COMP 3323 OPERATING SYSTEMS

Assoc. Prof. Ahmet KOLTUKSUZ
TA's: Anas Maazu Kademi, Ph.D., Bora Güzel, M.Sc.

2020-2022 SPRING TERM

Lab Exam March 31st, 2022

Notes :

i. **Duration is for 2 hours.**
ii. **Answer only one question of your choice**
ii. **Collaboration by any means is strictly and positively prohibited.**
iii. **Return the answers as "name_surname.c" .**
iv. **Explain your answers where necessary with comments in your code.**

Q.1 Sleepy Dentist

Consider a dentist a hypothetical dentist's office. This office has one dentist, one dentist's chair, and N chairs for waiting patient, if any, to sit in. If there is no patient at present, the dentist sits down on the his/her chair and falls asleep. When a patient arrives, he/she has to wake up the sleepy dentist. If additional patients arrive while the dentist is in session with someone else, they either sit down (if there is an empty chair at the waiting room) or leave the office (if all chairs are full). (i) *Write a program to coordinate the dentist and the patients.*

Q.2 Round robin

` Round-robin schedulers generally has a list, with each process occurring exactly once. (i)*What would happen if a process occurred more than once?*

(ii)Write a program to implement a Round Robin algorithm. You May fill the BT and AT values by creating five different threads and returning random values from each. (iii) *Calculate CT, TAT, WT, average TAT values in your code.*

Q.3 Ding Dong

Consider a Ding Dong communication between devices allowing each computer to know if the opposite end is still present and available, where the Ding transmitted packet to a destination computer, and the Dong is the response.

Let Ding and Dong be two separate threads executing their respective procedures. The code below is intended to cause them to forever take turns, alternately printing "Ding" and "Dong" to the screen. The ***thread_stop()*** blocks the calling thread, and ***thread_start ()*** unblocks a specific thread, making it runnable, if that thread has previously been stopped/blocked.

```
void Ding ()
{
        while(true){
                thread_stop();
                Printf("Ding is calling\n");
                 Y = deliverthermessage(Y);
                thread_start(dongthread);
        }
}
```

```
void Dong ()
{
        while(true) {
                Printf("Dong is available and responding\n");
                Y = deliverthermessage(Y);
                thread_start(Dingthread);
                thread_stop();
        }
}
```

I. *Which synchronization flaw does the above code exhibit? When would this code fail (scenario), and what is the outcome of that scenario.*
II. *Show how to fix the problem by replacing the* **thread_stop** *and* ***thread_start*** *calls with semaphore Wait (P/down) and Signal (V/up) operations.*
III. *Implement Ding and Dong correctly using a mutex and condition variables.*

Hints

- **Completion Time**: The time when processes complete their execution.
- **Turn Around Time**: The time difference between the completion time (CT) and the arrival time (AT).
  Turn Around Time (TAT) = Completion Time (CT) - Arrival Time (AT)
- **Waiting Time**: The total time between requesting action and acquiring the resource.
  Waiting Time (WT) = Turn Around Time (TAT) - Burst Time (BT)