

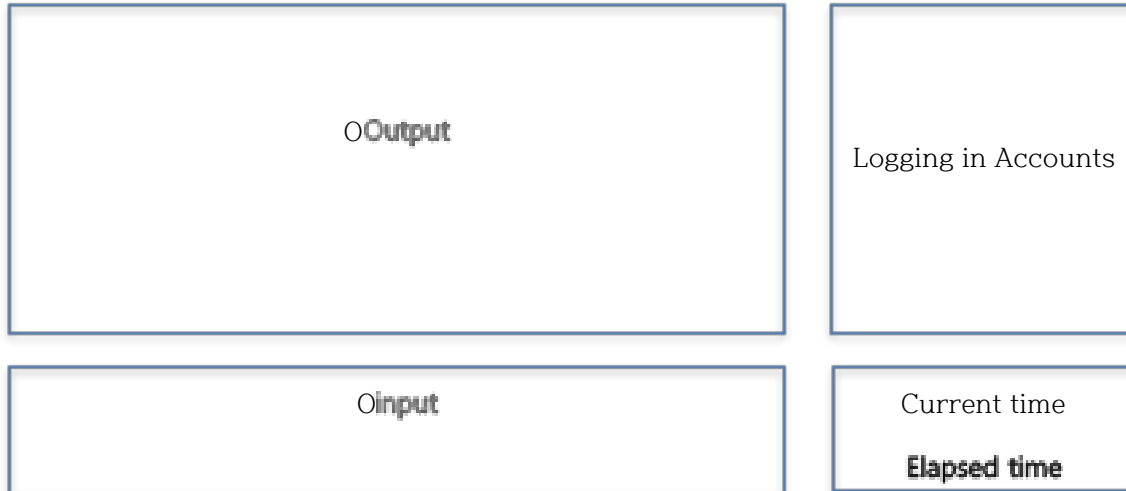
Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

Design your chatting app GUI including basic input and output functions.

The goal of homework 3 is to design and implement the chatting app with basic input and out functions.



[Guide line]

No need to use fork(), thread synchronization, and etc.

Modify "chat.c" I showed this week as the followings:

[1] Use ncurses to divide the main screen into 4 sub-windows

[2] Each sub-windows has box line.

- Terminal size is 80 X 24
- Output window size is 60 X 20
- Input window size is 60 X 4
- "Logging in Accounts" window size is 20 X 20
- "Time" window size is 20 X 4

[3] Display the current time and elapsed time (time will be refreshed in every 500ms)

[4] When a person logs in, his/her account will be appeared in the "Logging in Accounts" window.

When he/she type "/bye", his/her account will be disappeared in the "Logging in Accounts" window.

(Question: do we need another shared memory?)

[5] Try to do your best to apply your output produced in Homework#2 to this homework!!

[6] How to test your app:

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

- Open the first terminal and run chat Jico to write text messages (Actually, you don't need to create a new user account "Jico". It is enough to use the terminal where you logged in)
- Open the second terminal and run chat Izzy to write text messages
- Open the third terminal and run chat GD to write text messages
- Chat sequences are randomly and one person by one person. We do not consider that three persons chat at the same time. (We need threads. However, we do not need thread synchronization, fork, and etc for this work.)

=====

1. Put your program source as here (Do not put the screen shot of your source code!! If you insist, you will get zero point)

1) makefile

CC=gcc

CFLAGS=-g -Wall

all :chat

chat: chat.o

\$(CC) \$(CFLAGS) -o \$@ \$^ -lncurses -lpthread

chat.o: chat.c

clean:

rm -f *.o

rm -f chat

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

2) chat.c

```
/*
 * chat.c
 * Hw#3 Make Simple Chat Ncurses
 * Lee Jae Yoon(이재윤), 201624548
 * 유닉스 응용 프로그래밍(CP33357-059)
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>
#include <ncurses.h>
#include <sys/ipc.h>
#include <sys/shm.h>

#define BUFFSIZE 1024
#define MAXACCOUNT 20
#define NAMESIZE 20

typedef struct message_buffer {
    char name[NAMESIZE];
    char msg[BUFFSIZE];
    char account[MAXACCOUNT][NAMESIZE];
    int id;
} MSG_BUFF;

MSG_BUFF * msg_buff;

int is_running;
int account_cnt;

void *print_chat();
void *get_input();
void *recv_send();
void chat();
void cleanup();
void die(char * msg);
void setbox();
void refreshing();
void *show_time();
```

