David Acevedo
Kyle Thakker

Asst1: Memory Allocation++

Running the program:

To execute the program, run the make file and execute the output file. The makefile includes the mymalloc.h where we define the malloc and free functions.

Description:

For this program, we implement malloc and free library calls with the capability to detect common programming and usage errors. Both functions use the “first free” algorithm to select free blocks to allocate. A static char array of size 5,000 bytes is used to simulate main memory in a computer. A struct called “Node” is used to represent a metadata node that gives information on the following block of memory including its state and the size of the block.

In the malloc function, there is a check to test whether the size of the pointer trying to be allocated is greater than the total memory minus the size of the metadata node. If the check is true, an error statement is printed and the program returns without allocating any memory. However, if the check returns false, the block of memory is initialized and the memory is then looped through. If a block of memory is being used the loop then points to the next block in memory. If the block is free, and the size of the block can fit a metadata node, that node is then initialized and its state and size are defined. If the size of the block is just big enough to store the data trying to be allocated, no node is created and the state is updated to show that the block is being used. Afterwards a pointer to the block of memory is returned to the programmer.

In the free function, there are three checks to test to see if the free function can free the pointer. The checks are if the pointer is null, if the pointer is within the heap, or if the pointer is already freed. If any of these checks are true, the program prints an error statement and returns without freeing the pointer. After these conditions are checked, the pointer is freed, and if the blocks before or after are also free, they are merged with the recently freed block.

Testcase data:

	Workload A	Workload B	Workload C	Workload D	Workload E	Workload F
Total Time	~ 1.2 seconds	~ .0022 seconds	~ .07 seconds	~ .36 seconds	~ 1.3 seconds	~ .6 seconds
Average Time	~ .012 seconds	~ .000022 seconds	~ .0007 seconds	~ .0036 seconds	~ .013 seconds	~ .006 seconds