Mert Şaşmaz

In main globe, I created an array called thread_id with the scope of NUM_THREADS and threads with the thread function. After that, the threads are joined in main function. Later, dump_memory prints out the memory. The loop in that function iterates as much as the memory size. In my_malloc function, a new node is created then pushed in the queue while mutexes are used to lock and unlock the process. In the server function, the loop runs as long as the program runs. If queue is not empty, the first node gets taken. And if the node's size is less than the MEMORY_SIZE, thread message[i] gets the value of index value then I up the semaphore. In thread function, a random variable is created with rand() % ((MEMORY_SIZE / 6)+1) method for size value and later semaphore gets down. If the thread message value is -1, an error message gets printed. If not -1, all of the 10 threads are allocated to their ids.