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

```
void *show_account();

int shmid;
char userID[NAMESIZE];
void * shmaddr = (void*) 0;

WINDOW *terminal_scr;
WINDOW *output_scr;
WINDOW *input_scr;
WINDOW *account_scr;
WINDOW *time_scr;

int main(int argc, char* argv[])
{

    if (argc < 2) {
        fprintf(stderr, "[Usage]: ./chat UserID Wn");
        exit(-1);
    }

    memset(userID, 0, NAMESIZE);
    strcpy(userID, (const char*)argv[1]);

    shmid = shmget((key_t)20200406, sizeof(MSG_BUFF), 0666|IPC_CREAT);
    shmaddr = shmat(shmid, (void *)0, 0666);
    msg_buff = (MSG_BUFF *) shmaddr;

    for (int i = 0; i < MAXACCOUNT; i++) {
        if (msg_buff->account[i][0] == 'W0') {
            strcpy(msg_buff->account[i], (const char*)argv[1]);
            account_cnt = i;
            break;
        }
    }

    initscr();
    chat();
    endwin();
    return 0;
}

void chat()
{
    curs_set(0);
```

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

```
scrollok(input_scr, TRUE);
terminal_scr = newwin(24, 80, 0, 0);
output_scr = subwin(terminal_scr, 20, 60, 0, 0);
input_scr = subwin(terminal_scr, 4, 60, 20, 0);
account_scr = subwin(terminal_scr, 20, 20, 0, 60);
time_scr = subwin(terminal_scr, 4, 20, 20, 60);
mvwhtline(output_scr, 1, 1, 68, 0);
wprintw(output_scr, "      ***** Type /exit to quit !! ***** \n\n");
setbox();
refreshing();
```

```
is_running = 1;
```

```
pthread_t thread[4];
pthread_create(&thread[0], NULL, get_input, NULL);
pthread_create(&thread[1], NULL, recv_send, NULL);
pthread_create(&thread[2], NULL, show_account, NULL);
pthread_create(&thread[3], NULL, show_time, NULL);
```

```
pthread_join(thread[0], NULL);
pthread_join(thread[1], NULL);
pthread_join(thread[2], NULL);
pthread_join(thread[3], NULL);
```

```
delwin(terminal_scr);
delwin(output_scr);
delwin(input_scr);
delwin(account_scr);
delwin(time_scr);
```

```
}
```

```
void *show_account()
```

```
{
    int cnt;
    while (is_running) {
        cnt = 1;
        werase(account_scr);
        for (int i = 0; i < MAXACCOUNT; i++) {
            if (msg_buff->account[i][0] != '\0') {
                mvwprintw(account_scr, cnt++, 2, "%s", msg_buff->account[i]);
            }
        }
        box(account_scr, ACS_VLINE, ACS_HLINE);
    }
}
```

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

```
wrefresh(account_scr);
usleep(300);

}

}

void *show_time()
{
    time_t start,now;
    start = time(0);
    struct tm ts;
    char buf[80];
    while(is_running){
        //시간 출력
        time(&now);
        ts = *localtime(&now);
        strftime(buf, sizeof(buf), "%H-%M-%S",&ts);
        mvwprintw(time_scr, 1,2,"%s",buf);

        now = time(0);
        time_t tmp = now-start;
        ts = *localtime(&tmp);
        // Match with our time
        ts.tm_hour -= 9;
        strftime(buf, sizeof(buf), "%H-%M-%S",&ts);
        mvwprintw(time_scr, 2,2,"%s",buf);
        box(time_scr, ACS_VLINE, ACS_HLINE);
        wrefresh(time_scr);
        usleep(500);
    }
}

void *get_input()
{
    char tmp[BUFFSIZE];
    int send_id = 1;
    while (is_running) {
        mvwgetstr(input_scr, 1, 1, tmp);
        sprintf(msg_buff->msg, " %s\n", tmp);
        strcpy(msg_buff->name, userID);
        msg_buff->id = send_id;

        if (strcmp (msg_buff->msg, " /exit\n") == 0 ) {
            die("exit");
        }
    }
}
```

