```
// Kyle McCullough
// Group D
// kymcculk@okstate.edu
// 4/26/2022

#include "Robby.h"
#include "Corey.h"
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <semaphore.h>
#include <pthread.h>
#include <sys/types.h>
#include <time.h>

/**
 *   Prints out that medical professional # is waiting for a patient
 *   struct threadStruct *contents is a pointer to the contents struct holding Medical Professional
information.
 */
void waitForPatients(struct threadStruct *contents)
{
   printf("Medical Professional %d (Thread ID: %d): Waiting for patient \n", contents->id, contents-
>threadID);
}

/**
 *   Syncs up the performance of the medical checkup from a medical professional with the getting of the
medical check up of the patient.
```

```
 *   struct threadStruct *contents is a pointer to the contents struct holding Medical Professional information.
 */
void performMedicalCheckup(struct threadStruct *contents)
{
    pthread_mutex_lock(&mutex[4]);


    contents->bondId = buffer;
    buffer = contents->id;


    pthread_mutex_unlock(&mutex[5]);


    printf("Medical Professional %d (Thread ID: %d): Checking patient %d\n", contents->id, contents->threadID, contents->bondId);
}


/**
 *   Only allows one thread to accept payment from the patients. Prints out that the medical professional # accepts payment from patient #.
 *   struct threadStruct *contents is a pointer to the contents struct holding Medical Professional information.
 */
void acceptPayment(struct threadStruct *contents)
{
    pthread_mutex_lock(&mutex[7]);
    pthread_mutex_lock(&mutex[2]);
    printf("Medical Professional %d (Thread ID: %d): Accepted payment from Patient %d\n", contents->id, contents->threadID, contents->bondId);
    pthread_mutex_unlock(&mutex[6]);
    pthread_mutex_unlock(&mutex[7]);
}
```