CS307 HOMEWORK 3
Dilara Nur Memiş- 27868

ADDITIONAL GLOBAL VARIABLES:

int charValues[NUM_THREADS]:
To easily update memory array, I created char values of each thread id and put them into an array.

int remainingMemory:
I define this variable to keep track of the remaining memory easily.

MAIN FUNCTION:
I use srand(time(NULL)) to generate different random integers for each thread.
In the main, I created an array of threads. I created and joined them one by one. After joining all of them, I cancel server thread.
The rest is the same with the shared template.

SERVER FUNCTION:

It runs in a continuous while loop until all threads are finished.

It first checks if the shared queue is empty or not. If it is not empty, it first locks the shared mutex. it processes the waiting requests one by one. If the size of the request is larger than remaining memory, it updates the thread messages array's requesting thread index as -1. Else, it updates the thread messages array's requesting thread index as start address. In either case, it up the value of semaphore of related thread to unblock thread. Finally, it releases mutex.

THREAD & MY_MALLOC FUNCTIONS:

It generates a random integer and call my_malloc() function. My_malloc() creates a node with size and thread id and pushes it to the end of the shared queue. After returning from my_malloc, thread downs it semaphore and blocks.

When the server function makes an up on its semaphore, it unblocks and continue execution. It first tries to acquire shared mutex. If it could acquire, it checks corresponding index of thread message array. If the value is -1, it prints a message to indicate there is not enough memory to satisfy its demand. Else, it updates related range in the memory array. Finally, it releases the mutex.

DUMP_MEMORY FUNCTION

I print the content of the memory at the end of the program.