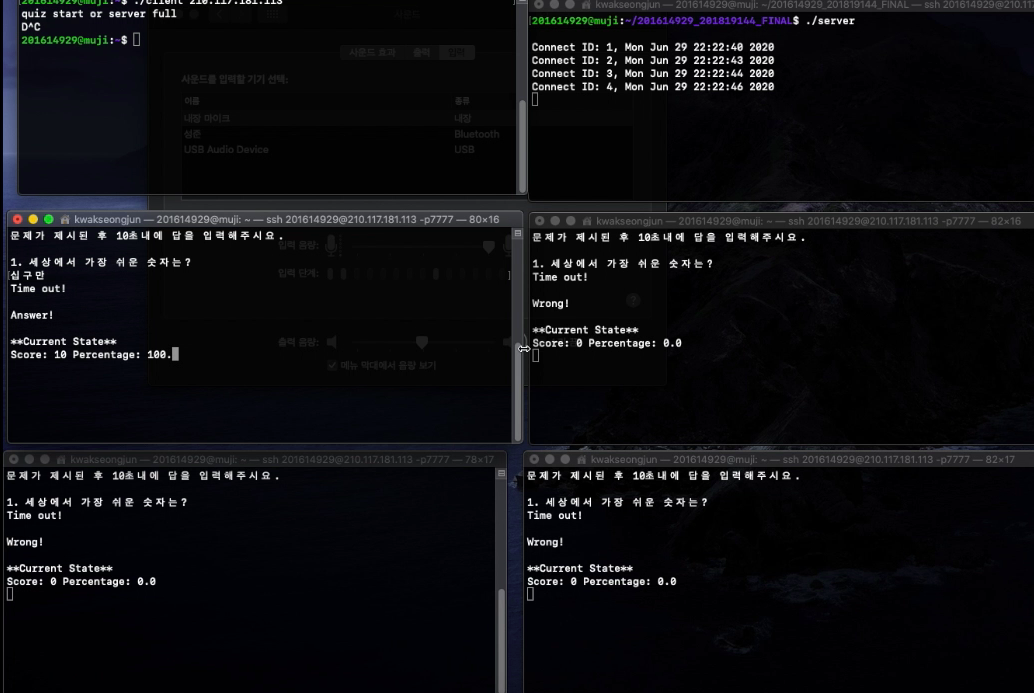
클라이언트(Client): 권민정

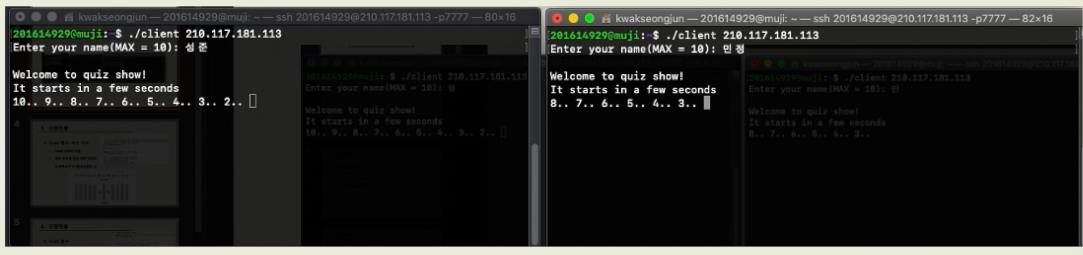
1. **프로그램 소개**

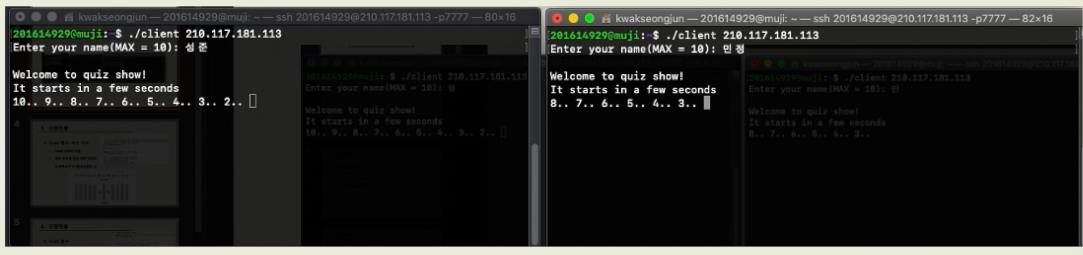
이 프로그램의 대략적인 내용은 ‘도전! 골든벨’이라는 퀴즈 프로그램과 비슷하다. 여러 명이 동시에 퀴즈를 풀고 시간 제한 내에 빠르게 문제를 풀어야 하기 때문에 스피드 퀴즈처럼 작동한다. 우선 한 명의 클라이언트가 서버에 접속하면서 퀴즈쇼가 시작된다. 크게 닉네임 설정, 카운트 다운, 퀴즈쇼 진행, 결과로 나누어 설명하고자 한다.

