시스템 프로그래밍 보고서

7팀: 강승우, 문준범, 박서린, 정제현

1. 프로그램 개요

1) 프로그램 이름: 조커 뽑기 게임

3) 주제: 소켓통신을 이용한 “도둑 잡기 카드게임” 제작

2) 게임 규칙

(1) 조커 한 장을 포함한 53장의 카드를 네명의 플레이어에게 분배한다

(2) 분배 받은 카드들을 보고, 숫자 or 글자가 같은 모든 카드 쌍들을 버린다.

참여자들이 카드 버리는 것을 완료하면 게임이 시작된다

(3) (반)시계방향 순으로 진행하며, 자신의 턴이 되면 다음 사람의 카드를 한 장

가져온다.

(4) 가져온 카드와 짝이 맞는 카드가 있으면 버리고 아니면 가진다.

(5) (3),(4)의 과정을 반복하여 자신이 가진 카드를 모두 버리면 게임에서 승리.

(6) 마지막에 조커를 가진 사람이 게임에서 패배한다.

(\* 카드를 버릴 때는 반드시 한 쌍씩 버린다.)

3) 프로그램 작동 원리

-서버에서 4개의 클라이언트가 연결될 때까지 기다린다.

-4개의 클라이언트가 연결되면 게임을 시작한다.

-서버는 카드를 섞어 각 클라이언트에게 분배한다.

각 클라이언트에게 분배한 카드 중 버릴 수 있는 카드 쌍을 자동으로 버린다.

-모든 클라이언트가 시작할 준비가 되면 서버에 지정된 순서대로 턴이 시작된다.

-current client는 서버로부터 next client가 가진 카드의 개수(n)를 전달 받는다.

N 이하의 index를 current client가 서버에게 전달하여 해당 인덱스의 카드를

가져온다. 버릴 수 있는 카드는 서버에서 자동으로 버린다.

-턴이 종료되면 다음 클라이언트에게 턴이 넘어간다. 위의 과정을 반복한다.

-클라이언트의 카드가 모두 버려지면 승리. 승리한 클라이언트를 제외하고,

나머지 클라이언트들의 순서를 재조정한다.

-승리한 클라이언트 3개가 나오면 조커를 확인하고 게임 종료.

2-1. 코드 분석(server.c)

1) main 함수:

(1) 카드 섞기 밑 분배

Make\_card(): 조커를 포함한 53장의 카드 생성

Shuffle\_card(): 카드를 섞는다.

Distribute\_card(): 클라이언트에게 줄 카드를 분배한다.

(2) 서버 생성

Socket(PF\_INET, SOCK\_STREAM,0): IPv4 인터넷 프로토콜 환경으로 생성

S\_addr,sub\_addr.s\_addr=htonl(INADDR\_ANY): IP주소 생성

S\_addr.sin\_port=htons(atoi(argv[1])): 사용자가 입력한 포트번호로 설정

Bind(~): 소켓과 주소를 연결하여 서버 생성

(3) client 받기

Listen(s\_socket, MAX\_CLIENT): MAX\_CLIENT만큼 개통 요청.

Client 연결을 받는다.

while문{

C\_socket=accept(~): client 연결 요청 수신.

C\_socket에 클라이언트 소켓 번호 저장

Res=pushClient(c\_socket): 소켓 번호를 배열에 추가한다.

If res<0 : 연결 실패 문자열 출력, 해당 client의 소켓 close

Else: 연결 성공 문자열 출력, do\_game()을 실행하는 thread 생성

}

2)do\_game() 함수:

(1) 게임 시작:

If(count\_num==MAX\_CLIENT && !strat\_flat): 4명의 client와 연결되고

시작을 안했을 경우

{ client 주소를 파일 형식으로 fdopen

게임시작 문구 출력

Print\_card(): 각각의 client에게 각자의 카드 정보 출력

Discard(): 버릴 수 있는 카드를 한 쌍씩 버린다.

Start\_flag=TRUE: 게임이 시작했음을 알린다.

}

(2) 승패 확인:

Loser\_check(current\_turn): 현재진행중인 client 확인

만약, current client만 있다면

패배 문자열 해당 client에게 출력

PopClient(): client 정보 삭제

종료

(3) 카드 고르기:

고를 수 있는 index의 문자열로 출력

사용자가 select [index]를 입력하여 뽑을 카드 번호 입력받음

If((c\_socket-4)==current\_turn): 만약 현재 진행자가 입력한 거라면

Sub=strstr(chatData,”select”): select부터의 문자열 저장

Recv=sub[7]-’0’: 카드 인덱스 저장

Bring\_card(): 해당 인덱스의 카드 선택

Discard\_card(): 버릴 수 있는 카드 짝은 버린다.