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

```
        wprintw(output_scr, " [Send : %d] > %s", send_id++, msg_buff->msg);
        werase(input_scr);
        setbox();
        refreshing();
        usleep(100);
    }
}

void setbox() {
    box(output_scr, ACS_VLINE, ACS_HLINE);
    box(input_scr, ACS_VLINE, ACS_HLINE);
    box(account_scr, ACS_VLINE, ACS_HLINE);
    box(time_scr, ACS_VLINE, ACS_HLINE);
    wrefresh(input_scr);
}

void refreshing() {
    wrefresh(output_scr);
    wrefresh(input_scr);
    wrefresh(account_scr);
    wrefresh(time_scr);
}

void *recv_send()
{
    MSG_BUFF bef_msg;
    strcpy(bef_msg.msg, msg_buff->msg);
    strcpy(bef_msg.name, msg_buff->name);
    bef_msg.id = msg_buff->id;
    int myrecv = 1;
    while (is_running) {
        if (strcmp(msg_buff->msg, ".exit\n") == 0) {
            fprintf(stderr, "Chat is closed\n");
            is_running = 0;
        }
        else {
            if ( ( strcmp(msg_buff->msg, bef_msg.msg) != 0 ||
                    strcmp(msg_buff->name, bef_msg.name) != 0 ||
                    bef_msg.id != msg_buff->id ) &&
                strcmp(msg_buff->name, userID) != 0 ) {
                if (strcmp(msg_buff->msg, ".exit\n") == 0) {
                    wprintw(output_scr, " [Exit %s] Goodbye %s !\n",
                            msg_buff->name, msg_buff->name);
                    strcpy(bef_msg.msg, msg_buff->msg);
                }
            }
        }
    }
}
```

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

```
        else {
            wprintw(output_scr, " [Recv #%%d By %s] > %s",
                myrecv++, msg_buff->name, msg_buff->msg);
        }
        bef_msg.id = msg_buff->id;
        strcpy(bef_msg.msg, msg_buff->msg);
        strcpy(bef_msg.name, msg_buff->name);
        box(output_scr, ACS_VLINE, ACS_HLINE);
        wrefresh(output_scr);
        wrefresh(input_scr);
    }
}

void cleanup()
{
    memset(msg_buff->account[account_cnt] , 0, NAMESIZE);
    delwin(terminal_scr);
    delwin(output_scr);
    delwin(input_scr);
    delwin(account_scr);
    delwin(time_scr);
    endwin();
    shmdt(shmaddr);
}

void die(char * msg)
{
    cleanup();
    perror(msg);
    exit(-1);
}
```


Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

3) chat.c 에 대한 idea 설명

// 이름과 메시지, Account 와 메시지 id 를 공유하는 버퍼

```
typedef struct message_buffer {  
    char name[NAMESIZE];  
    char msg[BUFSIZE];  
    char account[MAXACCOUNT][NAMESIZE];  
    int id;  
} MSG_BUFFER;
```

메시지, 이름, 메시지 id 와 계정을 공유하는 구조체를 선언하여, 계정 리스트 출력과 중복 방지를 하였다.

계정 리스트 출력은 공유메모리를 제외한 파일입출력 시스템을 고려해 보았으나, 공유메모리를 사용하는 김에 공유메모리를 사용하였다.

// recv_send 에 사용되는 함수인데 메시지 내용이 이전과 다르거나, 보낸이의 이름이 다르거나, 메세지 아이디가 다른 상태에서

// 보낸 이가 내가 아니라면 출력을 하는 시스템이다. 고민하는데 꽤나 고생했다.

```
if ( ( strcmp(msg_buff->msg, bef_msg.msg) != 0 ||  
      strcmp(msg_buff->name, bef_msg.name) != 0 ||  
      bef_msg.id != msg_buff->id ) &&  
      strcmp(msg_buff->name,userID) != 0 )
```

// 내 아이디의 위치와, 메모리 정리를 위한 account number 을 선언

```
int account_cnt;
```

4) chat.c 에 포함된 함수 설명

/*

// MAXACCOUNT 만큼 루프를 돌며, NULL 이 아닌 객체를 출력.

// 모든 프로세스는 시작 시 msg_buff->account[account_cnt]에 자기 아이디를 복사

*/

```
void *show_account();
```

// 시간 출력 HW #1 을 기용하였음. usleep 을 이용해 0.5 초 쉬는 시간을 주었음.

```
void *show_time();
```

/*

// 입력받아 버퍼에 저장한다.

// id 에는 send_id 를 저장하는데, 항상 자기를 기준으로 하므로, 자신이 보낸 메시지는 id 가 겹칠일이 절대 없다.

*/

```
void *get_input();
```

/*

// 받은 메시지 버퍼의 내용을 출력한다.

// myrecv 는 내가 받은 내용의 카운트를 출력한다. 나를 기준으로 출력해야 하기 때문이다.

