|  |  |
| --- | --- |
| 학번/이름 | 5049986 / 엄승욱 |

목차

1. 프로토콜 ------------------------------------------------------------------------------- [1](#프로토콜)
2. 실행화면 ------------------------------------------------------------------------------- [3](#실행화면)  
   1) 서버 ------------------------------------------------------------------------- [3](#실행화면_서버)  
   2) 클라이언트 ------------------------------------------------------------------------- [4](#실행화면_클라이언트)
3. 코드 ------------------------------------------------------------------------------- [6](#코드)  
   1) 프로토콜 처리 코드 --------------------------------------------------------------- [6](#프로토콜_처리_코드)  
   2) 서버 코드 --------------------------------------------------------------- [17](#서버_코드)  
   3) 클라이언트 코드 --------------------------------------------------------------- [28](#클라이언트_코드)
4. 실행방법 ------------------------------------------------------------------------------ [43](#실행방법)

1. 프로토콜

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **코드** | **방향** | **설명** | **파라미터** |
| 1 | 10, 100  (Login) | C->S  S->A | 입력 받은 ID를 서버에 보내서 접속하고 모든 클라이언트에게 알림 | 접속한 사람의 ID(userId) |
| 2 | 11, 101  (Chat) | C->S  S->A | 로그인 되었을 때, 서버에 채팅 내용을 보내면 서버에서 보낸 유저의 ID와 채팅내용을 접속한 모든 클라이언트에게 보냄 | C->S  채팅 내용(chat)  S->A  유저ID(userId), 채팅내용(chat) |
| 3 | 12, 103  (Ready) | C->S  S->C | 클라이언트에서 코드를 보내면 서버에서 보낸 클라이언트의 준비상태를 바꾸고 다른 유저를 찾는 스레드를 만듦(스레드는 최초에만 생성), 이후 준비완료 메시지 출력하게 함 | 없음 |
| 4 | 104  (Start) | S->P1, P2 | 2명의 준비 상태의 클라이언트에서 게임이 돌아갈 수 있게 하기위해 클라이언트에 코드를 보내서 게임시작 상태로 바꿈, 플레이어로 지정된 사람에게 보냄 | S->P1  P2의 ID(p2.userId)  S->P2  P1의 ID(p1.userId) |
| 5 | 13, 105  (Select) | P->S  S->P1, P2 | 플레이어가 항목 선택 시 항목의 인덱스와 ID를 서버로 보내서 확인 후 순서 등의 조건이 맞으면 모든 플레이어에게 바뀌는 것들을 보냄 | P->S  항목 인덱스(num),  Player ID(userId)  S->P1, P2  OorX(flag a),항목 인덱스(num) |
| 6 | 106  (Result) | S->P1, P2 | 승리조건을 확인 후 누군가 승리조건에 부합하면 플레이어에 코드를 보내 승, 무, 패를 결정하는 한 문자를 보내서 결정 | 승자 : 승리 문자(w)  패자 : 패배 문자(l)  비김 : 비김 문자(d)  (draw는 모든 플레이어에게 보냄) |
| 7 | 107, 14  (Init) | S->P1, P2  P1, P2->S | 게임이 끝나면 초기 상태로 돌리기 위한 작업, 서버에서 먼저 게임시작에 필요한 값들로 초기화를 한 후 플레이어에게 보내서 각 플레이어마다의 게임 값을 초기화 하고 초기화되었다는 확인으로 서버에게 보내서 준비상태를 해제함 | 없음 |
| 8 | 102  (Error) | S->C | 위의 코드 처리 중 논리적으로 위배되는 경우(자기 차례가 아닌데 항목을 선택하거나 )에 이 코드를 보냄 | Error 내용 |
| 9 | 15, 108  (order) | P->S  S->P1, P2 | 시작하면 먼저 할 사람을 보내 줌 | P->S : 없음  S->P1, P2 : 먼저 하는 유저 ID(p1.userId or p2.userId) |
| 10 | 16, 109  (List) | C->S  S->C | 현재 접속자 명단을 알려줌 | C->S : 없음  S->C : 현재 접속자 ID 목록(userID) |

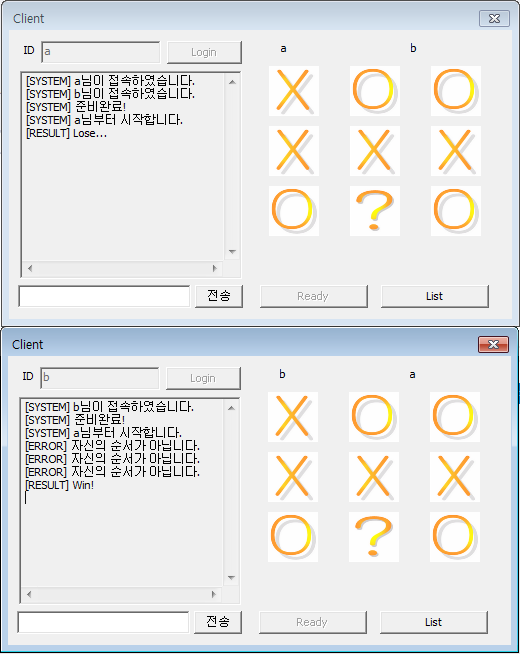
\* 부가설명  
1) 기본적으로 모든 프로토콜 코드를 보냅니다. 이러한 이유는 내부 조작으로 가능한 기능의 경우 프로토콜 코드만 보내서 실행시킬 수 있고, 코드 전송이 제대로 되었는지 확인하기 위함입니다.  
2) 코드를 2/3자리수로 나눈 이유는 누가 송신하는지를 구분하기 위함과 다른 코드에서도 사용할 수 있게 하기 위함입니다.(클라이언트->서버 : 2자리수 코드, 서버->클라이언트 : 3자리수 코드)  
3) P는 player1 or player2를 뜻하고 A는 모든 클라이언트를 뜻합니다.  
4) 여기서 모든 클라이언트는 현재 접속하고 있는 상태의 클라이언트입니다.

