作業系統期末專案報告  
第14組

103703013 資科三 黃育萱

103703015 資科三 蔡雨芝

103702012 心理三 陳冠聞

目錄

1. 報告實作主題、內容簡介
2. Thread類型與功能介紹
3. Multi-Thread之間的同步與合作
4. 關鍵區域程式碼
5. GUI呈現與附加功能
6. Github與版本管理、分工
7. 組員心得
8. 參考資料

一、報告實作主題、內容簡介

主題：The Sleeping Barber (SB) Problem

語言：C ++ 11

內容：The barber shop has m barbers with m barber chairs, and n chairs (m < n) for waiting customers, if any, to sit in. If there are no customers present, a barber sits down in a barber chair and falls asleep. When a customer arrives, he has to wake up a sleeping barber.

If additional customers arrive while all barbers are cutting customers’ hair, they either sit down (if there are empty chairs) or leave the shop (if all chairs are full). The thread synchronization problem is to program the barbers and the customers without

getting into race conditions.

二、Thread類型與功能介紹

1. barberThread：  
   每個barber各自擁有的、獨立的thread，用來控管barber的狀態(sleeping or being busy)。
2. customerThread：  
   用來產生客人的thread

三、Multi-Thread之間的同步與合作

四、關鍵區域程式碼

void \*barberThread(void\* arg) {

int \*pID = (int\*)arg;

while(1) {

barMutex.lock();

cusMutex.lock();

if(totalServedCustomers < realNum\_customer){

cusMutex.unlock();

customers.wait(); // Try to acquire a customer.

//Go to sleep if no customers

Mutex.lock(); // Acquire access to waiting

//When a barber is waken -> wants to modify # of available chairs

barbers.signal(); // The barber is now ready to cut hair

int nowCut = nextCut;

nextCut = (nextCut+1) % NUM\_CHAIRS;

availableChairs++;

Mutex.unlock(); // Release waiting

/\* GUI change barber's mode \*/

isBusy[\*pID-1] = true;

cutting[\*pID-1] = waitingChairs[nowCut].data->cusID;

isSit[waitingChairs[nowCut].seqNumber] = false;

glutPostRedisplay(); //////////////GUI

cutHair(\*pID, waitingChairs[nowCut]); //pick the customer which counter point

isBusy[\*pID-1] = false;

glutPostRedisplay(); //////////////GUI

}

else{

barMutex.unlock();

cusMutex.unlock();

break;

}

}

}

關鍵區域程式碼(cont.)

void \*customerThread(void\* arg) {

struct customerData \*data = (struct customerData\*)arg;

Mutex.lock(); // Acquire access to waiting

if( availableChairs == 0 ) {

comeCus[data->cusID-1] = false;

cusMutex.lock();

--realNum\_customer;

cusMutex.unlock();

Mutex.unlock();

pthread\_exit(0);

}

comeCus[data->cusID-1] = false;

seat[nextSit] = data->cusID;

waitingChairs[nextSit].data = data;

nextSit = (nextSit+1) % NUM\_CHAIRS;

availableChairs--;

showWhoSitOnChair();

glutPostRedisplay(); //////////////GUI

customers.signal(); // Wake up a barber (if needed)