1. 닉네임 설정

클라이언트가 서버에 접속하면 클라이언트는 자신의 닉네임을 설정할 수 있으며, 닉네임을 설정한 후에 서버에서 클라이언트의 닉네임과 접속 시간을 받아 관리한다. 동시에 최대 4명의 인원이 함께 퀴즈를 풀게 되며, 동시 접속자가 없을 때는 클라이언트 하나가 단독으로도 퀴즈쇼에 참여가 가능하다.

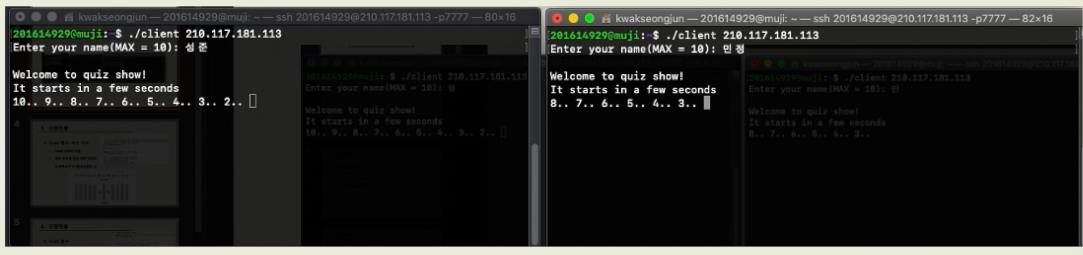
 (server 스크린샷)

(client 스크린샷)

(client 스크린샷)

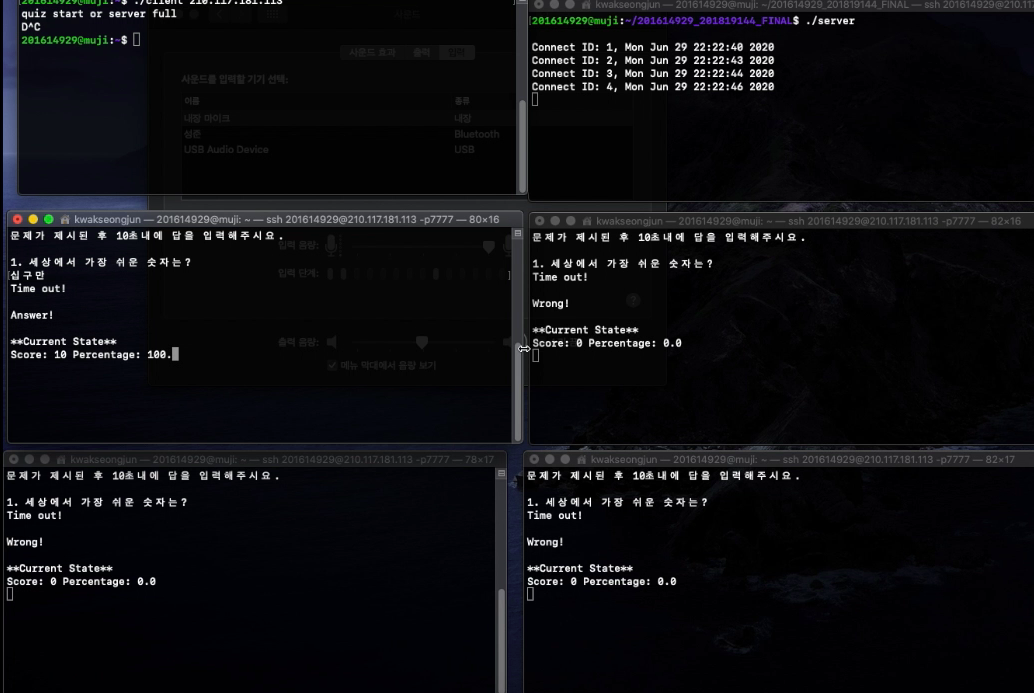
1. 카운트 다운

서버에 접속한 클라이언트 중 가장 먼저 닉네임을 입력한 클라이언트가 퀴즈쇼의 대표 역할을 하게 되는데, 최초로 닉네임을 입력한 대표 클라이언트가 닉네임을 입력하고 나면 10초동안 다른 클라이언트들이 접속하기를 기다리게 된다. 이때 주어진 10초동안 아무도 접속하지 않으면 혼자 퀴즈쇼를 진행하게 되고, 다른 클라이언트들(최대 3명)이 접속하게 되면 새로 접속한 클라이언트들은 대표 클라이언트의 카운트가 0이 될 때까지 함께 대기하였다가 동시에 퀴즈쇼를 시작한다.

