Mert Ilgınoğlu, 26774

CS307
Homework 3 Report

In main, we first create arrays that store thread ids and threads. Then following that, in a for loop we assign values to these arrays. Afterwards, we initialize semaphores, memory and start the server in init() function. When we are done with that, we move onto multithreaded part and start our threads with thread_function().

In thread_function, we first assign a random number between 1 and MEMORY_SIZE/6 to thread's size requirement. Then we call malloc. Malloc pushes a node in the queue that contains thread's id and size. When we are done with malloc, we down the semaphore of the thread in order to wait for server's response for our memory allocation request.

Server function is continuously running until all of the threads are finished their requests. This function takes requests from the queue which we pushed nodes in malloc function. It takes the front of the queue, check its size and see if it can fit into the memory. If it does not fit, assigns the thread_message[thread_id] with -1. If it can fit, it assigns the index of the memory which we last used and increases the index by the size. Then ups the semaphore of the thread.

Now we are back to thread_function. We look at thread_message array, if corresponding value is -1 then it means we didn't have enough memory, hence we print an error message. If it is not -1, then it means we can fit size into memory. We start filling from index with thread_id until we do this step size times.

Now we are finished with thread_function, we join all of the threads and dump all of the memory. We finish with printing the starting indices of threads.