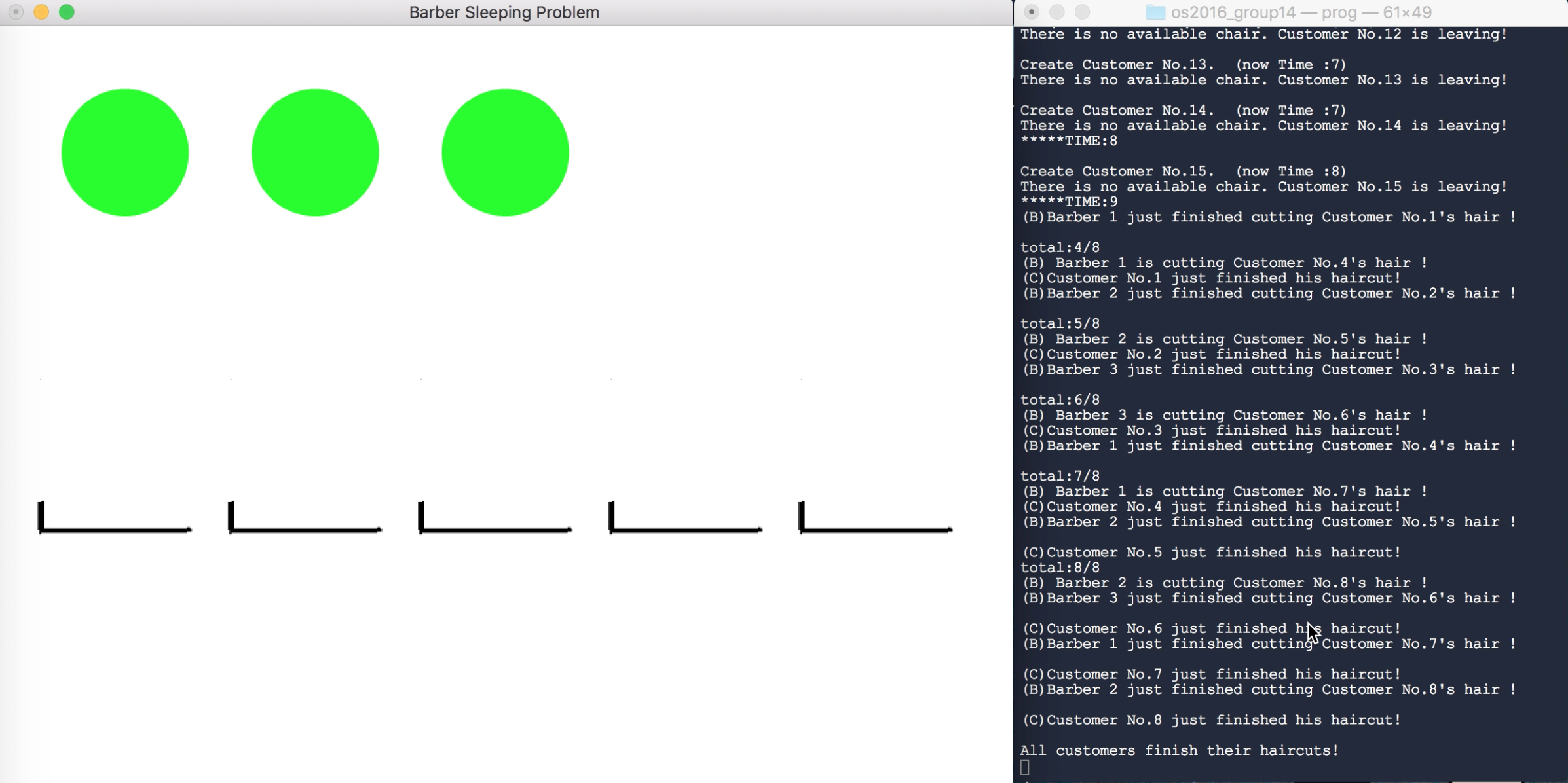
Mutex.unlock(); // Release waiting