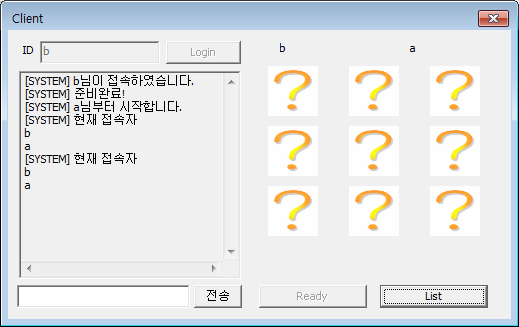
2. 실행화면  
1) 서버

Loading Success  
Login On  
a 접속  
Login 완료  
Login On  
b 접속  
Login 완료  
Ready On  
준비 플래그 완료  
Ready On  
준비 플래그 완료  
게임 시작  
순서 보내기  
순서 보내기  
a : 2번 선택  
b : 5번 선택  
a : 4번 선택  
b : 9번 선택  
a : 1번 선택  
b : 3번 선택  
chat on  
b : 쩝  
전송 성공  
chat on  
a : ㅅㄱㅇ  
전송 성공  
a : 7번 선택  
p1 승리  
초기화 시작  
리스트 출력  
b 나감  
a 나감

\* 추가설명  
(1) 순서는 a접속=>b접속=>a준비=>b준비=>게임시작=>게임중=>채팅=>게임결과=>게임종료 및 초기화=>클라이언트 종료 순으로 작동합니다.

