Michael Nitzsche, Nicholas Harras Systems Programming Project 1

In this project, we were tasked with creating improved versions of malloc and free which could detect and prevent user errors. Mymalloc.c accomplishes this by delegating dynamic memory from a 5000 byte static array through two new functions, mymalloc and myfree.
Mymalloc and myfree are run on top of existing code through the use of a header file macro which replaces all instances of malloc and free with mymalloc and myfree, so the user does not notice it is there outside of its superior error handling capabilities.

Metadata in this block of memory takes the form of a linked list consisting of three fields: a two byte index of the “next” field of next node, a two byte quantity of memory bytes contained in the node and the memory itself. The first four bytes are a dummy node with a pointer to the head and 0 memory in order to ensure the head location is always known. Insertion of the first node is a special case in which the memory field at bytes six and seven is set to the amount requested. Thereafter, all nodes are inserted by iterating through the linkedlist until there is a “0” next field indicating the end of the list, or there is a gap between the end of the current node’s memory and start of the next node large enough that the requested memory block can be inserted. After insertion, the user receives a pointer to the 4th byte into the new node, representing the start of memory. If the block runs out of memory or the initial request is too large to accommodate, error messages are displayed and a null pointer is returned.

To free memory, myfree iterates through the linkedlist until the pointer being freed is 4 bytes above the start of a header, indicating that it is a pointer intentionally allocated. If the user tries to free a value not in the 5000 byte array or a pointer between two nodes, an error message is displayed and the function is terminated. To free a pointer, the next field of the previous node is simply set to the following node, removing the freed node from the linked list. While it is not strictly necessary, the header values of the freed node are then set to 0 for ease of debugging.