CS 4323 Design and Implementation of Operating Systems I

Final Group Project: Full Marks 110
(Due Date: 12/01/2020, 11:59 PM CST)

This is the final group project that needs to be done in the group as specified. If you do not know the group members, then please refer the module **FinalGroupProjectFormation** under **Home** page in Canvas.

Objectives:

1. Process scheduling
2. Deadlock detection and handling
3. Process synchronization and resource allocation

There is a very expensive fitness center in the town famous for its professional trainers and equipment. No customer can work without a fitness trainer. The gym has ***N*** number of trainers and one trainer provides fitness training to one customer at a time. If there is no customer, then the trainer gets busy on his/her phone. The fitness center has a big waiting room with ***M*** number of couches in case if no trainer is free in the fitness center. If the customer arrives to the fitness center and none of the trainer is available, then the customer goes to the waiting room on foot and takes one of the couches in the waiting room. If all the couches are occupied in the waiting room, then the customer leaves and does not wait any longer. If any trainer becomes available in the fitness center, then he/she goes to the waiting room on electric unicycle scooter and bring the customer based on who came first to the fitness center (i.e. provides service in the first-come-first served order). On the other hand, if the customer arrives in the fitness center and there is a trainer available and he/she is free (i.e. not providing service to the customer), then the trainer immediately becomes available to assist the customer.

The upper bound for the values of ***N*** and ***M*** that your program should support are 5 and 6 respectively while the minimum bound are 3 and 3 respectively.

Part 1: Synchronization of the customers using Semaphore

Now, there can be a deadlock and starvation under following scenarios:

1. Consider the customer arrives and finds all the trainer busy with the customer. In that case, the customer goes to the waiting room, which is empty at the moment. As soon as the customer leaves the fitness center, the trainer becomes available and he/she goes to the waiting room to get the customer. Since the trainer is on scooter and the customer is on foot, the trainer arrives the waiting room faster. As the trainer does not see any customer waiting in the waiting room, he/she returns to the fitness center and gets busy with his/her phone, thinking there is no customer waiting for the trainer in the waiting room. Meanwhile, when the customer arrives in the waiting room, he/she takes a couch and waits in the waiting room in the hope that the trainer will come and take him/her.

So, both the trainer and customer are unable to proceed thereby causing the deadlock scenario.

2. Consider all the couches in the waiting room are occupied except one. Now, two customers arrive to the fitness center and both find all trainers busy with their respective customers. So, both customers go to the waiting room and see one couch available. Both of them try to access it. This can causes the problem of having more customer than the number of couches in the waiting room.

There can be violation in the first-come-first-served policy as well.

3. Consider the customer arrives and finds all the trainer busy with the customer. In that case, the customer goes to the waiting room, which is empty at the moment. As soon as the customer leaves the fitness center, the trainer becomes available and he/she goes to the waiting room to get the customer. Since the trainer is on scooter and the customer is on foot, the trainer arrives the waiting room faster. As the trainer does not see any customer waiting in the waiting room, he/she returns to the fitness center and gets busy with his/her phone, thinking there is no customer waiting for the trainer in the waiting room. Now, if the second customer arrives, then he/she will be served instead of serving the first customer who is waiting in the waiting room. This will cause starvation to the customer waiting first.

In the first part of the project, your program needs to:

a) Show the occurrence of these three boundary cases (i.e. deadlock, starvation and unfair scheduling) as specified above.
b) Provide synchronization mechanism among customers using semaphores that avoids all these three boundary cases. Note that you need to redo the simulation to solve the problems in part (a)

Part 2: Deadlock detection and recovery

In this part, you will need to implement the deadlock detection algorithm and recover from the deadlock. The trainer will decide the number of sets (i.e. round of exercise to do). So, if the trainer decides 2, then for each set the customer needs to read from the file the following inputs:

- number of grip plates (i.e. weight plates) of different weight. For e.g.: 2.5 pounds, 5 pounds, 10 pounds, 15 pounds, 20 pounds, 25 pounds, 35 pounds, 45 pounds
- number of plates of each weight
- current plates being held by each customer
- request to each weight type that will be made by each customer

This has to be done for each of the ***N*** customers who are currently training in the fitness center. In the case of deadlock detected, the customer who has performed less number of sets would be chosen as victim. When the victim customer restarts the exercise, it should continue from where it left (i.e. employ partial rollback).