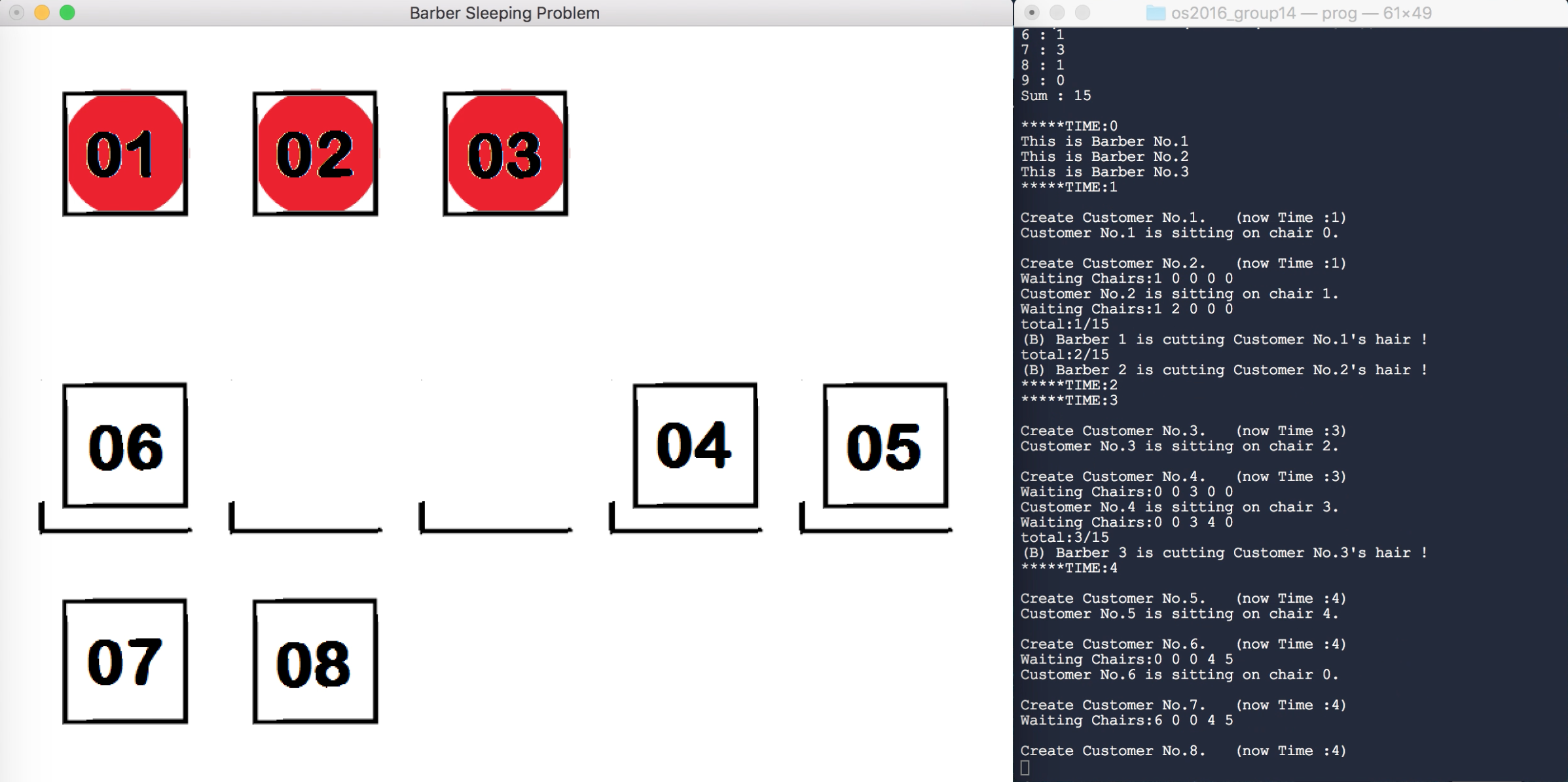
barbers.wait(); // Go to sleep if number of available barbers is 0

