System Programming HW4

Zafer ALTAY 161044063

May 2021

1 Problems and Solutions

First, I will explain the thread function that reads the assignments and adds them to the queue.This thread will continue to read all the assigments in the file as long as we have money.However, the queue is limited by the number of students. I used semaphores to achieve this synchronization.Two semaphores count the free space and the full space in the queue, so it works synchronously to avoid problems if the queue is completely empty or completely full.If the queue is completely empty, the student waits for homework, if the queue is full, this thread waits for at least 1 homework to be done.I used array for the tail structure. I keep the tail structure by checking the index with variables named head and tail.

Second, let me talk about the student thread.Whichever student's thread is, it works according to that student's pace. I synchronized it using a semaphore.According to the speed of the student, sleep decreases the money. Deletes the assignment from the Queue. Sets the queue semaphores and waits for new assignment.It sets itself as available. I keep the availability of the student in parallel arrays.Thus, a student becomes eligible if available.

Now let me explain the main thread, first I read the file with student information and parse it.I keep the name, price, speed and availability status of the students in parallel arrays.In this way, I facilitate access.Then I init the semaphores I use in other threads and create the threads.The main thread first reads the first assignment in the queue in a loop, and looks for a suitable student based on the type of assignment.I check different suitability within if conditions. If speed and quality are important, I look for maximum value, if price is important, I look for minimum value.In addition to these, I also stipulate whether the students are avaliable or not, so that I look like for avaliable minimum price or avaliable maximum quality.While I am searching for a suitable student, I search according to the remaining money. This detail is important.If I do not have enough money among the eligible students, I will leave the program as stated.If I find it suitable, I set the semaphore of the thread that I syncronize with semaphores and that thread runs.If the main thread reads anything other than expected assignments (incorrect input, line break, etc.), it terminates the program.If the program will close after the main thread finishes its work, it sets the flag first, and sets the semaphores to terminate the waiting threads.In this way, all processes are terminated, then they are terminated by releasing their own resources.

2 Pass File

In general, I cite the expectations and shortcomings for convenience:

I created as many threads as indicated.

Homework works as desired.

I adjusted the synchronization as desired.

I checked it with Wall and ran it smoothly.

I checked for leaks using Valgrind and there is 1 possibly leak.I searched for the reason on the internet. It looks like this is a valgrid bug.