

## Homework #2

School ID: 201624548

Name: 이재윤

**[1] Read Unix Homework Coding Style Guidelines carefully. You must follow these guidelines for your homework. Otherwise, you will get zero points**

**[2] Test scenario**

- 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 do not need threads, synchronization, fork, and etc for this work.)

**[3] The "chat.c" program behaves like the following steps:**

- [Constraints]
  - Shared memory key is 20200406 (already existed, just attach)
  - Define the shared memory data structure that supports multi-user chat.
  - Each message has a unique ID and so messages will not be duplicated.
  - Your messages buffered in the shared memory will also be displayed (echoed...)
  - We don't need to scroll up and down text messages
- [Procedures]
  - Create a shared memory. Attach the shared memory to the chat process (If the shared memory has already existed, just attach the shared memory to the chat process)

## Homework #2

School ID: 201624548

Name: 이재윤

- After you write some messages, you can see display the chat messages in the shared memory (If you want to display new messages arrived at the shared memory, you just write “..” will not be sent to the shared memory)
- Type “/bye” to quit the chat process.
- Run the chatremove (you need to implement your own chatremove.c)

[3] Your program consists of at least two files and one your own header.

You must build your own Makefile.

=====

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 chatremove

chat: chat.o

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

chatremove: chatremove.o

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

chat.o: chatInfo.h chat.h chat.c

chatremove.o: chatInfo.h chat.h chat.c

clean:

rm -f \*.o

rm -f chat

rm -f chatremove

## 2) chat.c

```
/*
 * chat.c
 * Hw#2 Make simple chat program
 * Lee Jae Yoon(이재윤), 201624548
 * 유닉스 응용 프로그래밍(CP33357-059)
 */

#include "chat.h"
#include "chatInfo.h"

int main(int argc, char* argv[])
{
    //define variables
    int shmid;
    char userID[id_length];
    CHAT_INFO * chatInfo = NULL;
    void * shmaddr = (void*) 0;

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

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

    shmid = shmget((key_t)20200406, sizeof(CHAT_INFO), 0666|IPC_CREAT);

    if (shmid < 0) {
        perror("shmget failed: ");
        exit(-1);
    }

    // Attach Share memory
    shmaddr = shmat(shmid, (void *)0, 0666);
```

```

chatInfo = (CHAT_INFO *) shmaddr;

chatInfo->messageTime = 0;

printf("Chat Start#\n");

while (true){

    char inputstr[message_length];

    fgets(inputstr,message_length, stdin);

    // replace escape letter

    rstrip(inputstr, message_length);

    // loading Chatting

    if (strcmp(inputstr, "..") == 0) {

        printf("[%s] #%ld : %s\n",chatInfo->userID, chatInfo->messageTime, chatInfo->message);

    }

    // Exit chat

    else if (strcmp(inputstr, "./bye") == 0) {

        printf("Chat Program Exit !!\n");

        printf("GoodBye %s\n", userID);

        break;

    }

    // Save chat to share memory

    else {

        strcpy(chatInfo->userID, userID);

        chatInfo->messageTime++;

        strcpy(chatInfo->message, inputstr);

    }

}

// detach Share memory

if (shmdt(shmaddr) == -1)

{

    printf("Failed to detach Share Memory !!");

    exit(-1);

}

return 0;

}

```

### 3) chat.h

```
/*
 * chat.h
 * Hw#2 Make simple chat program
 * Lee Jae Yoon(이재윤), 201624548
 * 유닉스 응용 프로그래밍(CP33357-059)
 * include files, define lengths
 */

#ifndef __CHAT_H__
#define __CHAT_H__

#include <stdio.h>

// To use strcmp, strcpy
#include <string.h>

// To use exit
#include <stdlib.h>

// To use Share Memory
#include <sys/ipc.h>
#include <sys/shm.h>

// define lengths
#define id_length 20
#define message_length 40

//define true
#define true 1
#define false 0
```

```
// \n 개행문자 제거함수
```

```
void rstrip(char * arr, int length) {  
    for (int i = 0; i < length; i++)  
    {  
        if (arr[i] == '\n') {  
            arr[i] = '\0';  
            break;  
        }  
    }  
}
```

```
#endif // CHAT_H
```

#### 4) chatInfo.h

```
/*
 * chatInfo.h
 *
 * Hw#2 Make simple chat program
 *
 * Lee Jae Yoon(이재윤), 201624548
 *
 * 유닉스 응용 프로그래밍(CP33357-059)
 *
 * Struct About chatInfo
 */
#ifndef __CHAT_SHM_H__
#define __CHAT_SHM_H__
#include "chat.h"
typedef struct chatInfo {
    char userID[id_length];
    long messageTime;
    char message[message_length];
} CHAT_INFO;
#endif //__CHAT_SHM_H__
```

5) chatremove.c

```
/*
 * chatInfo.h
 * Hw#2 Make simple chat program
 * Lee Jae Yoon(이재윤), 201624548
 * 유닉스 응용 프로그래밍(CP33357-059)
 */

#include "chat.h"
#include "chatInfo.h"

int main(void)
{
    int shmid;

    shmid = shmget((key_t)20200406, sizeof(CHAT_INFO), 0666|IPC_CREAT);

    if (shmid < 0) {
        perror("shmget failed: ");
        exit(-1);
    }

    // share memory 삭제
    if (shmctl(shmid, IPC_RMID, 0) < 0) {
        printf("Failed to delete shared memory\n");
        exit(-1);
    }

    printf("Chat Remove Success!!\n");

    return 0;
}
```