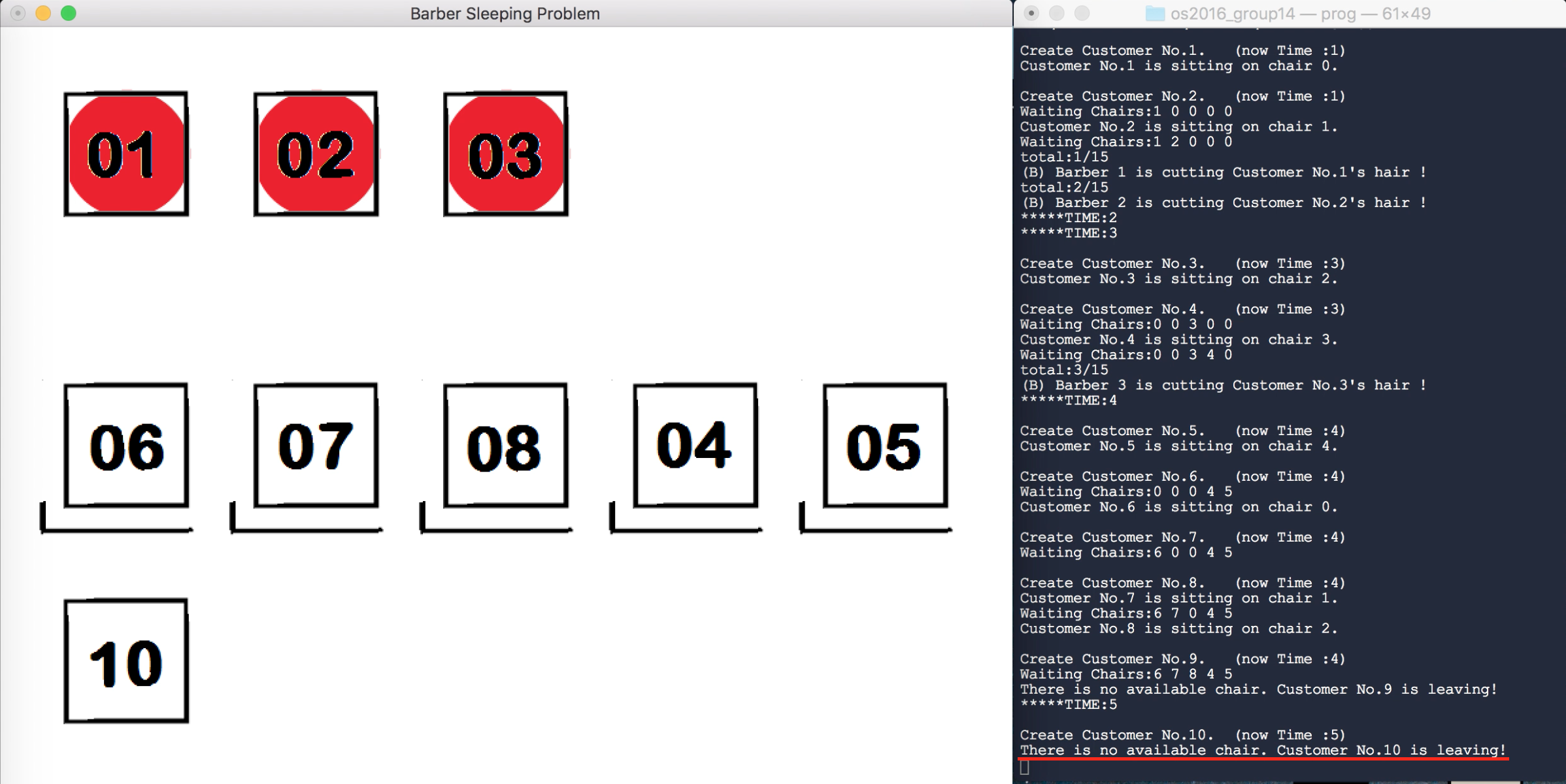
waitForHairCut(data);

