**CECS 326-01**

# Operating Systems

## Matthew Zaldana (ID 027008928)

## Assignment 5

### Due Date: 11/9/2021

### Submission Date: 11/8/2021

# Program Description

1. The program is supposed to create 3 child processes that write to a shared memory segment. Each child process calculates seats left in a bus in its respective bus class structure pointer. However, they all write to the shared memory segment. Nonetheless, here come the beauty of semaphores to the rescue in this assignment. The semaphores allow a designated limit for each shared memory to be written to it at the same moment. For this reason, the output is the same every time, the executable file is run, and there are no random outputs from each shared memory.

2. The booking.h file is the header file for a structure called BUS. It holds all the variables necessary for the structure that the other two programs use. Shmp2.cpp is considered the master file. It creates process ids, a process numbers, shared memory segment ids, a semaphore name and forks 3 children processes. I added the semaphores opening, unlinking, and closing functions to this file, aka, the parent file so that the parent can control the moment all shared memory segments in the child process end. Shmc2.cpp executes the seats idea. It stores the process name, id, and all the other arguments passed from when a child process was created in Shmp2.cpp. It has a function called sell\_seats that instantiates a random seed generator that sells the number of seats the child process contains and outputs the current value at that moment to the screen. Although the amount of time it waits to output to the screen is random, it calculates the seats left based on the semaphore in question that is writing to the shared memory in that moment. Once it returns a valid close call to shmp2, then shmp2 unlinks the semaphore from the shared memory segment, invalidates the shared memory id, and closes the semaphore and terminates.