

Homework #1

School ID: 201624548

Name: 이재윤

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

[2] Use ncurses and threads (at least one thread). Design a digital clock that has three subwindows: Each subwindow has its own border (ACS_HLINE, ACS_VLINE)

The first subwindow displays the current local date (i.e, Korean date). The format looks like yyyy-mm-dd-day (Googling keywords: C Epoch Converter Routines)

The second subwindow displays the current local time (i.e, Korean time). The format looks like hh-mm-ss

The third subwindow display the elapsed time after your clock process executed. The format looks like hh-mm-ss

[3] Your program consists of at least three files and one your own header. You must build your own Makefile.

[4] The clock GUI style is up you

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 source

```
CC=gcc
```

```
CFLAGS=-g -Wall
```

```
OBJS=HW1_main.o HW1_func.o
```

```
TARGET=a.out
```

```
all : $(TARGET)
```

```
$(TARGET): $(OBJS)
```

```
$(CC) -o $@ $^ -lncurses -lpthread
```

```
HW1_main.o: HW1_header.h HW1_main.c
```

```
HW1_func.o: HW1_header.h HW1_func.c
```

```
clean:
```

```
rm -f *.o
```

```
rm -f $(TARGET)
```

2) HW1_header.h

```
/*
 * HwHeader.h
 * Hw#1 Make Clock With Threads and Ncurses
 * Lee Jae Yoon(이재윤), 201624548
 * 유닉스 응용 프로그래밍(CP33357-059)
 */
#include <stdio.h>
#include <time.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>
#include <ncurses.h>
// 스레드 동기화를 위한 뮤텍스 전역변수 선언
pthread_mutex_t mutex_lock;
// 첫번째 subwindow에 날짜를 출력
void* showDate(void * arg);
// 두번째 subwindow에 시간을 출력
void* showTime(void * arg);
// 세번째 subwindow에 시작 후 경과한 시간을 출력
void* showTimer(void * arg);
```

3) HW1_main

```
#include "HW1_header.h"

int main(void)
{
    //declare variable
    pthread_t date_thread,time_thread,timer_thread;

    WINDOW *dateWnd, *timeWnd,*timerWnd;

    //ncurses default setting end

    initscr();

    noecho();

    dateWnd = subwin(stdscr, 5, 50, 0, 0);
    timeWnd = subwin(stdscr, 5, 50, 5, 0);
    timerWnd = subwin(stdscr, 5, 50, 10, 0);

    curs_set(0);

    wrefresh(dateWnd);
    wrefresh(timeWnd);
    wrefresh(timerWnd);

    //ncurses default setting end

    //setting mutex
    pthread_mutex_init(&mutex_lock, NULL);

    // create date_thread
    if ( pthread_create(&date_thread,NULL,showDate,(void*)dateWnd) < 0){
        printf("cannot create t1");
        sleep(1);
        return 1;
    }

    // create time_thread
    if ( pthread_create(&time_thread,NULL,showTime,(void*)timeWnd) < 0){
        printf("cannot create time_thread");
        sleep(1);
        return 1;
    }

    // create timer_thread
    if ( pthread_create(&timer_thread,NULL,showTimer,(void*)timerWnd) < 0){
```

```
printw("cannot create timer_thread");

sleep(1);

return 1;

}

// exit whie "q"
while (TRUE) {

int input = getch();

if ( input == 'q' ) break;

}

// 쓰레드 종료

pthread_cancel(date_thread);

pthread_cancel(time_thread);

pthread_cancel(timer_thread);

// ncurses 종료

endwin();

return 0;

}
```