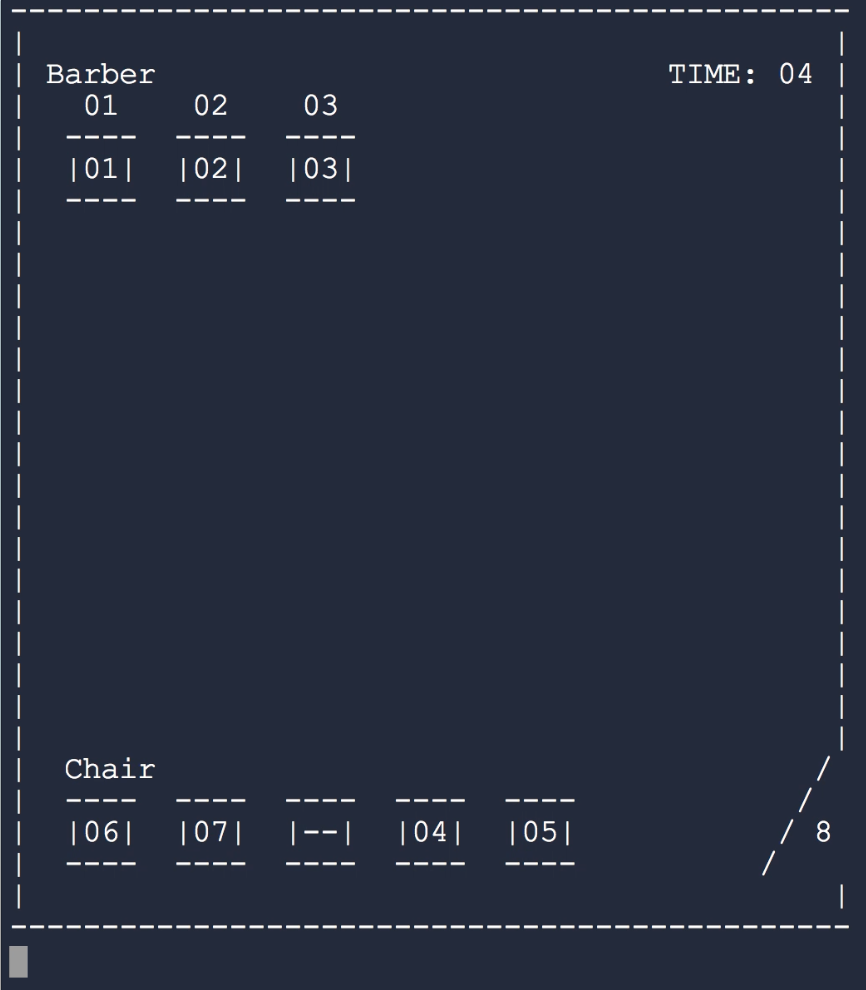
}

五、GUI呈現與附加功能

# 初始/結束畫面（附圖為結束）

# 客人進入理髮店等待

# 若客人進入理髮店時無位置則離開

# 附加功能：文字UI顯示

六、Github與版本管理、分工

# Version 1：<pthread.h>(適用Windows/ Ubuntu(Mac OS有編譯問題))

[SB01~SB10.cpp]  
虛擬碼撰寫：陳冠聞  
c++程式碼撰寫、優化：陳冠聞、蔡雨芝

# Version 2：Windows -> Mac OS(解決Mac OS編譯問題)

[SB11.cpp] 程式碼轉換：黃育萱

# Version 3：程式碼優化、轉換為適用於Windows版本

[SB12、SB13win.cpp] 程式碼優化：陳冠聞、蔡雨芝

# Version 4：GUI (using Qt)

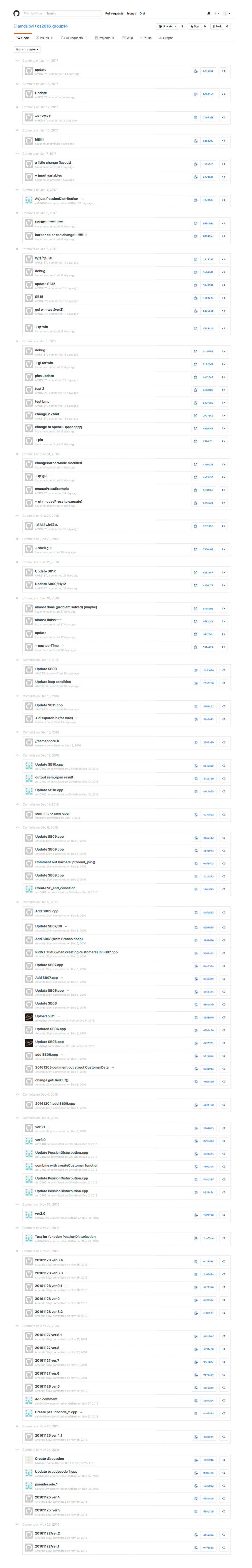
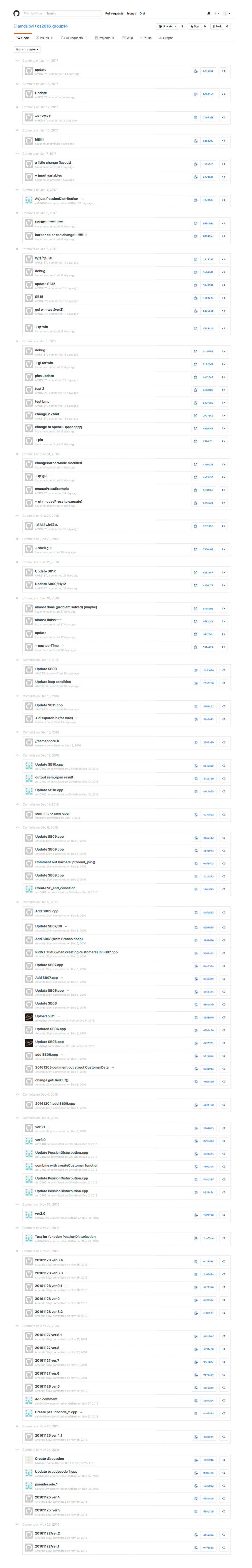
[SB13.cpp] GUI程式碼：黃育萱

# Version 5：<thread> & <mutex>(適用於Windows/Mac OS/Ubuntu)

[SB15.cpp] 程式碼轉換：蔡雨芝

# Version 6 (Final)：GUI (using OpenGL)

[SB16.cpp] GUI程式碼：黃育萱



七、組員心得