2) 클라이언트

  
그림1. 기본적인 클라이언트 작동 화면

  
그림2. 접속자 현황 출력

3. 코드

3.1 프로토콜 처리 코드

|  |  |
| --- | --- |
| 1.  10  (login)  C->S | case IDC\_BUTTON\_LOGIN:  EnableWindow(hButtonLogin, FALSE);  GetDlgItemText(hDlg, IDC\_EDIT\_ID, userId, 20);  if(strlen(userId) == 0)  EnableWindow(hButtonLogin, TRUE);  else{  EnterCriticalSection(&cs);  opCode = OP\_LOGIN;  retval = send(sock, (char \*)&opCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  retval = send(sock, userId, 20, 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  EnableWindow(hEditId, FALSE);  LeaveCriticalSection(&cs);  }  return TRUE; |
| 100  (login)  S->A | case OP\_LOGIN:  printf("Login On \n");  //ID받기  retval = recv(sock, ci[index].userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Receive Error2!\n");  return -1;  }  else if(retval==0) return 0;  ci[index].userId[retval]=NULL;  printf("%s 접속\n", ci[index].userId);  //시스템 출력 명령어 전송  sendCode=100;  for(int re=0;re<3;re++){  if(ci[re].userId[0]!=NULL){  retval = send(ci[re].clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error1!\n");  return -1;  }  //출력문 전송  retval = send(ci[re].clientSock, ci[index].userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error2!\n");  return -1;  }  }  }  printf("Login 완료 \n");  break; |
| recv | case OP\_SYSTEM\_LOGIN:  char connectId[20];  EnterCriticalSection(&cs);  retval = recv(sock, connectId, 20, 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] ");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)connectId);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"님이 접속하였습니다.");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 2.  11  (chat)  C->S | case IDC\_BUTTON\_SEND:  EnableWindow(hButtonSend, FALSE);  GetDlgItemText(hDlg, IDC\_EDIT\_INPUT, chat, BUFSIZE);  if(strlen(userId) == 0 || userId[0]==NULL);  else{  EnterCriticalSection(&cs);  opCode = OP\_CHAT;  retval = send(sock, (char \*)&opCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  retval = send(sock, chat, BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  LeaveCriticalSection(&cs);  }  EnableWindow(hButtonSend, TRUE);  return TRUE; |
| 101  (chat)  S->A | case OP\_CHAT:  printf("chat on \n");  EnterCriticalSection(&cs);  //출력문 받기  retval = recv(sock, chat, BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Receive Error3!\n");  return -1;  }  else if(retval==0) return 0;  chat[retval] = NULL;  printf("%s : %s \n", ci[index].userId, chat);  //유저가 있는 경우  if(ci[index].userId[0]!=NULL){  for(int i=0;i<3;i++){  if(ci[i].userId[0]!=NULL){  ZeroMemory(&sendCode, sizeof(byte));  sendCode=OP\_SYSTEM\_USER;  retval = send(ci[i].clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error3!\n");  return -1;  }  retval = send(ci[i].clientSock, ci[index].userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error4!\n");  return -1;  }  retval = send(ci[i].clientSock, chat, BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error5!\n");  return -1;  }  }  }  printf("전송 성공 \n");  } |
| recv | case OP\_SYSTEM\_USER:  EnterCriticalSection(&cs);  retval = recv(sock, aUserId, 20, 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)aUserId);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)": ");  retval = recv(sock, chat, BUFSIZE, 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)chat);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  SetWindowText(hEditInput, "");  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 3.  12  (ready)  C->S | case IDC\_BUTTONREADY:  if(userId[0]==NULL);  else{  EnableWindow(hButtonReady, FALSE);  EnterCriticalSection(&cs);  opCode = OP\_READY;  retval = send(sock, (char \*)&opCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  LeaveCriticalSection(&cs);  }  return TRUE; |
| 103  (ready)  S->C | case OP\_READY:  printf("Ready On \n");  EnterCriticalSection(&cs);  if(checkReadyThread==NULL){  checkReadyThread = CreateThread(NULL, 0, checkReady, NULL, 0, &ThreadId);  if(checkReadyThread==NULL) printf("Thread Error!\n");  }  sendCode=OP\_SYSTEM\_READY;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error8!\n");  return -1;  }  printf("준비 플래그 완료 \n");  ci[index].readyFlag=TRUE;  LeaveCriticalSection(&cs);  break; |
| recv | case OP\_SYSTEM\_READY:  EnterCriticalSection(&cs);  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] 준비완료!");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  for(int k=0;k<9;k++){  SendMessage(hImg[k], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[0]);  }  SetWindowText(hP1, userId);  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 4.  104  (start)  S->P1, P2 | EnterCriticalSection(&cs);  retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_Player1\_Code!\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_Player2\_Code!\n");  return -1;  }  retval = send(p1.clientSock, p2.userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_Player1\_Name!\n");  return -1;  }  retval = send(p2.clientSock, p1.userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_Player2\_Name!\n");  return -1;  }  printf("게임 시작\n");  LeaveCriticalSection(&cs); |
| recv | case OP\_SYSTEM\_START:  EnterCriticalSection(&cs);  sendCode = OP\_ORDER;  retval = recv(sock, aUserId, 20, 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  SetWindowText(hP2, aUserId);  startFlag=true;  …  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 5.  13  (select)  P->S | case IDC\_PLT:  if(startFlag){  if(sel\_Img[0]==false){  opCode = OP\_SELECT;  byte num=1;  retval = send(sock, (char \*)&opCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  retval = send(sock, userId, 20, 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  retval = send(sock, (char \*)&num, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  }  else{  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  }  }  else{  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  }  return TRUE; |
| 105  (select)  S->P1, P2 | case OP\_SELECT:  char checkId[21];  ZeroMemory(checkId, sizeof(checkId));  EnterCriticalSection(&cs);  retval = recv(sock, checkId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Receive Error\_SELECT\_NUM1!\n");  return -1;  }  checkId[retval]=NULL;  retval = recv(sock, (char \*)&num, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Receive Error\_SELECT\_NUM2!\n");  return -1;  }  if(orderFlag){  if(sel\_Img[num-1]){  sendCode=OP\_SYSTEM\_ERROR;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  retval = send(sock, "이미 선택된 항목입니다.", BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  }  else if(strcmp(checkId, p1.userId)==0){  sendCode = OP\_SHOW\_IMAGE;  bool a = true;  retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p1.clientSock, (char \*)&a, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&a, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p1.clientSock, (char \*)&num, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&num, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  sel\_Img[num-1] = true;  p1\_sel[num-1] = true;  playNum = checkPlay();  orderFlag=false;  printf("%s : %d번 선택 \n", checkId, (int)num);  }  else{  sendCode=OP\_SYSTEM\_ERROR;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  retval = send(sock, "자신의 순서가 아닙니다.", BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  }  }  else{  if(sel\_Img[num-1]){  sendCode=OP\_SYSTEM\_ERROR;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  retval = send(sock, "이미 선택된 항목입니다.", BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  }  else if(strcmp(checkId, p2.userId)==0){  sendCode = OP\_SHOW\_IMAGE;  bool a = false;  retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p1.clientSock, (char \*)&a, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&a, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p1.clientSock, (char \*)&num, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&num, sizeof(bool), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_105!\n");  return -1;  }  sel\_Img[num-1] = true;  p2\_sel[num-1] = true;  playNum = checkPlay();  orderFlag=true;  printf("%s : %d번 선택 \n", checkId, (int)num);  }  else{  sendCode=OP\_SYSTEM\_ERROR;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  retval = send(sock, "자신의 순서가 아닙니다.", BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  }  } |
| recv | case OP\_SHOW\_IMAGE:  EnterCriticalSection(&cs);  bool oxFlag;  byte num;  retval = recv(sock, (char \*)&oxFlag, sizeof(bool), 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  retval = recv(sock, (char \*)&num, sizeof(byte), 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  if(oxFlag){  SendMessage(hImg[num-1], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[1]);  }  else{  SendMessage(hImg[num-1], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[2]);  }  sel\_Img[num-1]=true;  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 6.  106  (result)  S->P1, P2 | case OP\_SELECT:  …  switch (playNum)  {  case 1:  sendCode=OP\_PLAY\_RESULT;  retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_1\n");  return -1;  }  retval = send(p1.clientSock, "w", 1, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_2\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_3\n");  return -1;  }  retval = send(p2.clientSock, "l", 1, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_4\n");  return -1;  }  printf("p1 승리 \n");  endFlag = true;  break;  case 2:  sendCode=OP\_PLAY\_RESULT;  retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_5\n");  return -1;  }  retval = send(p1.clientSock, "l", 1, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_6\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_7\n");  return -1;  }  retval = send(p2.clientSock, "w", 1, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_8\n");  return -1;  }  printf("p2 승리 \n");  endFlag = true;  break;  case 3:  sendCode=OP\_PLAY\_RESULT;  retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_9\n");  return -1;  }  retval = send(p1.clientSock, "d", 1, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_10\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_11\n");  return -1;  }  retval = send(p2.clientSock, "d", 1, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_106!\_12\n");  return -1;  }  printf("무승부 \n");  endFlag = true;  break;  default:  break;  } |
| recv | case OP\_PLAY\_RESULT:  EnterCriticalSection(&cs);  retval = recv(sock, (char \*)&result, sizeof(char), 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  nLength = GetWindowTextLength(hEditChat);  switch(result){  case 'w':  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[RESULT] Win!");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  break;  case 'l':  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[RESULT] Lose...");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  break;  case 'd':  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[RESULT] Draw");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  break;  default:  break;  }  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 7.  107  (init)  C->P1, P2 | case OP\_SELECT:  …  if(endFlag){  printf("초기화 시작 \n");  for(int i=0;i<9;i++){  sel\_Img[i]=false;  p1\_sel[i]=false;  p2\_sel[i]=false;  }  sendCode = OP\_PLAY\_INIT;  retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_end1\n");  return -1;  }  retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_end2\n");  return -1;  }  playNum=0;  endFlag=false;  }  LeaveCriticalSection(&cs);  break; |
| 14  (init)  P1, P2  ->S | case OP\_PLAY\_INIT:  EnterCriticalSection(&cs);  Sleep(2000);  for(int i=0;i<9;i++){  sel\_Img[i]=false;  SendMessage(hImg[i], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[0]);  }  SetWindowText(hP1, "Player1");  SetWindowText(hP2, "Player2");  startFlag=false;  opCode = OP\_INIT;  retval = send(sock, (char \*)&opCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return -1;  }  EnableWindow(hButtonReady, true);  LeaveCriticalSection(&cs);  break; |
| recv | case OP\_INIT:  EnterCriticalSection(&cs);  ci[index].readyFlag=false;  checkReadyThread=NULL;  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 8.  11  (error)  S->C | Case OP\_CHAT:  …  //로그인을 안했을 시  else{  sendCode=OP\_SYSTEM\_ERROR;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error6!\n");  return -1;  }  retval = send(sock, "아직 접속하지 않았습니다.", BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error7!\n");  return -1;  }  printf("접속 요구 완료 \n");  }  … |
| 프로토콜 5번 내의 ERROR code 전송 참고  else{  sendCode=OP\_SYSTEM\_ERROR;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  retval = send(sock, "자신의 순서가 아닙니다.", BUFSIZE, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_102!\n");  return -1;  }  } |
| recv | case OP\_SYSTEM\_ERROR:  EnterCriticalSection(&cs);  retval = recv(sock, buf, BUFSIZE, 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[ERROR] ");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)buf);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 9.  15  (order)  P->S | case OP\_SYSTEM\_START:  …  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR) break;  … |
| 108  (order)  S->P1, P2 | case OP\_ORDER:  EnterCriticalSection(&cs);  sendCode = OP\_SYSTEM\_ORDER;  printf("순서 보내기\n");  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_end1\n");  return -1;  }  if(orderFlag){  retval = send(sock, p1.userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_end1\n");  return -1;  }  }  else{  retval = send(sock, p2.userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_end1\n");  return -1;  }  }  LeaveCriticalSection(&cs);  break; |
| recv | case OP\_SYSTEM\_ORDER:  EnterCriticalSection(&cs);  ZeroMemory(orderId, sizeof(orderId));  retval = recv(sock, orderId, 20, 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] ");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)orderId);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"님부터 시작합니다.");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  LeaveCriticalSection(&cs);  break; |

