**Phase 1:**

1. Can we make the assumption that the ready queue is always larger than or equal the total number of processes in the input file?

Yes, you can.

1. Is it OK to let the scheduler be responsible for decrementing the remaining time of the running process as it is synchronized with the clk? Or should the process decrement its remaining time?

Both are acceptable.

1. How can I calculate STD (standard deviation) of WTA?

According to this formula where x represents the random variable (WTA).

Explanation: Suppose you have 4 processes (A, B, C and D).

1. You calculate the WTA of A, B, C and D (say WTA1, WTA2, WTA3 and WTA4)

2. You then calculate the average of these four values (say X)

3. You then subtract each value from this average, square it, and add them together. (WTA1-X)^2 + (WTA2-X)^2 + (WTA3-X)^2 + (WTA4-X)^2

4. Divide what you computed in (3) by the number of processes which is 4. What you computed so far is the variance.

5. Standard deviation is the square root of the variance.

1. How can we calculate the CPU utilization in the scheduler?

CPU utilization is simply the fraction of time when there were actual processes that are running in the system. If at the end of our simulation, the clock elapsed 20 seconds divided as follows: 15 seconds for useful work (running processes) and 5 seconds (waiting for a process to arrive or idle for any reason), then the CPU utilization should be (15/20)\*100.

**Phase 2:**

1. Can the size of a single process exceed the maximum memory size?

No, it can’t.

1. It's mentioned in the document that we have to allocate memory space for each process as it enters the system. I am a little confused about what this means. Should we allocate the memory as the process *arrives to the scheduler* or when *it's this process turn to be forked*?

Before you fork any process, you have to check first whether it can be allocated or not.

* If there is enough memory space for this process, then memory is allocated and it is forked and scheduled.
* If there is not enough memory space for this process, then it should be pushed to a waiting list until another process finishes and its memory is deallocated from the system. You then check the waiting list again if there is any process that needs to be allocated.

1. In regards to the previous question, what order should I adopt in this waiting list if a process is deallocated and there are two processes that were in the waiting list and only one can be allocated? Should I schedule the oldest or the shortest?

It’s up to you. Both designs are acceptable.

1. In regards to the previous question, how will the waiting time be computed for such a process?

Waiting time is the time that the process has been waiting SINCE its arrival. So, all the clock cycles where the process kept waiting for a memory space will be counted in waiting time IN addition to the waiting time due to scheduling or pre-emption.

1. There is a mistake in the sample output of Phase 2, right?

Yes, this sample was just a demonstration of the output format. For the correct values:

#At time x allocated y bytes for process z from i to j

At time 1 allocated 200 bytes for process 1 from 0 to 255

At time 3 allocated 170 bytes for process 2 from 256 to 511

At time 6 freed 170 bytes from process 2 from 256 to 511

At time 10 freed 200 bytes from process 1 from 0 to 255

1. Should the process be deleted from memory in switching and inserted again when resumed or be deleted only when it finishes?

Of course, it should be deleted when it finishes. Otherwise, it stays allocated in the memory even if it is not running right now (due to a RR scheduling for example)

1. In the allocation of the buddy algorithm should we always allocate the first suitable block?

No, the smallest possible block. If there exists more than one block that satisfies this criteria, you can take any of them.

**Phase 3:**

1. Are we restricted to use message queues in the producer consumer problem?

No, you are not restricted to that. You can use any IPC communication or message passing method (message queues, semaphores or shared memory).

1. Which code should prompt the user for the size of the buffer? Producer or Consumer?

Let’s define a simple macro #define N 10 in both codes of the producer and consumer. No need to prompt the user for any size of the buffer.

1. In multi-producer mode, say I will run two producers, how can I differentiate between them to test that my code is running perfectly?

Let the numbers generated from one producer start from some value (say 1), and the second producer start from another value (say 100), and the third start from another value (say 200). So, we can keep track of the values in the buffer.

1. What do you mean by adding sleeps in the code to simulate the different speeds of the producer and consumer?

Add a sleep(5) in the producer (e.g. after producing an item), and sleep(3) in the consumer (e.g. after consuming an item) to simulate a faster producer. Repeat the same experiment but with a faster consumer.

1. In phase 3, can I use semaphores and shared memory only without message queues?

Refer to question 8. You can use whatever you want.

1. How can I clear the IPC resources in case of multiple producers and consumers?

There are many options. You can track the number of processes (whether they are producers and consumers) and let the final process be the responsible for destroying the shared buffer and other IPC objects. You can alternatively write this as a post-processing script that you run after terminating all producers and consumers in the system. You can think of any other way.

1. Phase 3 doesn't depend on any prior phase, is that true?

Yes.

**Miscellaneous:**

1. My code has a problem. It does not enter into section X/ The process enters the CS although a down semaphore is there/ The output is random/ The last process has an issue and terminates weirdly/ The first process has an issue and delays some time/ etc.

My only advice is to DEBUG, DEBUG and DEBUG. You are the expert of your own code. Also note that 95% of your code problems can be easily spotted by debugging. You need to learn debugging skills. You are in your third year and should learn to solve your code problems on your own.

1. One of my teammates is not answering on their phone. What should I do?

Whatsapp her, maybe? \*sarcastically speaking\* Guys, come on. I am not your nanny!

During the discussion, we will evaluate each team member based on their understanding and based on their effort in the project. If one team member is found to be extremely hesitant and keeps reciting what his teammates told him to say during the discussion, this will extremely harm their grade.