4) HW1_func.c

// 함수 정의

```
#include "HW1_header.h"
```

```
/*
```

```
* 첫번째 subwindow에 날짜를 출력 (subwindow는 파라미터로 받아옴)
```

```
* 생성된 윈도우를 받아 2,2 지점부터 날짜를 출력
```

```
* 쓰레드 동기화를 위해 뮤텍스 락, 언락
```

```
*/
```

```
void* showDate(void * arg)
```

```
{    WINDOW * TargetWnd = (WINDOW *) arg;
```

```
    time_t now;
```

```
    struct tm ts;
```

```
    char buf[80];
```

```
    while(TRUE){
```

```
        pthread_mutex_lock(&mutex_lock);
```

```
        time(&now);
```

```
        ts = *localtime(&now);
```

```
        strftime(buf, sizeof(buf), "%Y-%m-%d-%a",&ts);
```

```
        mvwprintw(TargetWnd, 2,2,"%s",buf);
```

```
        box(TargetWnd, ACS_VLINE, ACS_HLINE);
```

```
        wrefresh(TargetWnd);
```

```
        pthread_mutex_unlock(&mutex_lock);
```

```
    }
```

```
}
```

```
/*
```

```
* 두번째 subwindow에 시간을 출력 (subwindow는 파라미터로 받아옴)
```

```
* 생성된 윈도우를 받아 2,2 지점부터 시간을 출력
```

```
* 쓰레드 동기화를 위해 뮤텍스 락, 언락
```

```
*/
```

```
void* showTime(void * arg)
```

```
{    WINDOW * TargetWnd = (WINDOW *) arg;
```

```
    time_t now;
```

```
    time(&now);
```

```
    struct tm ts;
```

```

char buf[80];

while(TRUE){

pthread_mutex_lock(&mutex_lock);

time(&now);

ts = *localtime(&now);

strftime(buf, sizeof(buf), "%H-%M-%S",&ts);

mvwprintw(TargetWnd, 2,2,"%s",buf);

box(TargetWnd, ACS_VLINE, ACS_HLINE);

wrefresh(TargetWnd);

pthread_mutex_unlock(&mutex_lock);

}

}

/*
* 세번째 subwindow에 시작 후 경과한 시간을 출력 (subwindow는 파라미터로 받아옴)
* 생성된 윈도우를 받아 2,2 지점부터 타이머를 출력
* 쓰레드 동기화를 위해 뮤텍스 락, 언락
*/

void* showTimer(void * arg)
{
    WINDOW * TargetWnd = (WINDOW *) arg;

    time_t start,now;

    start = time(0);

    struct tm ts;

    char buf[80];

    while(TRUE){

pthread_mutex_lock(&mutex_lock);

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(TargetWnd, 2,2,"%s",buf);

box(TargetWnd, ACS_VLINE, ACS_HLINE);

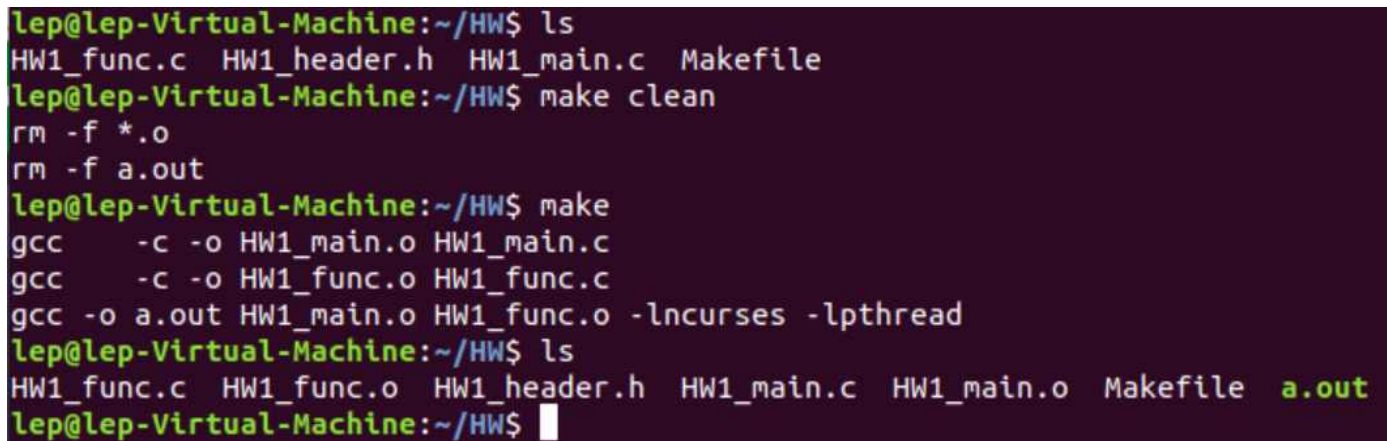
wrefresh(TargetWnd);