|  |  |
| --- | --- |
| 10.  16  (list)  C->S | case IDC\_BUTTONLIST:  if(userId[0]==NULL);  else{  EnterCriticalSection(&cs);  opCode = OP\_LIST;  retval = send(sock, (char \*)&opCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  return FALSE;  }  LeaveCriticalSection(&cs);  }  return TRUE; |
| 109  (list)  S->A | case OP\_LIST:  printf("리스트 출력\n");  z=0;  EnterCriticalSection(&cs);  sendCode = 109;  retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_List1\n");  return -1;  }  retval = send(sock, (char \*)&ccnt, sizeof(int), 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_List1\n");  return -1;  }  while(z<3){  if(ci[z].userId[0]==NULL);  else{  retval = send(sock, ci[z].userId, 20, 0);  if(retval==SOCKET\_ERROR){  printf("Send Error\_List1\n");  return -1;  }  }  z++;  }  LeaveCriticalSection(&cs);  break; |
| recv | case OP\_CLIENT\_LIST:  EnterCriticalSection(&cs);  retval = recv(sock, (char \*)&cnt, sizeof(int), 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  nLength = GetWindowTextLength(hEditChat);  SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] 현재 접속자 \r\n");  for(int z=0;z<cnt;z++){  char tempId[20];  retval = recv(sock, tempId, 20, 0);  if(retval==SOCKET\_ERROR) break;  else if(retval==0){  closesocket(sock);  ExitThread(-1);  }  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)tempId);  SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");  }  LeaveCriticalSection(&cs);  break; |

\* 부가설명  
1) recv의 의미는 각 프로토콜의 전송 구간의 마지막 recv지점을 보여주는 것으로 어떤 데이터를 보냈을 때, 처리는 어떻게 하는 것인지 보여주는 부분입니다.

3.2 서버 코드

#include <stdio.h>

#include <WinSock2.h>

#include <stdlib.h>

#define BUFSIZE 512

#define OP\_LOGIN 10

#define OP\_CHAT 11

#define OP\_READY 12

#define OP\_SELECT 13

#define OP\_INIT 14

#define OP\_ORDER 15

#define OP\_LIST 16

#define OP\_SYSTEM\_LOGIN 100

#define OP\_SYSTEM\_USER 101

#define OP\_SYSTEM\_ERROR 102

#define OP\_SYSTEM\_READY 103

#define OP\_SYSTEM\_START 104

#define OP\_SHOW\_IMAGE 105

#define OP\_PLAY\_RESULT 106

#define OP\_PLAY\_INIT 107

#define OP\_SYSTEM\_ORDER 108

#define OP\_CLIENT\_LIST 109

typedef struct clientInfo{

SOCKET clientSock;

SOCKADDR\_IN clientAddr;

char userId[21];

int index;

bool useFlag;

bool readyFlag;

}clientInfo;

clientInfo ci[3];

bool maxFlag=FALSE;

CRITICAL\_SECTION cs;

int ccnt=0;

int cindex=0;

clientInfo p1, p2;

HANDLE checkReadyThread=NULL;

bool sel\_Img[9]={false,};

bool p1\_sel[9]={false,};

bool p2\_sel[9]={false,};

bool orderFlag=true;

bool endFlag=false;

//게임의 승패를 결정짓는 확인 함수

int checkPlay(){

if(orderFlag){

if((p1\_sel[0] && p1\_sel[1] && p1\_sel[2]) || (p1\_sel[3] && p1\_sel[4] && p1\_sel[5]) ||

(p1\_sel[6] && p1\_sel[7] && p1\_sel[8]))

return 1;

else if((p1\_sel[0] && p1\_sel[3] && p1\_sel[6]) || (p1\_sel[1] && p1\_sel[4] && p1\_sel[7]) ||

(p1\_sel[2] && p1\_sel[5] && p1\_sel[8]))

return 1;

else if((p1\_sel[0] && p1\_sel[4] && p1\_sel[8]) || (p1\_sel[2] && p1\_sel[4] && p1\_sel[6]))

return 1;

}

else{

if((p2\_sel[0] && p2\_sel[1] && p2\_sel[2]) || (p2\_sel[3] && p2\_sel[4] && p2\_sel[5]) ||

(p2\_sel[6] && p2\_sel[7] && p2\_sel[8]))

return 2;

else if((p2\_sel[0] && p2\_sel[3] && p2\_sel[6]) || (p2\_sel[1] && p2\_sel[4] && p2\_sel[7]) ||

(p2\_sel[2] && p2\_sel[5] && p2\_sel[8]))

return 2;

else if((p2\_sel[0] && p2\_sel[4] && p2\_sel[8]) || (p2\_sel[2] && p2\_sel[4] && p2\_sel[6]))

return 2;

}

if(sel\_Img[0] && sel\_Img[1] && sel\_Img[2] && sel\_Img[3] && sel\_Img[4] &&

sel\_Img[5] && sel\_Img[6] && sel\_Img[7] && sel\_Img[8])

return 3;

else

return 0;

}

//준비 확인 스레드

DWORD WINAPI checkReady(LPVOID arg){

int retval;

byte sendCode = OP\_SYSTEM\_START;

bool esc=false;

int p1\_index;

while(1){

for(int i=0;i<3;i++){

if(ci[i].readyFlag){

p1=ci[i];

p1\_index=i;

esc=true;

break;

}

}

if(esc)

break;

}

esc=false;

while(1){

for(int i=0;i<3;i++){

if(ci[i].readyFlag && i!=p1\_index){

p2=ci[i];

esc=true;

break;

}

}

if(esc){

break;

}

}

EnterCriticalSection(&cs);

retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_Player1\_Code!\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_Player2\_Code!\n");

return -1;

}

retval = send(p1.clientSock, p2.userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_Player1\_Name!\n");

return -1;

}

retval = send(p2.clientSock, p1.userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_Player2\_Name!\n");

return -1;

}

printf("게임 시작\n");

LeaveCriticalSection(&cs);

ExitThread(0);

return 0;