(client 스크린샷)

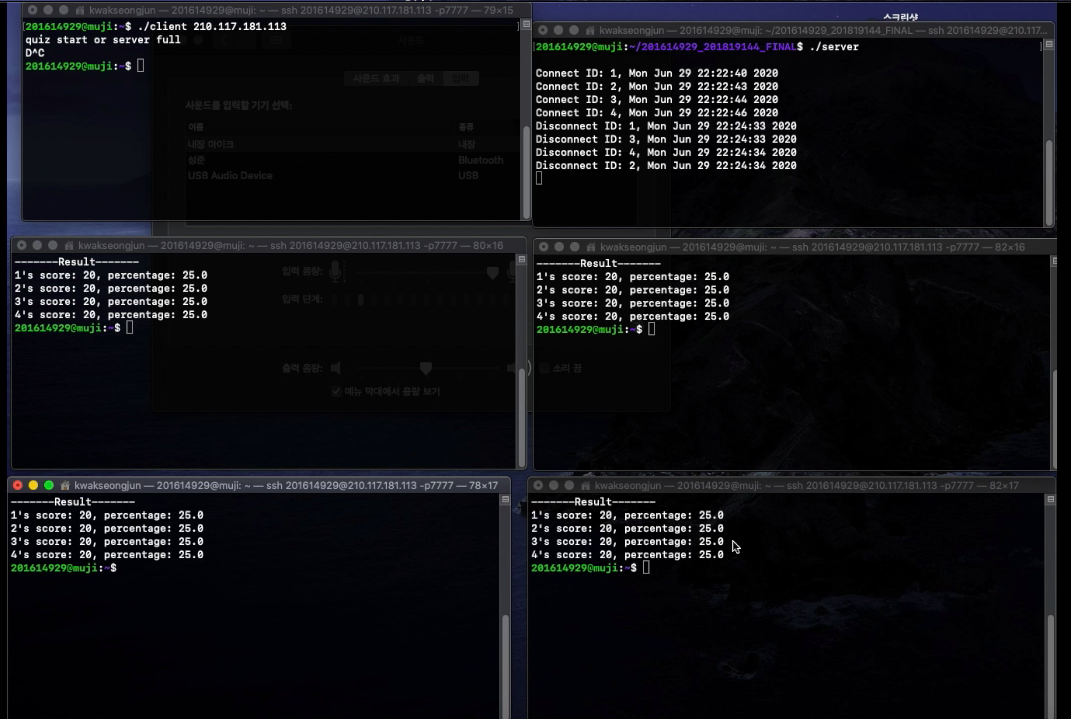
1. 퀴즈쇼 진행

카운트 다운이 끝난 후 퀴즈쇼가 시작되면 새로운 클라이언트가 서버에 접속할 수 없으며, 각 문제를 풀기 위해 각각의 클라이언트들에게는 10초의 시간이 주어진다. 서버가 퀴즈 파일에서 문제와 정답을 읽어 문제는 클라이언트로 보낸다. 문제를 수신한 퀴즈쇼의 참가자들은 각 문제당 10초 내에 답을 작성하여야 하며, 10초가 지나 Time out상태에 도달하면 사용자가 입력한 답이 정답인지 오답인지 채점이 된다. 퀴즈 참여자가 입력한 답이 정답일 경우 “Answer!”이라는 메시지가 나타나며 이 참여자는 10점을 얻을 수 있다. 반대로 오답일 경우 “Wrong!”이라는 메시지가 나타나며 점수를 받을 수 없다. 매 문제가 출제될 때마다, 문제에 대한 채점이 완료되고 나면 각 퀴즈쇼 참여자는 자신의 점수와 정답률을 확인할 수 있다.

(client 스크린샷)

1. 결과

모든 문제의 출제가 완료되고 난 후에 결과창이 나타난다. 결과창에서는 모든 사용자들의 총점과 정답률을 확인할 수 있다. 여러 명의 클라이언트들과 함께 퀴즈쇼가 진행되는 도중에 어떤 하나의 클라이언트가 서버와의 연결을 끊는 일이 발생한다면, 결과창에는 이 클라이언트의 점수와 정답률에 관한 정보는 나타나지 않는다.