```

```
pthread_mutex_unlock(&mutex_lock);  
  
}  
  
}
```

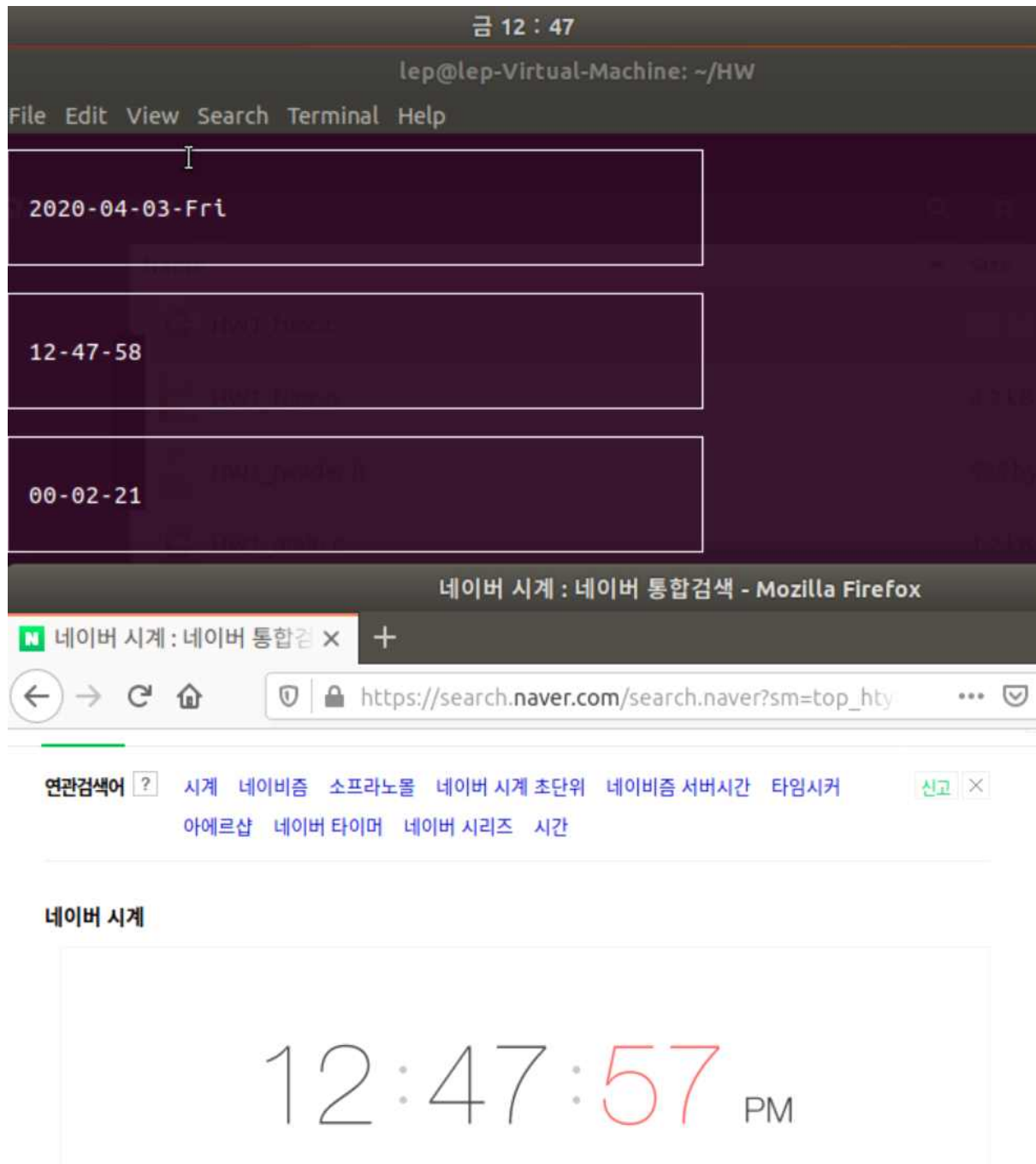
2. You must show the building result after compiling and linking your source codes. You must show no warnings and errors.

(Put a screen shot of your CPP debugging output)



```
lep@lep-Virtual-Machine:~/HW$ ls  
HW1_func.c HW1_header.h HW1_main.c Makefile  
lep@lep-Virtual-Machine:~/HW$ make clean  
rm -f *.o  
rm -f a.out  
lep@lep-Virtual-Machine:~/HW$ make  
gcc -c -o HW1_main.o HW1_main.c  
gcc -c -o HW1_func.o HW1_func.c  
gcc -o a.out HW1_main.o HW1_func.o -lcurses -lpthread  
lep@lep-Virtual-Machine:~/HW$ ls  
HW1_func.c HW1_func.o HW1_header.h HW1_main.c HW1_main.o Makefile a.out  
lep@lep-Virtual-Machine:~/HW$
```


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.



네이버 시계와 완벽히 동기화 되지 않았지만, 컴퓨터 시계와는 동기화 된 것을 알 수 있다.