}

//통신 스레드

DWORD WINAPI ClientProc(LPVOID arg){

SOCKET sock;

int retval, addrLen, index=(int)arg, playNum=0, z=0;

byte sendCode = 0, opCode = 0, num = 0;

SOCKADDR\_IN sockAddr;

char chat[BUFSIZE+1];

DWORD ThreadId;

sock = ci[index].clientSock;

//소켓 정보 저장

addrLen = sizeof(sockAddr);

getpeername(sock, (SOCKADDR \*)&sockAddr, &addrLen);

while(1){

ZeroMemory(&sendCode, sizeof(byte));

ZeroMemory(&opCode, sizeof(byte));

ZeroMemory(chat, sizeof(chat));

//명령어(opcode) 받아오기

retval = recv(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

break;

}

else if(retval==0) break;

//명령어 종류에 따른 기능 실행

switch(opCode){

//로그인

case OP\_LOGIN:

printf("Login On \n");

//ID받기

retval = recv(sock, ci[index].userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Receive Error2!\n");

return -1;

}

else if(retval==0) return 0;

//시스템 출력문 생성

ci[index].userId[retval]=NULL;

printf("%s 접속\n", ci[index].userId);

//시스템 출력 명령어 전송

sendCode=100;

for(int re=0;re<3;re++){

if(ci[re].userId[0]!=NULL){

retval = send(ci[re].clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error1!\n");

return -1;

}

//출력문 전송

retval = send(ci[re].clientSock, ci[index].userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error2!\n");

return -1;

}

}

}

printf("Login 완료 \n");

break;

//채팅

case OP\_CHAT:

printf("chat on \n");

EnterCriticalSection(&cs);

//출력문 받기

retval = recv(sock, chat, BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

printf("Receive Error3!\n");

return -1;

}

else if(retval==0) return 0;

chat[retval] = NULL;

printf("%s : %s \n", ci[index].userId, chat);

//유저가 있는 경우

if(ci[index].userId[0]!=NULL){

for(int i=0;i<3;i++){

if(ci[i].userId[0]!=NULL){

ZeroMemory(&sendCode, sizeof(byte));

sendCode=OP\_SYSTEM\_USER;

retval = send(ci[i].clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error3!\n");

return -1;

}

retval = send(ci[i].clientSock, ci[index].userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error4!\n");

return -1;

}

retval = send(ci[i].clientSock, chat, BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error5!\n");

return -1;

}

}

}

printf("전송 성공 \n");

}

//로그인을 안했을 시

else{

sendCode=OP\_SYSTEM\_ERROR;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error6!\n");

return -1;

}

retval = send(sock, "아직 접속하지 않았습니다.", BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error7!\n");

return -1;

}

printf("접속 요구 완료 \n");

}

LeaveCriticalSection(&cs);

break;

//준비코드 들어오면 준비 플래그 온

case OP\_READY:

printf("Ready On \n");

EnterCriticalSection(&cs);

if(checkReadyThread==NULL){

checkReadyThread = CreateThread(NULL, 0, checkReady, NULL, 0, &ThreadId);

if(checkReadyThread==NULL) printf("Thread Error!\n");

}

sendCode=OP\_SYSTEM\_READY;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error8!\n");

return -1;

}

printf("준비 플래그 완료 \n");

ci[index].readyFlag=TRUE;

LeaveCriticalSection(&cs);

break;

//이미지 선택 시 플레이어에 따른 모양 변경 및 승패 확인

case OP\_SELECT:

char checkId[21];

ZeroMemory(checkId, sizeof(checkId));

EnterCriticalSection(&cs);

retval = recv(sock, checkId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Receive Error\_SELECT\_NUM1!\n");

return -1;

}

checkId[retval]=NULL;

retval = recv(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Receive Error\_SELECT\_NUM2!\n");

return -1;

}

if(orderFlag){

if(sel\_Img[num-1]){

sendCode=OP\_SYSTEM\_ERROR;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

retval = send(sock, "이미 선택된 항목입니다.", BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

}

else if(strcmp(checkId, p1.userId)==0){

sendCode = OP\_SHOW\_IMAGE;

bool a = true;

retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p1.clientSock, (char \*)&a, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&a, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p1.clientSock, (char \*)&num, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&num, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

sel\_Img[num-1] = true;

p1\_sel[num-1] = true;

playNum = checkPlay();

orderFlag=false;

printf("%s : %d번 선택 \n", checkId, (int)num);

}

else{

sendCode=OP\_SYSTEM\_ERROR;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

retval = send(sock, "자신의 순서가 아닙니다.", BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

}

}

else{

if(sel\_Img[num-1]){

sendCode=OP\_SYSTEM\_ERROR;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

retval = send(sock, "이미 선택된 항목입니다.", BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

}

else if(strcmp(checkId, p2.userId)==0){

sendCode = OP\_SHOW\_IMAGE;

bool a = false;

retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p1.clientSock, (char \*)&a, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&a, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p1.clientSock, (char \*)&num, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&num, sizeof(bool), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_105!\n");

return -1;

}

sel\_Img[num-1] = true;

p2\_sel[num-1] = true;

playNum = checkPlay();

orderFlag=true;

printf("%s : %d번 선택 \n", checkId, (int)num);

}

else{

sendCode=OP\_SYSTEM\_ERROR;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

retval = send(sock, "자신의 순서가 아닙니다.", BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_102!\n");

return -1;

}

}

}

switch (playNum)

{

case 1:

sendCode=OP\_PLAY\_RESULT;

retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_1\n");

return -1;

}

retval = send(p1.clientSock, "w", 1, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_2\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_3\n");

return -1;

}

retval = send(p2.clientSock, "l", 1, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_4\n");

return -1;

}

printf("p1 승리 \n");

endFlag = true;

break;

case 2:

sendCode=OP\_PLAY\_RESULT;

retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_5\n");

return -1;

}

retval = send(p1.clientSock, "l", 1, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_6\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_7\n");

return -1;

}

retval = send(p2.clientSock, "w", 1, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_8\n");

return -1;

}

printf("p2 승리 \n");

endFlag = true;

break;

case 3:

sendCode=OP\_PLAY\_RESULT;

retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_9\n");

return -1;

}

retval = send(p1.clientSock, "d", 1, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_10\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_11\n");

return -1;

}

retval = send(p2.clientSock, "d", 1, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_106!\_12\n");

return -1;

}

printf("무승부 \n");

endFlag = true;

break;

default:

break;

}

if(endFlag){

printf("초기화 시작 \n");

for(int i=0;i<9;i++){

sel\_Img[i]=false;

p1\_sel[i]=false;

p2\_sel[i]=false;