(client 스크린샷)

1. **추가한 부가기능 소개**

이 프로그램은 제시된 기본기능 외에 퀴즈쇼 참가자들이 동시에 퀴즈를 풀 수 있도록 하는 기능이 있다는 것이 가장 큰 특징이다. 여러 명의 사용자가 아무 때나 서버에 접속하고 개인적으로 퀴즈쇼에 참가하여 문제를 푸는 것은 참가자들의 소켓에 관한 디스크립터 번호를 관리함으로써 쉽게 구현이 가능했다. 조금 더 특별하게 퀴즈쇼를 구현할 수 있는 방법을 찾다가 사용자들이 동시에 문제를 풀도록 하는 것은 어떨까 하는 아이디어가 떠올랐다. 서버에 접속한 클라이언트의 닉네임을 얻는 기능을 기본기능에서 추가로 구현하였고, 최초로 닉네임을 입력한 클라이언트의 시간에 맞추어 다른 클라이언트들이 동기화된다. 이것을 구현한 이유는 우리가 게임을 할 때, 게임방을 만들어 그 곳에서 함께 게임을 할 유저들을 기다리다가 적정 인원이 모이면 다 함께 게임을 시작하는 것과 비슷한 느낌을 주고 싶었기 때문이다. 이 10초의 대기시간 동안 클라이언트들은 자유롭게 서버에 접속하거나 접속을 끊을 수 있는데, 이는 게임방에 접속한 유저들이 게임이 시작되기 전까지 자유롭게 방에 들어오고 나가고 하는 것을 표현한 것이다. 게임방의 게임이 시작되면, 그 방에는 더 이상 다른 유저들이 입장할 수 없듯이 이 프로그램도 대기 시간이 끝난 후 퀴즈쇼가 시작되면 더 이상 다른 클라이언트들이 접속할 수 없다. 만약 클라이언트가 서버와의 접속을 끊으면 서버는 어떤 클라이언트가 종료되었는지 클라이언트의 닉네임과 함께 메시지를 출력한다. 앞에서도 한번 언급한 바와 같이, 중간에 퀴즈쇼를 나간 클라이언트들의 결과값은 마지막 결과창에서 제외되기 때문에 확인할 수 없도록 하였다.

그 다음 부가기능은 모든 퀴즈 참가자들에게는 동시에 퀴즈가 주어지며 문제를 풀기 위해 짧은 10초의 시간이 동일하게 주어지는 것이다. 모든 문제의 제한시간은 10초이며 주어진 시간 내에 문제를 풀지 못하거나 정답을 입력하지 않았을 시 오답으로 처리되도록 하였다. 보기 좋게 구현하기 위해 count down을 하는 동안 몇 초가 남았는지는 표시하지 않았다. 퀴즈 참가자들은 짧은 시간동안 동시에 주어지는 문제를 풀고 정답을 입력했을 시 10점을 획득하게 된다.

1. **서버 동작**

서버는 우선 프로세스간 공유 데이터인 sock\_args구조체를 통해 각 클라이언트들의 정보를 관리할 수 있다. 이 구조체는 클라이언트들의 파일 디스크립터, 프로세스 id, 접속 시간, 게임 참여자의 수와 닉네임, 점수와 성공률 등을 저장할 수 있게 한다. 서버 코드는 main함수 내에서 크게 9개의 함수를 통해 동작한다.

1. extern int make\_server\_socket() :

정해진 포트넘버 10223에 관한 소켓을 만들어주는 함수이다.

1. void client\_request(int fd) :

client의 요청을 실행해 주는 큰 틀을 가진 함수로, 이름 설정, 카운트 다운 등에 관한 함수들이 이 함수가 실행되면서 호출된다.

1. void add\_client(int fd):

클라이언트가 서버에 접속했을 때 해당 클라이언트의 정보를 초기화 해주기 위한 함수이다. sock\_args구조체의 객체 shared\_data의 값을 해당 클라이언트의 인덱스에 따라 sock배열에 파일 디스크립터를 할당해주고 점수와 정답률을 0으로 초기화, 마지막으로 유저 수를 증가시킨다.

