KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

Information and Computer Science Department

ICS431: Operating System

Project Two (Term 072)

Due Date: May 24th 2008

Project Submission:

You are requested to submit **both** hardcopy and softcopy on the due date. Hardcopy should contain your source code only. Softcopy (source code only) should be submitted through WebCT.

Project Rules:

1. You are given 3 grace days throughout the semester to submit your projects. You can divide these days between project1 and project2 as you wish. So if you submit Project1 late by 1 day, you will have only 2 grace days left for Project2.
2. Any cheating in the project will result in ZERO mark.
3. THERE WILL BE NO DIFFERENTIATION BETWEEN THE PERSON WHO COPIES AND THE PERSON WHO LETS OTHERS COPY. Both will have the same penalty.

Project Question:

You are required to implement a restaurant system where customers walk in, place there order and are served with their orders. As there is no sitting room in the restaurant, the customers leave as soon as they get their orders. As customers walk into the restaurant they stand in a single line (queue).
There are many cashiers in your restaurant and as each cashier is finished with the current customer, he calls the customer in the front of the queue to place his order. Once the order is placed, the customer will wait for his order and then leave the cash register once the order is ready. At this time the cashier is free to call the next customer.

Your restaurant has a fixed size and because of city ordinance, you are not allowed to have more customers than your capacity. If the restaurant is full, any new customer will have to wait outside until a customer leaves the building.

Specifically you will have to program the following parts for your project

Customer Thread:

For this part, you will program a thread that creates Customer Objects. This thread will run continuously in a loop such that in each iteration of the loop, the thread should create a new customer with a unique id and a delay that specifies how long it will take for his order to be prepared which should be a random number between 3-7 seconds (Its a super fast food restaurant). The thread should place this customer in a queue.

During program execution, if the user presses CTRL-C key combination, **this thread** should stop generating new customers and quit. Your program should have only one of these threads.

Cashier Thread:

Your program will have several Cashier Threads. The number of these threads should be a command line argument. Each thread should remove one customer from the queue, take his order(this requires no code!), and then wait for the time it requires to prepare this customers food (The delay field in the customer, use sleep function to simulate waiting). After the food is served, this thread is ready for the next customer. If there are no customers available in the queue, this thread is going to wait for the next customer to walk in. Do NOT busy wait or poll the queue. You should use mutexes and conditional variables for this part.

The Queue:

The code for the queue is provided for you in files queue.h and queu.c. DO NOT modify this code unless you have permission from me or if a modification has been announced in the lab or on WebCT.
The following functions are provided in queue.c file.
void Queue_Init(queue* q, int Size)

This function is used to initialize the queue. The size variable is the capacity of the queue. Its your responsibility to make sure you do not add more customers than the queue capacity. The capacity of the queue should be read as a command line argument.

int Queue_Add(queue* q, Customer c);

Use this function to add a customer to the end of the queue.

Customer Queue_Remove(queue* q);

Use this function to remove one customer from the front of the queue

int Get_Count(queue* q);

Use this function to find out the number of customers in the queue. Please note that this function does not consider the capacity of the queue and if you add more customers than the capacity, this function will report a number that is higher than the capacity of the queue.

The Customer:

The structure for customer is provided in customer.h class. Please use the same struct and DO NOT modify this code. The struct has the following fields.

int id

This should be a unique id assigned to each customer.

int delay;

This is the time required to process this customers order. It should be a random number between 3 and 7.

int type;

This field is not used currently. Set it to anything your want.

Main Program:

Your main program should initialize all mutexes and conditional variables and should start all the threads. It should also initialize a global queue. Then the program should wait for the threads to finish their execution.

Program Termination:

Your program should continue execution until the user presses CTRL-C key combination. At this point the Customer thread should stop generating more customers but the Cashier Threads should continue processing those customers that are already in queue. Once all the customers have been processed, each thread will terminate.

Program Output:

Your program should output all the events to a file called “log.txt”.

1. When a customer is generated, you will log the following event to the file:
   ThreadID - Event: Customer “id” added to the queue. Delay is “d” seconds.
   Where id and d are the id and delay fields of the customer respectively and ThreadID is the thread id of the thread reporting this event.

2. When a Cashier thread finishes processing a Customer it should report the following event.

   ThreadID - Event: Customer “id” removed and processed. Currently n customers in the queue.
   Where n is the number of customers still in the queue that need to be processed.

3. When the user presses CTRL-C, the Customer Thread should report the following event.

   ThreadID - Event: Store shutting down. Currently n customers in the queue.

4. When the queue is empty and new customers are not being generated (User has already pressed CTRL-C) the Cashier Thread should report the following event and the terminate.

   ThreadID - Event: Restaurant closed. No customers waiting. Time to go home!

5. When all the threads have terminated, the main program should report the following event:

   Main Program Terminating! Customers in Queue are n.
   Where n is the number of customer in the queue. This number should be 0 if everything is working properly.

Extra Credit (5 points):

Design your program such that Customers are generated in a separate process. The main program will open the required pipes and then start the Customer process as a child process. The Customer process will generate a customer and write the attributes to the open pipe and then wait for the main program to read the customer before generating the next customer. You will create a thread in the main program that will read customers from the pipe, write the customer data to the queue and then send a message to the Customer process that customer has been read. The communication between the thread and Customer process has to be done on two separate pipes.