}

sendCode = OP\_PLAY\_INIT;

retval = send(p1.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_end1\n");

return -1;

}

retval = send(p2.clientSock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_end2\n");

return -1;

}

playNum=0;

endFlag=false;

}

LeaveCriticalSection(&cs);

break;

case OP\_INIT:

EnterCriticalSection(&cs);

ci[index].readyFlag=false;

checkReadyThread=NULL;

LeaveCriticalSection(&cs);

break;

case OP\_ORDER:

EnterCriticalSection(&cs);

sendCode = OP\_SYSTEM\_ORDER;

printf("순서 보내기\n");

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_end1\n");

return -1;

}

if(orderFlag){

retval = send(sock, p1.userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_end1\n");

return -1;

}

}

else{

retval = send(sock, p2.userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_end1\n");

return -1;

}

}

LeaveCriticalSection(&cs);

break;

case OP\_LIST:

printf("리스트 출력\n");

z=0;

EnterCriticalSection(&cs);

sendCode = 109;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_List1\n");

return -1;

}

retval = send(sock, (char \*)&ccnt, sizeof(int), 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_List1\n");

return -1;

}

while(z<3){

if(ci[z].userId[0]==NULL);

else{

retval = send(sock, ci[z].userId, 20, 0);

if(retval==SOCKET\_ERROR){

printf("Send Error\_List1\n");

return -1;

}

}

z++;

}

LeaveCriticalSection(&cs);

break;

default:

printf("Not exist opCode!\n");

break;

}

}

printf("%s 나감\n", ci[index].userId);

closesocket(sock);

EnterCriticalSection(&cs);

ci[index].useFlag=FALSE;

ci[index].userId[0]=NULL;

ccnt--;

if(ccnt<3)

maxFlag=FALSE;

LeaveCriticalSection(&cs);

return 0;

}

int main(void){

int retval;

for(int i=0;i<3;i++){

ci[i].userId[0]=NULL;

}

InitializeCriticalSection(&cs);

//WinSock Initialize

WSADATA wsa;

if(WSAStartup(MAKEWORD(2,2), &wsa)!=0)

return -1;

//listenSock Creation

SOCKET listenSock = socket(AF\_INET, SOCK\_STREAM, 0);

if(listenSock==INVALID\_SOCKET){

printf("Socket Error!\n");

system("pause");

return -1;

}

//bind

SOCKADDR\_IN serverAddr;

ZeroMemory(&serverAddr, sizeof(serverAddr));

serverAddr.sin\_family=AF\_INET;

serverAddr.sin\_port=htons(9721);

serverAddr.sin\_addr.s\_addr=htonl(INADDR\_ANY);

retval=bind(listenSock, (SOCKADDR \*)&serverAddr, sizeof(serverAddr));

if(retval==SOCKET\_ERROR){

printf("Bind Error!\n");

system("pause");

return -1;

}

//listen

retval=listen(listenSock, SOMAXCONN);

if(retval==SOCKET\_ERROR){

printf("Listen Error!\n");

system("pause");

return -1;

}

for(int i=0;i<3;i++){

ci[i].useFlag=FALSE;

}

printf("Loading Success\n");

int addrLen;

HANDLE hThread;

DWORD ThreadId;

while(1){

//accept

int index=cindex;

if(ccnt<3){

addrLen = sizeof(ci[index].clientAddr);

ci[index].clientSock = accept(listenSock, (SOCKADDR \*)&ci[index].clientAddr, &addrLen);

if(ci[cindex].clientSock==INVALID\_SOCKET){

printf("Accept Error!\n");

continue;

}

//Thread on

hThread = CreateThread(NULL, 0, ClientProc, (LPVOID)index, 0, &ThreadId);

if(hThread==NULL) printf("Thread Error!\n");

else CloseHandle(hThread);

EnterCriticalSection(&cs);

ci[index].useFlag=TRUE;

ccnt++;

for(int k=0;k<3;k++){

if(!ci[k].useFlag){

cindex=k;

break;

}

}

LeaveCriticalSection(&cs);

}

if(ccnt>=3){

maxFlag=TRUE;

printf("최대 수용인원 달성.\n");

while(1){

if(!maxFlag){

break;

}

}

for(int i=0;i<=ccnt;i++){

if(!ci[i].useFlag){

EnterCriticalSection(&cs);

cindex=i;

LeaveCriticalSection(&cs);

break;

}

}

}

}

DeleteCriticalSection(&cs);

closesocket(listenSock);

WSACleanup();

return 0;

}

3.3 클라이언트 코드

#include <WinSock2.h>

#include <stdio.h>

#include <stdlib.h>

#include "resource.h"

#define BUFSIZE 512

#define OP\_LOGIN 10

#define OP\_CHAT 11

#define OP\_READY 12

#define OP\_SELECT 13

#define OP\_INIT 14

#define OP\_ORDER 15

#define OP\_LIST 16

#define OP\_SYSTEM\_LOGIN 100

#define OP\_SYSTEM\_USER 101

#define OP\_SYSTEM\_ERROR 102

#define OP\_SYSTEM\_READY 103

#define OP\_SYSTEM\_START 104

#define OP\_SHOW\_IMAGE 105

#define OP\_PLAY\_RESULT 106

#define OP\_PLAY\_INIT 107

#define OP\_SYSTEM\_ORDER 108

#define OP\_CLIENT\_LIST 109

//ID박스, 로그인버튼, 채팅창, 내용입력창, 전송버튼 핸들

HWND hEditId, hButtonLogin, hEditChat, hEditInput, hButtonSend;

//이미지박스, 레디버튼, 시작버튼 핸들, 플레이어1, 플레이어2

HWND hImg[9], hButtonReady, hButtonStart, hP1, hP2, hButtonList;

//채팅용 버퍼, 일반 버퍼

char chat[BUFSIZE+1], buf[BUFSIZE+1];

//유저 ID, 타 유저 ID

char userId[21], aUserId[21];

//소켓

SOCKET sock;

//임계영역

CRITICAL\_SECTION cs;

//글자 수

int nLength;

//이미지 저장용 변수

HBITMAP hBit[3];

//게임 시작 플래그

bool startFlag=false;

//선택된 개체 확인 변수

bool sel\_Img[9]={false,};

//대화상자 프로시저 및 센드 전용 프로시저

BOOL CALLBACK DlgProc(HWND hDlg, UINT uMsg, WPARAM wParam, LPARAM lParam){

//반환값

int retval;

byte opCode = 0;

