CS 421 Spring 2021
Homework #3
Due Wednesday, April 14th.

Remember to show your work and explain your answers for all problems.

Chapter 5

1. Most versions of UNIX use a multi-level feedback queue scheme with priority scheduling used between the queues and round-robin scheduling within each queue. Since that is pretty complicated, let's say we were to replace that scheduling algorithm with single-queue preemptive Shortest Job First scheduling (SRTF).
a) Assuming a correct and successful implementation, what impact on **average turnaround time** would you expect the modified scheduler to have, assuming typical interactive computing workloads? (Note, address both the original and new scheduling algorithms in your answer.)
b) Assuming a correct and successful implementation, what impact on the **worst case maximum waiting time** would expect the modified scheduler to have? (Again, address both.)
c) Assuming you were able to get enough expert programming help (say your classmates in this course), would you expect to be successful in implementing the SRTF scheduler? Why or why not?

Chapter 6

2. Consider the following implementation of a "fair" counting contest where "i" and "mutex" are shared between the processes:

```
Process A                Process B
while (i < 10)             while (i > -10)
{                          {
  wait(mutex);               wait(mutex);
  i = i + 1;                 i = i - 1;
  signal(mutex);             signal(mutex);
}                          }
printf("A wins");          printf("B wins");
```

Note that since reading variables is atomic, the tests of i are "safe" even though they are outside the critical sections. Suppose that the processes are started with i initialized to 0 and mutex to 1. You may make no assumptions about the scheduling algorithm used.
a) Is it obvious which process will win? Explain why or why not.
b) Explain what happens if process B decides to cheat by omitting the signal(mutex) call. Does this guarantee that B will win? Why or why not?
c) Explain what happens if B omits both the wait(mutex) and signal(mutex) calls. Does this guarantee that B will win? Why or why not?
d) What is the best way for B to cheat (by adding, removing, or moving wait() and signal() calls in its own code only)? Explain why.

3. Assume that a finite number of resources of a single resource type must be managed. Processes may ask for a number of these resources and —once finished—will return them. As an example, many commercial software packages provide a given number of licenses, indicating the number of applications that may run concurrently. When the application is started, the license count is decremented. When the application is terminated, the license count is incremented. If all licenses are in use, requests to start the application are denied. Such requests will only be granted when an existing license holder terminates the application and a license is returned.
The following program segment is used to manage a finite number of instances of an available resource. The maximum number of resources and the number of available resources are declared as follows:

```
#define MAX RESOURCES 5
int available resources = MAX RESOURCES;
```

When a process wishes to obtain a number of resources, it invokes the decrease_count() function:

```
/* decrease available resources by count resources */
/* return 0 if sufficient resources available, */
/* otherwise return -1 */
int decrease_count(int count) {
    if (available resources < count)
        return -1;
    else {
        available resources -= count;
        return 0;
    }
}
```

When a process wants to return a number of resources, it calls the increase_count() function:

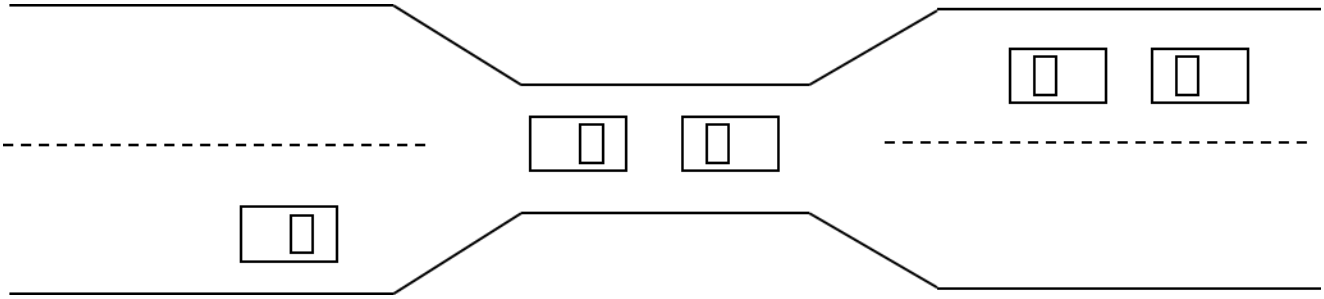
```
/* increase available resources by count */
int increase_count(int count) {
    available resources += count;
    return 0;
}
```

The preceding program segment produces a race condition. Do the following:
a) Identify the variables involved in the race condition.
b) Identify the location (or locations) in the code where the race condition occurs and explain how the race condition can occur.
c) Using a semaphore, fix the race condition and explain why it solves the problem.

Chapter 8

4. Consider the traffic deadlock depicted in Slide 7.5 of the Chapter 7 PowerPoint presentation and shown below. There are 2 resources (the 2 sections of bridge, SectionLeft and SectionRight)

and 2 cars involved (CarLeft and CarRight).
a) Show that the four necessary conditions for deadlock indeed hold in this example.
b) State a simple rule for avoiding deadlocks in this system.



5. Consider the following snapshot of a system:

	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P0	0	0	1	2	0	0	1	2	1	5	2	0
P1	1	0	0	0	1	7	5	0				
P2	1	3	5	4	2	3	5	6				
P3	0	6	3	2	0	6	5	2				
P4	0	0	1	4	0	6	5	6				

Answer the following questions using the banker's algorithm:
a) What is the content of the matrix Need? Show your work.
b) Is the system in a safe state? Remember that you must find a safe sequence in order to answer this question. Show your work.
c) If a request from process P1 arrives for (0,4,2,0), can the request be granted immediately? Show your work. (Note that you must use the Banker's algorithm here.)