161044083 **Galip Tayfun Saygılı**

CSE 344 MIDTERM

REPORT FILE

In this homework, at first, the first process starts with creating and initializing the shared memory and the semaphores, and then it initializes all the static values.

The first process creates all the other processes and waits for equalizing the total number of processes to the sum of the total number of nurses, vaccinators and citizens. The first process signals to each process and waits for a signal back to make sure that each process is ready to go on.

For nurse processes, each process get the offset number of the cursor for itself from while condition, in a mutex semaphore. Then waits for the buffer with buffer semaphore, increases the vaccine number in vaccine mutex semaphore and posts vaccine semaphore to inform vaccinators waiting for vaccines in a semaphore.

For vaccinator processes, the total number of being in while loop is citizen number multiplied by the total number of the dose for each citizen. And they check that with injection mutex semaphore. Processes in while loop gets the semaphores of the both vaccines, gets the operation mutex semaphore and starts checking for the number of total doses injected for each citizen from the shared memory. For bonus part, I made a restriction that younger citizens are not able to get their 1st dose earlier than their elders. When the citizen is picked, a signal is sent to the selected citizen. (Beware of that the operation mutex semaphore is not released yet.)

For citizen processes, each citizen is waits for a signal and then gets the vaccine mutex semaphore and decreases the number of both vaccines, releases both mutex semaphores (vaccine and operation) back and posts to the buffer twice. Each citizen repeats this procedure t times. (total number of the doses he/she will get)

After program operates, parent process waits for all the children, releases all the resources and exit the program.

In case of SIGINT signal happens, All nurses closes their file descriptors, all child processes exits the program, then the parent process releases all the resources and exits the program by informing the user.