CS307 HW3 Report

In this homework, we need to create a shared memory in order to manage memory requests coming from multiple threads and grant or decline requests based on free space we have. We also used mutexes and semaphores. In main we initialize firstly the rand() in order to get random integers for memory block sizes. Then, I created and ID array to pass it as a parameter for thread function. I also created threads that are going to be used for allocation purposes. Then, I joined them and then by using dump memory, I printed out the memory array. In my malloc I added nodes to our shared queue with respected id and size. I do not need to use mutex for adding nodes to the queue because when I call my malloc I am already in locked state. Hence, there will not be an inaccuracy.

In server function, we checked the memory size to grant or decline a thread to allocate a memory block. If it is eligible to allocate, then we assign the index to thread_message variable and increment the array index according to the memory block size. Otherwise, we assigned -1. And increment Done_threads value by 1 when a thread finishes its job. This is for the while loop continuity

In thread function we generated a random number, then send it as a parameter to my malloc function. Then block by sem list I will release the block after I have done the memory assignment to the character array in thread function Then if message related to id is -1 I print not enough memory I am using another mutex for only guaranteeing no interruption in cout statement. Otherwise, I am filling the memory array with 1 in range of its respected block of memory, and I will push the queue node of respected memory block in server function and make the necessary changes for thread message to assign the memory indexes. After that since I have finished my job with thread I will up the semaphore.

I have assigned two shared variables and a shared mutex in addition. They have helped me to control index assignment, loop control, and statement controls. I think it was insightful to represent memory allocation with threads to understand and implement the concepts further

Yours Sincerely
İlker Gül