# 103703013 資科三 黃育萱：

我負責的工作主要是將有文字輸出的資料圖形化，在組員剛寫完初步文字介面程式的版本，遇到的第一個問題就是作業系統的差別而完全不能執行（主要是因為iOS系統不能接受無名的Semaphore而會有初始問題），所以找了很多方法嘗試，大概經過跟組員完成主程式程式碼的時間差不多。在終於能在電腦裡執行之後，當初先利用QT Creator原本預想在短時間內就能完成，卻因為不能先顯示gui介面再開始執行程式，也試著用openGL實作，因為都不能成功，先做過了一個以符號表示的介面，後來又試著將pthread改成thread及mutex，最後經過跟同學討論中發現實作gui需要以函式呼叫才能正常同步執行（經測試後的QT也可以在multi-thread的環境執行gui）。我覺得從討論到實作其實常常會遇到各種不同的問題，有很多是在跨平台的執行結果不同，就要猜測及檢視程式碼，經過反覆的測試才能完成這個專案。

# 103703015 資科三 蔡雨芝：

# 103702012 心理三 陳冠聞：

八、參考資料

1. C++ Reference <http://www.cplusplus.com/reference/>

2. pthread程式撰寫 <http://blog.xuite.net/nikoung/wretch/154071878>

3.multi-thread同步 <http://blog.yam.com/swwuyam/article/11459124>

4.C++multi-thread程式開發 <https://kheresy.wordpress.com/2012/07/06/multi-thread-programming-in-c-thread-p1/>