PA3 보고서

2018312280 이상수

Client:

확인:
우선 arg 가 3 개인 경우와 4 개인 경우로 나눠서, stdin, file 중 무엇을 읽는지 나눴습니다.

모든 입력에 대해 숫자공백숫자공백숫자인지 확인했습니다. 만약 공백이 맨처음에 오면 invalid, 공백이 숫자사이에 2 개이상 오면 invalid, 공백이 맨 뒤에 추가로 들어오면 invalid 처리하였습니다.
그후 errno=0 설정을하고, strtok 로 나눈 숫자를 하나씩 struct 에 각 변수에 저장했습니다. 그리고 저장하기전에 항상 strtok 로 나눈 값이 null 인지 확인하고 null 이면 저장하지 않거나 맞지 않는 입력이면 invalid 를 출력했습니다. 또한 null 이 아니라도, strtol 로 struct 에 저장할 때 errno 에 변화가 생기면 invalid 로 처리했습니다.
또한 user 가 0~1023 이 아니거나, action 이 0~5 가 아니면(0 은 추가로 나머지 요소도 0 인지 확인) invalid 입니다.

그리고 struct 의 필요한 요소에 숫자를 못 넣거나, 넣고나서, 여전히 숫자가 뒤에 남아있으면, (숫자 공백 숫자 공백 숫자 이외는)invalid 로 하였습니다.

실행:
대부분 struct 를 보내고, 성공시 1, 실패시 -1 을 되돌려 받아서 결과 출력을 진행했습니다.
2,4 번 액션의 경우만 성공시 좌석번호를 return 햇습니다.

Action3 의 경우에만 좌석번호들이 들어있는 int array 로 받아서 그 좌석중 1 이 들어간 좌석은 for 문으로 출력했습니다.
또한 action 이 0 인경우 서버에서 0 0 0 인지 판단하여 256 을 돌려주었고, 256 을 받으면 while 문을 break 하고 종료됩니다.

File 을 읽는 경우도 한줄씩 읽어서 위와 같이 진행하였습니다.

한 줄을 읽는 버퍼의 최대 길이는 100 입니다.

Server:

Thread 방식:

이전에 배운대로 connfd 를 받아서 연결 실패시 오류를 출력하고 성공 시 넘어와서 pthread 를 생성한 다음 pthread 는 detach 하였습니다.

thread 에서 우선 무엇이든 입력 받으면 mutex 로 전역변수들을 잡았습니다. 그후 처리가 끝나면 unlock 을 해주거나, 0 0 0 을 받아서 while 문이 끝나면 unlock 을 해주었습니다.

Action 처리:

Ac1 의 경우 우선 현재 사용중인 유저가 있는지 확인하고
(따로 use 변수를 만들어 -1 을 저장. -1 이면 아무도 클라이언트를 사용중이 아님. -1 이 아니면 사용중인 user 의 값이 들어가있음)
data 값이 음수가 아닌지 확인한 다음 (비밀번호)
사용중인 유저가 없으면 진행.

"upass 라는 유저와 비밀번호를 저장하는 전역 array"에 입력 받은 유저의 비밀번호가 저장되어 있지 않으면(초기값 -1) 새로운 유저로 인식하고 지금 받은 data 를 비밀번호로 저장했습니다.
비밀번호가 저장되어있으면 그 비밀번호와 맞는지 확인후 맞으면 로그인, 아니면 실패햇습니다.
당연히 모든 위에 작업이 성공할 경우 use=user 로 현재 사용중인 유저가 있음을 유저에 저장해주었고, 이는 나중에 혹시라도 현재 사용중인 유저와 다른 유저를 로그아웃시키거나 그런 사용자가 사용하는지 방지합니다.
예를 들어 3 1 1 로 3 번유저가 로그인했는데, 다른 클라이언트에서 5 1 1 로 5 번유저가 로그인햇는데, 3 번유저의 클라이언트에서 5 번 유저가 로그아웃을 시도하거나 하면 use 와 user 가 맞는지 확인하여 틀리니 실패처리합니다.
또한 위에 작업이 무엇이든 성공하면(로그인하면) 전역 array ulog[0~1023]에 해당 유저의 자리에 1 을 넣어서, (초기값, 로그아웃시 0) 그 유저가 로그인중이란 것을 다른 클라이언트들도 알 수 있게 해주었습니다.

Action 0 는 유저와 데이터값도 0 인지 확인하고, struct 의 변수가 0 0 0 일 경우 ulog 의 use 자리에 (ulog 의 현재 유저 자리에) 0 을 넣고(로그아웃), use 는 -1 로 해준 뒤 (클라이언트 사용중인 유저 없음) 클라이언트에 256 을 넘겨주고 break 하여 thread 의 while 에서 탈출해 mutex 를 해제하고 thread func 은 종료됩니다.

Ac 2,3,4,5 는 유선 ulog 에 입력받은 user 자리가 1 인지(로그인 되어있는지), 현재 use 가 user 인지 (시도하는 유저가 사용중이던 유저가 맞는지) 확인후 실행합니다.

Ac2 는 우선 data 가 0 이상, 255 이하인지 확인, 현재 그 data 자리가 seat array 에 -1 인지(초기값) 확인 후 둘다 해당하면 그 자리에 user 를 저장하고 자리값을 리턴합니다

Ac3 는 모든 seat 을 for 로 돌면서 user 의 값이 들어있는지 확인해 있으면 int len 에 그 값을 대입, 새로운 array 좌석에 그 자리에 1 대입.
만약 len 이 0 이면 -1 리턴.
Len 이 0 이 아니면, 그 len 을 리턴하여 클라이언트는 마지막 자리를 받음. 그 후 array 를 리턴하여 클라이언트는 그 array 를 받아서 "seat, "로 출력하고 마지막에는 "seat/n"로 출력해야 하기에 len 을 활용하여 마지막 자리만 따로 출력.

Ac4 는 2 처럼 data 가 범위에 맞는지 확인하고, seat 을 for 로 돌아서 해당 유저의 자리가 1 개라도 있는지 확인하고 있으면 진행 아니면 오류, 입력받은 자리가 seat 배열에서 -1 이면 오류 출력 (아무도 예약한적없음), 아니고 해당자리에 유저의 아이디가 저장되어 있으면 그 자리 리턴, 아니면 다른 유저가 저장되어 있으면 오류를 리턴 합니다.

Ac5 는 유저가 현재 use 와 같으면 진행 아니면 오류, 진행되면 ulog 의 user 자리에 0 입력, use 는 -1 로, 그 후 1 을 리턴 합니다.

Ac 처리가 모두 끝나면 mutex 를 unlock 하고 다시 위로 올라가서 입력을 기다립니다.