// idea 설명에서 말했던 if 문을 사용하여 출력하므로, 똑같은 내용을 출력하거나, 받은 내용을 빼 먹을 일이 없다.

*/

```
void *recv_send()
```

Homework #3 (Double Points)

School ID: 201624548

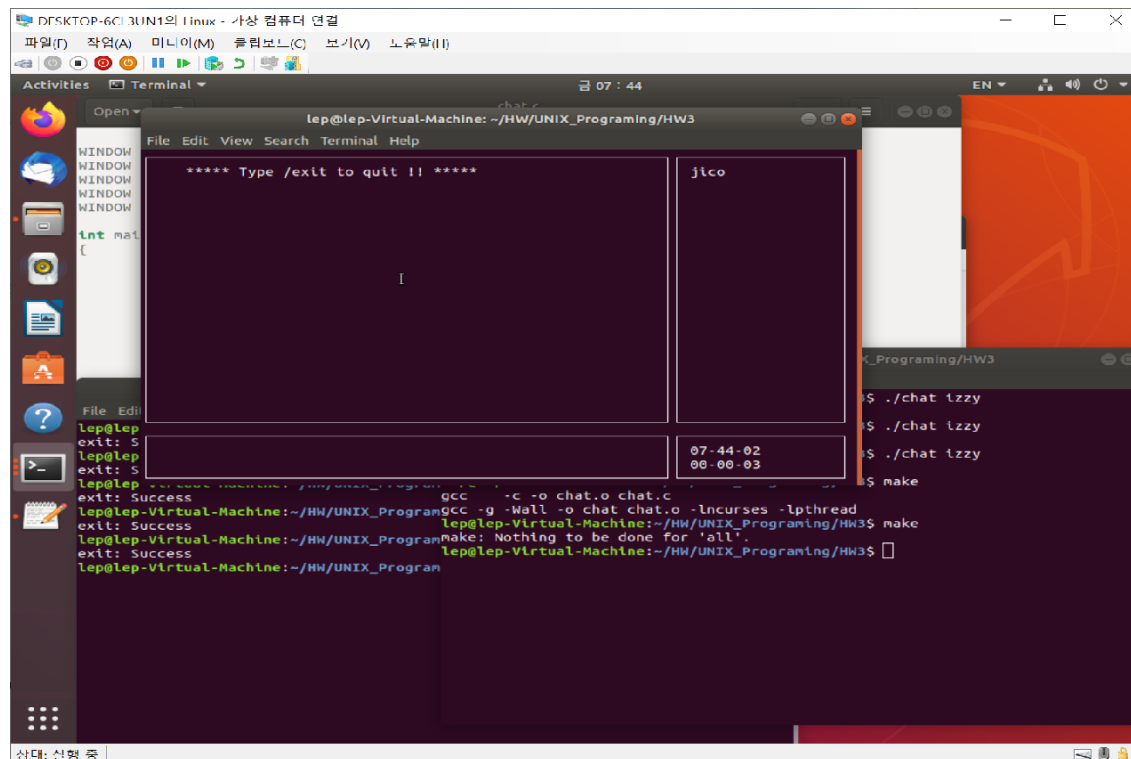
Name: 이재윤

2. You must show the building result after compiling and linking your source codes. You must show no warnings and errors (Use gcc -Wall option).

(Put a screen shot of your C debugging output)

```
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW3$ ls
Makefile chat.c chatd.c
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW3$ make
gcc -c -o chat.o chat.c
gcc -g -Wall -o chat chat.o -lcurses -lpthread
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW3$ ls
Makefile chat chat.c chat.o chatd.c
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW3$
```

3. Put a screen shot of output generated by your program. Your output screen shot must be readable for me to verify your chat program. (쓰레드 동기화를 이용하지 말라고 하여, 쓰레기 값이 보일 수 있습니다.)



Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤

The image displays two screenshots of a Linux desktop environment, likely Ubuntu, showing a network programming exercise. The desktop has a sidebar with various application icons and a top panel with system status and time (07:44).

Top Screenshot:

- The terminal window shows a chat program running. The prompt is `lept@lept-Virtual-Machine: ~/HW/UNIX_Programing/HW3`.
- The program output shows: `**** Type /exit to quit !! ****`, `[Send : 1] > hi`, `[Send : 2] > hello`, and `[Recv #1 By lizzy] > good morning`.
- The terminal also shows the command `lnt mal` and the output `[`.
- A LibreOffice Writer window is open in the background.

