**Homework #1**

**정보컴퓨터공학부**

**201424423**

**김영진**

**1. Put your program source as here**

**<chatshm.h>**

#ifndef \_\_CHAT\_SHM\_H\_\_

#define \_\_CHAT\_SHM\_H\_\_

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/ipc.h>

#include <sys/shm.h>

#include <unistd.h>

#include <time.h>

// Declare struct for saving chat information

typedef struct checkInfo {

    char userID[20];

    char messageTime[10];

    char message[40];

    int messageID; // I add messageID variable to distinguish each message

} CHAT\_INFO;

// Define for checking current time

time\_t now;

struct tm t;

// Define function that print chat message

void printchat(CHAT\_INFO \*chatInfo, int i);

#endif // \_\_CHAT\_SHM\_H\_\_

**<chat.c>**

#include "chatshm.h"

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

    int shmID; // Define for saving shared memory ID

    char userID[20]; // Define for saving user ID

    // Declare instance of CHAT\_INFO struct and allocate memory 10 times of checkInfo struct

    CHAT\_INFO \*chatInfo = (CHAT\_INFO \*)malloc(sizeof(struct checkInfo) \* 10);

    // Declare shared memory address for 0 and type casting int to void\*

    void \*shmaddr = (void \*) 0;

    // If user doesn't type userID, messenger print error message

    if(argc < 2){

        fprintf(stderr, "[Usage]: ./chat UserID \n");

        exit(-1);

    }

    // Save user ID in userID variable

    strcpy(userID, argv[1]);

    // Create 1 shared memory, its key is 20200406

    shmID = shmget((key\_t)20200406, sizeof(CHAT\_INFO), 0666|IPC\_CREAT|IPC\_EXCL);

    // If there already exists shared memory which has key 20200406, then Dont' create sharedmemory

    // and just reference shared memory ID

    if(shmID < 0){

        shmID = shmget((key\_t)20200406, sizeof(CHAT\_INFO), 0666);

    }

    // attach shared memory into process

    shmaddr = shmat(shmID, (void \*)0, 0000);

    // type casting from void\* to CHAT\_INFO\*

    chatInfo = (CHAT\_INFO \*) shmaddr;

    printf("-------------- Chat Message -------------\n");

    while(1){

        // Define variables for saving message and current time

        char inputstr[40];

        char time\_now[50];

        // get input message from user and save in inputstr

        printf("Enter your message : ");

        fgets(inputstr, 40, stdin);

        // get current time

        time(&now);

        t = \*localtime(&now);

        // Change form to what we want

        strftime(time\_now, sizeof(time\_now), "%H:%M:%S", &t);

        // if user type "..", print all message except user's own message

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

            int i = 0;

            for(i=0; i<chatInfo->messageID; i++){

                if(strcmp(chatInfo[i].userID, userID) != 0){

                    printchat(chatInfo, i);

                }

            }

        }

        // if user type /bye, break loop

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

            break;

        }

        // If user type any message, save each message in chatInfo shared memory

        else{

            strcpy(chatInfo[chatInfo->messageID].userID, userID);

            strcpy(chatInfo[chatInfo->messageID].messageTime, time\_now);

            strcpy(chatInfo[chatInfo->messageID].message, inputstr);

            // Print user's own message echoed

            printchat(chatInfo, chatInfo->messageID);

            // Increase messageID for distinguish each message

            chatInfo->messageID++;

        }

    }

    return 0;

}

**<printchat.c>**

#include "chatshm.h"

// Print userID, messageTime, message

void printchat(CHAT\_INFO \*chatInfo, int i){

    printf("[%s] %s : %s\n", chatInfo[i].userID, chatInfo[i].messageTime, chatInfo[i].message);

}

**<shmremove.c>**

#include "chatshm.h"

int main() {

    // Define for saving shared memory Id

    int shmID;

    // Get shared memory ID which has key 20200406

    shmID = shmget((key\_t)20200406, sizeof(CHAT\_INFO), 0666);

    // If there's no shared memory ID which has key 20200406, then print error message

    if(shmID < 0){

        perror ("shmget failed : ");

        exit(-1);

    }

    // Delete shared memory

    // If there's no shared memory to delete, print error message

    if(shmctl(shmID, IPC\_RMID, 0) < 0) {

        printf("Failed to delete shared memory\n");

        exit(-1);

    }

    // If deleting is finished successfully, print success message

    else {

        printf("Successfully delete shared memory\n");

    }

    return 0;

}

**<Makefile>**

CC = gcc

CFLAGS = -g -Wall

TARGET = messenger

OBJECTS = chat.o printchat.o

all : $(TARGET)

$(TARGET): $(OBJECTS)

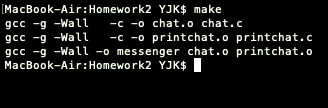
        $(CC) $(CFLAGS) -o $@ $^

clean :

        rm \*.o messenger

        ./shmremove

**2. You must show the building result after compiling and linking your source codes. You must show no warnings and errors.**



**3. Put a screen shot of output generated by your program as well as the Ubuntu system’s clock to verify if your clock is working correctly or not**