Select\_flag=TRUE: 선택했음을 알린다.

(4) 턴 진행:

If(select\_flag && sub): 만약 카드를 골랐다면

Print\_card(): current client에게 카드 정보 출력

Change\_turn(): 턴 진행

(5) 채팅:

If(list\_c[I]!=INVALID\_SOCK): 만약 유효한 소켓 번호라면

=> write(list\_c[],chatData,n): 채팅 정보 client들에게 출력

If(strstr(chatData, escape)!=NULL): 만약 client가 escape를 입력했다면

=> popClient(c\_socket): 해당 socket정보 제거

3) pushClient, popClient 함수(client 관련 함수):

(1)pushClient 함수: racing condition을 막기 위해 mutex\_lock을 한다.

List\_c[]의 정보 변경한다.

Mutex\_unlock(): racing condition이 끝났으므로 unlock

(2)popClient 함수: 해당 소켓 삭제

Racing condition을 막기 위해 mutext\_lock 한다.

해당 소켓의 정보를 list\_c[]에서 삭제

Mutex\_unlock(): racing condition이 끝났으므로 unlock

4) make\_card, print\_card, shuffle\_card, distribute\_card 함수(카드 관련 함수):

(1)make\_card: for문 안에서 card[]에 52장의 카드 정보를 저장한다

for문이 끝나고 card[0]에 조커 정보 입력

(2)print\_card: player[][]에 저장된 카드 정보를 for문을 사용해서 출력

만약 한 장 더 가진 플레이어라면

=>남은 한 장의 카드 정보를 출력한다.

(3)shuffle\_card(): 랜덤하게 섞는 횟수(40~60), 인덱스(0~52)를 설정한다.

card[src]와 card[dst]의 자리를 바꿔준다.

(4)distribute\_card: one\_more=rand()%4: 한 장 더 받을 client 무작위 설정

Player[client\_index][card\_index]=card[]: card[]정보를

player[][] 배열에 저장

One\_more\_client는 마지막 카드를 더 가져간다.

4)discard\_card, change\_turn, bring\_card, loser\_check 함수(게임 진행 관련):

(1)discard\_card: player[][I]와 player[][j] 비교(0<=I<13, I+1<=j<14)

If, 유효한 카드 num이고 서로 같다면

=>player[][I]=-1, player[][j]=-1: 해당 인덱스의 카드 정보 삭제

=>card\_num[]-=2: 해당 client의 카드 개수 줄임

(2)change\_turn: if(card\_num[next\_turn[turn]]==0): 다음 client에게 카드가 없다면

=>종료 문자열 출력

=>next\_turn[turn]=next\_turn[next\_turn[turn]]: 끝난 사람 제외

Else if(card\_num[turn]==0): current client의 카드가 없다면

=>끝난 사람은 제외하고 순서변경

(3)bring\_card: player[current\_turn][I].num==-1

:current client의 유효하지 않은 카드 인덱스를

target\_index에 저장

사용자가 입력한 인덱스와 target\_index의 정보를

서로 바꾼다.

(4)loser\_check: 만약 플레이어가 자기만 남았다면 TRUE return.

2-2. 코드 분석(client.c)

1) main 함수:

(1) 서버 생성

Socket(PF\_INET, SOCK\_STREAM,0): IPv4 인터넷 프로토콜 환경으로 생성

C\_addr.sin\_addr.s\_addr=inet\_addr(argv[1]): 사용자가 입력한 주소로 서버

생성

S\_addr.sin\_port=htons(atoi(argv[2])): 사용자가 입력한 포트번호로 설정

닉네임을 입력 받는다.

Connect(~): 설정한 주소를 통해 연결을 시도한다.

(2) 스레드 생성

연결된 소켓을 이용하여 do\_send\_chat, do\_receive\_chat 함수를

실행하는 thread를 만든다.

2) do\_send\_chat, do\_receive\_chat 함수(채팅 관련 함수):

(1)do\_send\_chat:

If, buf에 사용자 입력이 들어온다면

=> client 닉네임과 함께 붙여서 buf를 소켓에 출력

=>if, 종료 문자열이 들어온다면

==>pthread\_kill(): 현재 스레드 삭제

(2)do\_receive\_chat:

If, 서버로부터 입력이 들어온다면

=>chatData 문자열 표준출력