//?, X, O 이미지 저장

hBit[0] = (HBITMAP)LoadImage(NULL, "Q.bmp", IMAGE\_BITMAP, 50, 50, LR\_LOADFROMFILE);

hBit[1] = (HBITMAP)LoadImage(NULL, "O.bmp", IMAGE\_BITMAP, 50, 50, LR\_LOADFROMFILE);

hBit[2] = (HBITMAP)LoadImage(NULL, "X.bmp", IMAGE\_BITMAP, 50, 50, LR\_LOADFROMFILE);

switch(uMsg){

case WM\_INITDIALOG:

//핸들러 초기화

hEditId = GetDlgItem(hDlg, IDC\_EDIT\_ID);

hButtonLogin = GetDlgItem(hDlg, IDC\_BUTTON\_LOGIN);

hEditChat = GetDlgItem(hDlg, IDC\_EDIT\_CHAT);

hEditInput = GetDlgItem(hDlg, IDC\_EDIT\_INPUT);

hButtonSend = GetDlgItem(hDlg, IDC\_BUTTON\_SEND);

hImg[0] = GetDlgItem(hDlg, IDC\_PLT);

hImg[1] = GetDlgItem(hDlg, IDC\_PCT);

hImg[2] = GetDlgItem(hDlg, IDC\_PRT);

hImg[3] = GetDlgItem(hDlg, IDC\_PLM);

hImg[4] = GetDlgItem(hDlg, IDC\_PCM);

hImg[5] = GetDlgItem(hDlg, IDC\_PRM);

hImg[6] = GetDlgItem(hDlg, IDC\_PLB);

hImg[7] = GetDlgItem(hDlg, IDC\_PCB);

hImg[8] = GetDlgItem(hDlg, IDC\_PRB);

hButtonReady = GetDlgItem(hDlg, IDC\_BUTTONREADY);

hButtonStart = GetDlgItem(hDlg, IDC\_BUTTONSTART);

hP1 = GetDlgItem(hDlg, IDC\_STATIC\_P1);

hP2 = GetDlgItem(hDlg, IDC\_STATIC\_P2);

hButtonList = GetDlgItem(hDlg, IDC\_BUTTONLIST);

userId[0]=NULL;

//ID 길이 제한

SendMessage(hEditId, EM\_SETLIMITTEXT, 20, 0);

return TRUE;

case WM\_COMMAND:

ZeroMemory(&opCode, sizeof(byte));

ZeroMemory(chat, sizeof(chat));

switch(LOWORD(wParam)){

case IDC\_BUTTON\_LOGIN:

EnableWindow(hButtonLogin, FALSE);

GetDlgItemText(hDlg, IDC\_EDIT\_ID, userId, 20);

if(strlen(userId) == 0){

EnableWindow(hButtonLogin, TRUE);

}

else{

EnterCriticalSection(&cs);

opCode = OP\_LOGIN;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

EnableWindow(hEditId, FALSE);

LeaveCriticalSection(&cs);

}

return TRUE;

case IDC\_BUTTON\_SEND:

EnableWindow(hButtonSend, FALSE);

GetDlgItemText(hDlg, IDC\_EDIT\_INPUT, chat, BUFSIZE);

if(strlen(userId) == 0 || userId[0]==NULL);

else{

EnterCriticalSection(&cs);

opCode = OP\_CHAT;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, chat, BUFSIZE, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

LeaveCriticalSection(&cs);

}

EnableWindow(hButtonSend, TRUE);

return TRUE;

case IDC\_BUTTONREADY:

if(userId[0]==NULL);

else{

EnableWindow(hButtonReady, FALSE);

EnterCriticalSection(&cs);

opCode = OP\_READY;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

LeaveCriticalSection(&cs);

}

return TRUE;

case IDC\_BUTTONLIST:

if(userId[0]==NULL);

else{

EnterCriticalSection(&cs);

opCode = OP\_LIST;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

LeaveCriticalSection(&cs);

}

return TRUE;

case IDC\_PLT:

if(startFlag){

if(sel\_Img[0]==false){

opCode = OP\_SELECT;

byte num=1;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PCT:

if(startFlag){

if(sel\_Img[1]==false){

opCode = OP\_SELECT;

byte num=2;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PRT:

if(startFlag){

if(sel\_Img[2]==false){

opCode = OP\_SELECT;

byte num=3;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PLM:

if(startFlag){

if(sel\_Img[3]==false){

opCode = OP\_SELECT;

byte num=4;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PCM:

if(startFlag){

if(sel\_Img[4]==false){

opCode = OP\_SELECT;

byte num=5;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PRM:

if(startFlag){

if(sel\_Img[5]==false){

opCode = OP\_SELECT;

byte num=6;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PLB:

if(startFlag){

if(sel\_Img[6]==false){

opCode = OP\_SELECT;

byte num=7;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PCB:

if(startFlag){

if(sel\_Img[7]==false){

opCode = OP\_SELECT;

byte num=8;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDC\_PRB:

if(startFlag){

if(sel\_Img[8]==false){

opCode = OP\_SELECT;

byte num=9;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, userId, 20, 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

retval = send(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return FALSE;

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 이미 선택된 항목입니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

}

else{

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 게임이 준비되지 않았습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

return TRUE;

case IDCANCEL:

EndDialog(hDlg, 0);

return TRUE;

}

return FALSE;

}

return FALSE;

}

//리시브 전용 스레드

DWORD WINAPI RecvMain(LPVOID arg){

int retval, cnt=0;

char result=0;

byte opCode=0;

char orderId[20];

byte sendCode=0;

WSADATA wsa;

if(WSAStartup(MAKEWORD(2,2), &wsa)!=0) ExitThread(-1);

sock = socket(AF\_INET, SOCK\_STREAM, 0);

if(sock==INVALID\_SOCKET) ExitThread(-1);

SOCKADDR\_IN serverAddr;

ZeroMemory(&serverAddr, sizeof(serverAddr));

serverAddr.sin\_family=AF\_INET;

serverAddr.sin\_port=htons(9721);

serverAddr.sin\_addr.s\_addr=inet\_addr("127.0.0.1");

retval = connect(sock, (SOCKADDR \*)&serverAddr, sizeof(serverAddr));

if(retval==SOCKET\_ERROR){

closesocket(sock);

ExitThread(-1);

}

while(1){

ZeroMemory(&opCode, sizeof(byte));

ZeroMemory(aUserId, sizeof(aUserId));

ZeroMemory(chat, sizeof(chat));

ZeroMemory(buf, sizeof(buf));

retval = recv(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR) continue;

else if(retval==0) break;

switch(opCode){

//로그인

case OP\_SYSTEM\_LOGIN:

char connectId[20];