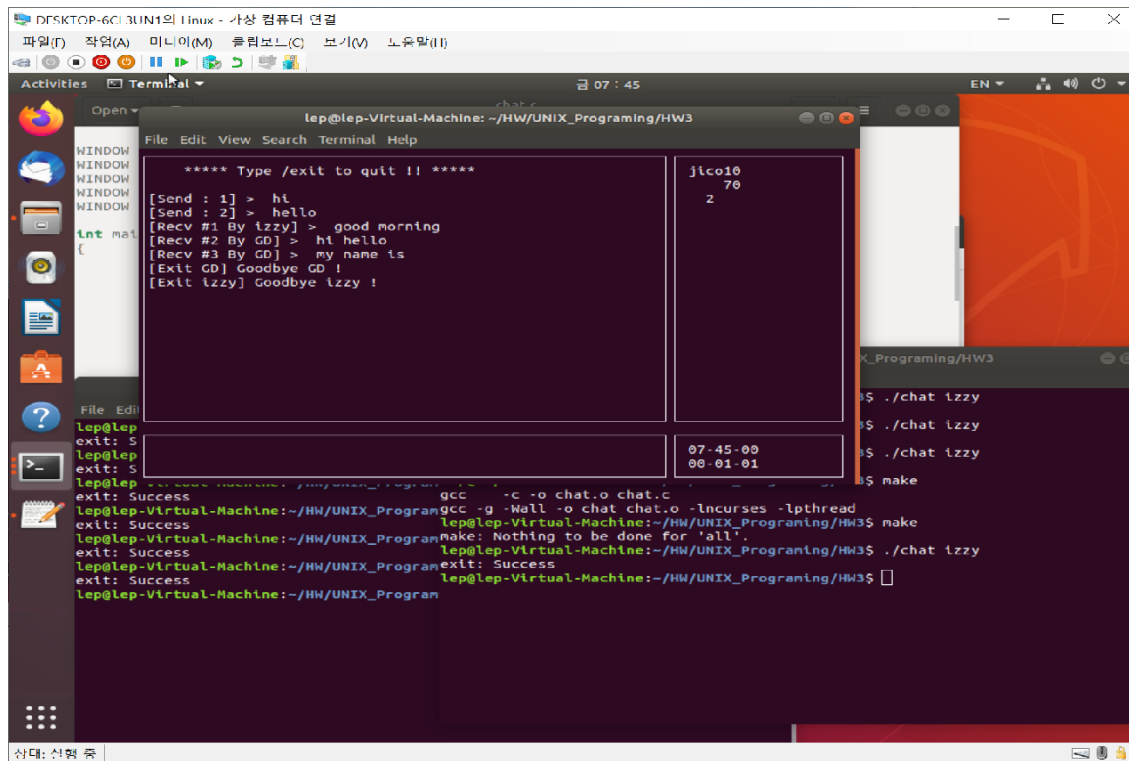
Bottom Screenshot:

- The terminal window shows the same chat program running. The prompt is `lept@lept-Virtual-Machine: ~/HW/UNIX_Programing/HW3`.
- The program output shows: `**** Type /exit to quit !! ****`, `[Send : 1] > hi`, `[Send : 2] > hello`, `[Recv #1 By lizzy] > good morning`, `[Recv #2 By jico] > hi hello`, and `[Recv #3 By GD] > my name is`.
- The terminal also shows the command `lnt mal` and the output `[`.
- A LibreOffice Writer window is open in the background.

Homework #3 (Double Points)

School ID: 201624548

Name: 이재윤



```
DESKTOP-6C13UN1의 Linux - 가상 컴퓨터 연결
파일(F) 작업(A) 미니미(M) 플러그인(C) 보기(V) 도움말(H)
Activities Terminal 07:45 EN
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3
File Edit View Search Terminal Help
**** Type /exit to quit !! ****
[Send : 1] > hi
[Send : 2] > hello
[Recv #1 By izzy] > good morning
[Recv #2 By GD] > hi hello
[Recv #3 By GD] > my name is
[Exit GD] Goodbye GD !
[Exit izzy] Goodbye izzy !
jico10
70
2
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ ./chat izzy
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ ./chat izzy
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ ./chat izzy
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ make
gcc -c -o chat.o chat.c
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ gcc -o chat chat.o -lncurses -lpthread
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ make
make: Nothing to be done for 'all'.
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ ./chat izzy
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$ exit
lep@lep-Virtual-Machine: ~/HW/UNIX_Programing/HW3$
```