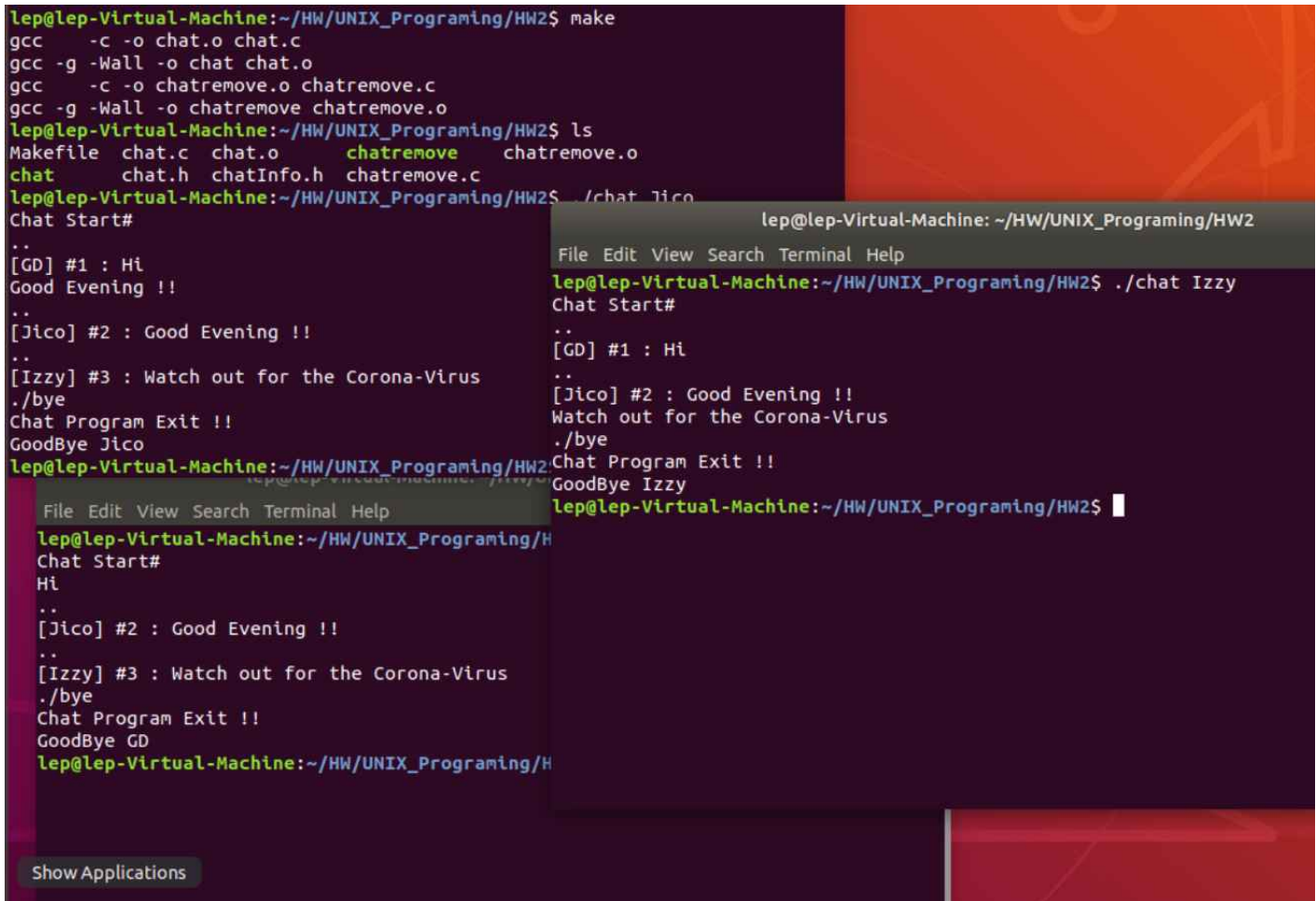


2. You must show the building result after compiling and linking your source codes. You must show no warnings and errors (Use gcc -Wall option).

(Put a screen shot of your C debugging output)

```
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ make clean
rm -f *.o
rm -f chat
rm -f chatremove
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ make
gcc -c -o chat.o chat.c
gcc -g -Wall -o chat chat.o
gcc -c -o chatremove.o chatremove.c
gcc -g -Wall -o chatremove chatremove.o
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ ls
Makefile  chat.c  chat.o  chatremove  chatremove.o
chat      chat.h  chatInfo.h  chatremove.c
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$
```

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.



```
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ make
gcc -c -o chat.o chat.c
gcc -g -Wall -o chat chat.o
gcc -c -o chatremove.o chatremove.c
gcc -g -Wall -o chatremove chatremove.o
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ ls
Makefile chat.c chat.o chatremove chatremove.o
chat chat.h chatInfo.h chatremove.c
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ ./chat Jico
Chat Start#
..
[GD] #1 : Hi
Good Evening !!
..
[Jico] #2 : Good Evening !!
..
[Izzy] #3 : Watch out for the Corona-Virus
./bye
Chat Program Exit !!
GoodBye Jico
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ ./chat Izzy
Chat Start#
..
[GD] #1 : Hi
..
[Jico] #2 : Good Evening !!
Watch out for the Corona-Virus
./bye
Chat Program Exit !!
GoodBye Izzy
lep@lep-Virtual-Machine:~/HW/UNIX_Programing/HW2$ ./chatremove
Chat Remove Success!!
```