1. void count\_down(int fd, int index) :

닉네임 입력 후 10초 동안의 대기시간을 구현하는 함수이다. 닉네임을 처음으로 입력한 대표 클라이언트가 shared\_data의 start\_index값을 자신을 나타내는 인덱스 값으로 설정한 후 0으로 설정되어 있던 count \_start 값을 1로 바꾸면서 카운트 다운을 시작할 준비를 한다. 대표 클라이언트에 관한 shared\_data의 s\_time 값을 1씩 감소시키고 이 값을 나머지 모든 클라이언트 들에 보내어 클라이언트들의 시간을 동기화한다.

1. void receive\_name(int fd) :

서버에 접속한 클라이언트에게 닉네임 입력을 요청하고 닉네임을 받은 후, 매개변수를 통해 전달받은 파일 디스크립터 번호에 해당하는 인덱스를 찾아 인덱스에 맞게 shared\_data의 user\_name배열에 저장한다.

1. int search\_index(int fd) :

파일 디스크립터 번호를 인자로 받아 해당 값이 저장되어 있는 sock배열의 인덱스를 반환한다.

1. void child\_wait(int signum) :

클라이언트가 서버와의 접속을 끊었을 때 호출되는 함수이다. 이 프로그램은 fork()를 호출하여 생성된 자식 프로세스들을 통해 동작하는데, 자식 프로세스가 종료되면 signal(SIGCHLD, child\_wait);를 통해 시그널을 전달하고 child\_wait()함수가 호출된다. Child\_wait()함수는 서버와의 연결을 끊은 클라이언트가 있을 때 이에 관한 메시지를 출력하고 remove\_client(shared\_data->sock[i]);를 호출하여 연결을 끊은 후의 sock 배열들을 정리한다. 이 때 shared data->usernum이 0이라면 init\_shared\_data()함수를 통하여 shared\_data의 값들을 서버와 연결된 클라이언트가 하나도 없을 때의 값으로 설정해준다.

1. void init\_shared\_data() :

클라이언트들이 접근하는 공유데이터, 즉 shared\_data의 값을 서버에 접속한 클라이언트가 0명인 상태일 때의 값으로 설정해주는 함수이다.

1. void remove\_client(int fd):

클라이언트가 서버와의 연결을 종료했을 때, 해당 클라이언트가 가지고 있던 파일 디스크립터가 닫히므로 shared\_data의 sock배열의 각 인덱스에 해당하는 값을 정리해주는 역할을 한다.

1. **클라이언트 동작**

클라이언트 코드는 main함수 내에서 크게 6개의 함수를 통해 동작한다.

1. connect\_to\_server(char \*host, int portnum) :

미리 정해 놓은 포트넘버 10223으로 서버와 접속하게 해주는 함수이다.

1. enter\_name(int fd) :

서버가 닉네임을 입력하라는 메시지를 보내면 그 메시지를 우선 읽는다. scanf를 통해 name 배열에 닉네임을 저장하여 서버로 보낸다.

1. receive\_countDown(int fd) :

닉네임을 입력하고 나면 서버로부터 count down을 위한 수를 매 초마다 전달받아 출력한다. 서버로부터 클라이언트가 0까지 수신하면 게임 시작을 위한 메시지를 받아 출력한 후 system(“clear”)을 호출하여 화면을 깨끗하게 정리한다.

1. quiz\_ing(int fd) :

서버로부터 퀴즈를 수신하고 클라이언트의 답을 송신하는 함수이다. 서버로부터 읽어온 퀴즈를 출력하고 스레드를 생성하여 enter\_answer함수를 실행한다. 이를 통해 카운트 다운을 하는 동안 동시에 사용자의 입력을 받도록 대기할 수 있으며 “Time out!” 메시지가 출력되면 스레드를 취소하여 더 이상 사용자로부터 입력을 기다리지 않도록 한다. “Time out!” 메시지를 출력한 후 현재 퀴즈까지의 점수와 정답률을 출력하고 화면을 깨끗하게 정리한다. 이 작업을 서버로부터 퀴즈 수신이 끝나 “end”를 수신할 때까지 반복하게 된다.

1. enter\_answer(void \*Fd) :

스레드를 위한 함수로, scanf 함수를 통해 클라이언트로부터 입력된 답을 받아 서버로 전달하는 역할을 한다. 문제를 풀기 위해 10초의 카운트다운이 진행될 동안 사용자가 답을 입력할 수 있게 한다.

1. Result(int fd) :

퀴즈쇼가 끝난 후, 참여했던 클라이언트들의 점수와 정답률을 알려주기 위한 함수이다. 서버로부터 클라이언트들의 점수와 정답률을 받아 출력한다.