COMP 304- Operating Systems: Assignment 2

Due: April 12, midnight

Notes: This is an individual assignment. No late assignment will be accepted. Please submit your answers through blackboard. This assignment is worth 10% of your total grade.

Corresponding TA: Aditya Sasongko (msasongko17@ku.edu.tr)

Problem 1

(20 points) Consider the following set of processes, with the length of the CPU-burst time given in milliseconds:

Process	Burst Time	Priority
P1	8	3
P2	5	1
P3	3	3
P4	1	4
P5	10	2

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at time 0.

a) Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, a non- preemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 4 ms) scheduling.

b) Calculate the waiting time of each process for each of the scheduling algorithms in part (a). Which of the schedules results in the minimal average waiting time?

c) Calculate average turnaround time for each of the scheduling algorithms in part (a).

Assume no context-switch overhead.

Problem 2

(10 points) Now assume that the context-switching overhead is equivalent to 0.5 ms. Calculate the CPU utilization for all four scheduling algorithms in Problem 1.

```c
//PROBLEM 3

#define MAX_CONNECTIONS 5000
int available_connections = MAX_CONNECTIONS;

/* When a thread wishes to establish a connection with the server,
it invokes the connect() function:*/

int connect() {
   if (available_connections < 1)
        return -1;
   else
       available_connections --;
   return 0;
}

/* When a thread wishes to drop a connection with the server,
it invokes disconnect() */
int disconnect() {
   available_connections ++;
   return 0;
}
```

Problem 3

(15 points) Consider the code example above that creates threads for opening connections at a server.
a) Identify the race condition(s) in the provided program.
b) If you have identified any race conditions, use pthread locks to prevent the race condition(s). Provide the code snippet.
c) Could we replace the integer variable available_connections with atomic integer, which allows atomic update of a variable of this type? Explain your answer.
atomic_t available_connections = MAX_CONNECTIONS;

Problem 4

(15 points) A hotel management would like to use online reservation system for their rooms. The hotel has M rooms available for online reservation. If all the rooms are occupied, the hotel will not accept any more customers until a room becomes available again (a party leaves the hotel).

Give an algorithm for the hotel using semaphore primitives to limit the number of reserved rooms so that it does not exceed M. Note that a room can accommodate up to 4 customers. If a party more than 4 customers arrives, multiple rooms may be needed. If the required number of rooms is not available, the room request for the entire party will be declined. Provide a pseudo-code and briefly explain your algorithm in a short paragraph.

Problem 5

(25 points) Consider a hospital that offers a free check-up service to senior people (65 years old or older) on Sundays. M many senior citizens $(n_1, n_2, ..., n_m)$ are expected to visit the hospital for this Sunday. Write a monitor (in pseudocode) that allocates 4 equally professional doctors to these people, using the priority numbers of patients for deciding the order of their doctor visit. The older the person, the higher her/his priority is.

When a patient arrives to the hospital, s/he should call *void request_doctor(int priority)* function. If all doctors are busy, the patient is made to wait. When a patient is done with her/his visit, the patient should call *release_doctor()* function. You can assume that there is a *sort()* function, which takes an integer array and sorts it.

Problem 6

(15 points) Consider the following snapshot of a system, where A, B, C and D are resources and P1-P5 are processes.

	Allocation	Max	Available
	A B C D	A B C D	A B C D
P1	0 0 1 4	0 6 5 6	1 5 2 0
P2	0 6 3 2	0 6 5 2	
P3	1 3 5 4	2 3 5 6	
P4	0 0 1 2	0 0 1 2	
P5	1 0 0 0	1 7 5 0	

Using the bankers algorithm for deadlock avoidance:

a) What is the content of the matrix Need?

b) Is the system in a safe state? Explain your answer.

c) If a request from process P5 requests (0, 3, 3, 0), can the request be granted immediately? Explain the reason.