CS 307 HW3 REPORT

Ayşegül Rana Erdemli 26744

First, we create an array for threads and an array for thread ids. We use thread id's while creating the threads, we send them to the thread function in each creation. In the thread function, we convert the thread id to integer in order to use it later. We generate a random integer for the thread's request of memory size. We call my malloc function and inside that function, we use mutex before and after accessing the shared queue and we insert the request of our thread to the queue. Then we block our thread until it is unblocked, we block it because the request may not be read yet. It will be read in the server function which has a loop that continuously checks if the queue is empty or not. If the queue is not empty, it gets the next request in the queue and pops it. Then if the request is suitable for the remaining space in the memory array, it writes the index that the thread can start its allocation to the message thread. If it is not suitable, it leaves the message -1 so that the thread would understand there's no space left for it. Then the server function signals to the thread which its request is processed, and our thread will be unblocked. We lock the mutex in order to access the shared data structures memory and message arrays. We check the message server gave us, if it is not -1 we fill the array starting from the index that thread message array gave until we reach to our requested size. If the message is -1, we simply cannot fill the array. After all threads are done with the operation, we join them in the main thread and we destroy the semaphores. Also, we join the server thread after we break its loop by cont=false. It doesn't need to loop anymore. Then the memory array and the memory indexes are printed.