In this part of the project, your program needs to:

c) Show the occurrence of deadlock i.e. run the algorithm to detect the deadlock.
    - To the code of part (b), you need to implement this portion

d) Once the deadlock is detected, the program should recover from the deadlock. Note that you need to redo the simulation to solve the problems in part (c).

Part 3: Mutex for synchronization

Before each customer leaves the fitness center (or equivalently, once each customer completes the exercise), each trainer needs to enter the total weight used by his/her customer in the record book. So, for each customer, the trainer needs to maintain this record. If two or more customers complete the exercise at the same time, the two trainers cannot perform writing to the record book at the same time. They need to synchronize their action. Use mutex lock to provide this synchronization.

In this part of the project, your program needs to:

e) Use mutex to provide synchronization for writing the weights used by the customer in the record book.

Points distribution for this project will be as follows:

Part 1: Synchronization of the customers using Semaphore	
a) Testing boundary cases	[20 points]
b) Provide synchronization mechanism	[20 points]
Part 2: Deadlock detection and recovery	
c) Deadlock detection	[15 points]
d) Deadlock recovery	[15 points]
Part 3: Mutex for synchronization	[20 points]
Report and group coordination	[10 points]
Final Representation	[10 points]

The final report needs to include:

- The implementation of your .c file, including explanation and illustration of each part.
- The various test case should be clearly specified highlighting each portion of the project assignment.
- The source code should be in the modular form and should be commented. Each function should be properly described with its argument and the return type. This should be done in both the .c file as well as the report.

  Note: While copying the code from the .c file to the report, you need to make sure that it is readable. It should be properly formatted and organized.

- Task distribution to each group member need to be clearly discussed.
- Individual contribution (all completed and incomplete work) of each group members need to be clearly specified. In the case of incomplete work, each member need to separately show their test cases.
- Need to discuss the limitation and possible extension of the program.

Grading Criteria:

This is a group project. So, grading will focus on students' ability to work in the group and successfully complete the work. Each member in the group should coordinate as a team rather than individual. And that's the key to score maximum points.

Points to remember:

- Project progress report is a part of this project and will be graded together with the final project submission.
- Failure to complete the project as a group will deduct 20 point, even though every individual has completed their part. As this is a group project, you need to learn to work in the group and meet the team's objective, and not individual objective.
- Work must be distributed uniformly among all group member and should be clearly specified in the report.
    - Be sure to submit a written report that summarizes your design and all the test cases you have performed.
    - Include each student's contribution clearly. Failure to be specific on this part will make us difficult to give fair grades. So, it is your responsibility to clearly indicate what functions have been implemented by which member in the group.
    - Each student should complete theory part of the job first before taking responsibility of group member's work. Failure to complete the initially assigned work would be penalized even if the student have done other group member's work.
- It is not necessary that all group members will get equal points. It depends on what responsibility each student has taken, whether or not the student has delivered the task.
- It is the group's responsibility to alert instructor with any issue that is happening in the group.
    - Students are supposed to use the group created in the Canvas for all correspondence. That will serve as the proof, in case there is any dispute among the group members.
    - Time is very important. Group members are expected to respond quickly.
    - You need to have sufficient days to combine individual work. You are going to make one submission as a group and not individually.
- Each group has to present the work to the TA by 12/04/2020, 5:00 pm CDT. During the presentation, all the group members need to be present. The TA will ask questions to each group member about the implementation of their part.

Submission Guidelines:

- You should submit your programming assignment as three separate driver .c files:
  - 1st driver file will implement part (a).
  - 2nd driver file will implement part (b) and (c)
  - 3rd driver file will implement part (b), (d) and (e)
- Include the readMe.txt file that shows how to run your code.
- All the input files should be provided.
- Your code should include the header information, which should include:
  - Group Number
  - Group member name
  - Email
- There should be sufficient comments in the program.
  - Each function should be clearly described along with the input arguments and return values.
- Report is to be submitted as pdf and the last page of the report should include all the code. Codes are to be copy pasted from the .c file(s) and not to use screenshot.