EnterCriticalSection(&cs);

retval = recv(sock, connectId, 20, 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] ");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)connectId);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"님이 접속하였습니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

LeaveCriticalSection(&cs);

break;

//채팅 전용

case OP\_SYSTEM\_USER:

EnterCriticalSection(&cs);

retval = recv(sock, aUserId, 20, 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)aUserId);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)": ");

retval = recv(sock, chat, BUFSIZE, 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)chat);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

SetWindowText(hEditInput, "");

LeaveCriticalSection(&cs);

break;

//에러 처리

case OP\_SYSTEM\_ERROR:

EnterCriticalSection(&cs);

retval = recv(sock, buf, BUFSIZE, 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[ERROR] ");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)buf);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

LeaveCriticalSection(&cs);

break;

//준비 코드

case OP\_SYSTEM\_READY:

EnterCriticalSection(&cs);

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] 준비완료!");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

for(int k=0;k<9;k++){

SendMessage(hImg[k], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[0]);

}

SetWindowText(hP1, userId);

LeaveCriticalSection(&cs);

break;

//게임 시작 코드

case OP\_SYSTEM\_START:

EnterCriticalSection(&cs);

sendCode = OP\_ORDER;

retval = recv(sock, aUserId, 20, 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

SetWindowText(hP2, aUserId);

startFlag=true;

retval = send(sock, (char \*)&sendCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR) break;

LeaveCriticalSection(&cs);

break;

//이미지 변경 및 출력 코드

case OP\_SHOW\_IMAGE:

EnterCriticalSection(&cs);

bool oxFlag;

byte num;

retval = recv(sock, (char \*)&oxFlag, sizeof(bool), 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

retval = recv(sock, (char \*)&num, sizeof(byte), 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

if(oxFlag){

SendMessage(hImg[num-1], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[1]);

}

else{

SendMessage(hImg[num-1], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[2]);

}

sel\_Img[num-1]=true;

LeaveCriticalSection(&cs);

break;

//결과 출력

case OP\_PLAY\_RESULT:

EnterCriticalSection(&cs);

retval = recv(sock, (char \*)&result, sizeof(char), 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

nLength = GetWindowTextLength(hEditChat);

switch(result){

case 'w':

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[RESULT] Win!");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

break;

case 'l':

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[RESULT] Lose...");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

break;

case 'd':

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[RESULT] Draw");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

break;

default:

break;

}

LeaveCriticalSection(&cs);

break;

//게임에 사용된 값 초기화

case OP\_PLAY\_INIT:

EnterCriticalSection(&cs);

Sleep(2000);

for(int i=0;i<9;i++){

sel\_Img[i]=false;

SendMessage(hImg[i], STM\_SETIMAGE, IMAGE\_BITMAP, (LPARAM)hBit[0]);

}

SetWindowText(hP1, "Player1");

SetWindowText(hP2, "Player2");

startFlag=false;

opCode = OP\_INIT;

retval = send(sock, (char \*)&opCode, sizeof(byte), 0);

if(retval==SOCKET\_ERROR){

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[CSYSTEM] 전송 오류");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

return -1;

}

EnableWindow(hButtonReady, true);

LeaveCriticalSection(&cs);

break;

//순서 출력

case OP\_SYSTEM\_ORDER:

EnterCriticalSection(&cs);

ZeroMemory(orderId, sizeof(orderId));

retval = recv(sock, orderId, 20, 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] ");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)orderId);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"님부터 시작합니다.");

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

LeaveCriticalSection(&cs);

break;

//접속자 현황 확인

case OP\_CLIENT\_LIST:

EnterCriticalSection(&cs);

retval = recv(sock, (char \*)&cnt, sizeof(int), 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] 현재 접속자 \r\n");

for(int z=0;z<cnt;z++){

char tempId[20];

retval = recv(sock, tempId, 20, 0);

if(retval==SOCKET\_ERROR) break;

else if(retval==0){

closesocket(sock);

ExitThread(-1);

}

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)tempId);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"\r\n");

}

LeaveCriticalSection(&cs);

break;

//나머지 코드

default:

nLength = GetWindowTextLength(hEditChat);

SendMessage(hEditChat, EM\_SETSEL, nLength, nLength);

SendMessage(hEditChat, EM\_REPLACESEL, (WPARAM)TRUE, (LPARAM)"[SYSTEM] 잘못된 명령이 왔습니다.\r\n");

break;

}

}

closesocket(sock);

return 0;

}

int APIENTRY WinMain(HINSTANCE hInst, HINSTANCE,

LPSTR lpCmdLine, int nCmdShow)

{

//임계영역 지정 이벤트

InitializeCriticalSection(&cs);

//메세지 받는 스레드

DWORD ThreadId;

HANDLE hThread;

hThread = CreateThread(NULL, 0, RecvMain, NULL, 0, &ThreadId);

if(hThread==NULL) return -1;

//대화상자 생성

DialogBox(hInst, MAKEINTRESOURCE(IDD\_DIALOG1), NULL, DlgProc);

//소켓 종료

DeleteCriticalSection(&cs);

TerminateThread(hThread, 0);

CloseHandle(hThread);

closesocket(sock);

WSACleanup();

return 0;

}

4. 실행방법

1) 먼저 서버를 실행합니다.  
2) 서버에서 Loading Success라는 문구가 뜨면 클라이언트를 실행합니다.  
3) 클라이언트에서 ID옆에 있는 텍스트창에 ID를 입력합니다.(ID는 최대 20자)  
4) 다음 옆에있는 Login 버튼을 누릅니다.  
5) 밑에 입력할 수 없는 창에서 로그인 확인 메시지가 출력됩니다.(메시지는 최대 512자)  
6) 좌측 하단 입력창에서 채팅메세지를 입력합니다.  
7) 전송을 누르면 접속한 모든 사람에게 메시지가 전송됩니다.  
8) ready버튼을 누르면 게임 준비를 합니다.  
9) 서버에서 2명의 ready버튼 입력이 확인되면 게임을 시작하고 선을 보여줍니다.  
10) 게임을 합니다(게임방법은 버튼을 누르면 순서에 따라 O or X로 바뀌며 한 줄 완성 시 승리합니다)  
11) 게임이 끝나면 결과를 출력하고 클라이언트를 초기화 합니다.  
12) 현재 접속자를 확인하는 방법은 List버튼을 누르면 출력됩니다.  
13) 최대 접속할 수 있는 사람은 3명입니다.